VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

ROZPOZNÁVÁNÍ OSOB PODLE OBLIČEJE S VYUŽITÍM NEURAL COMPUTE STICK

FACE RECOGNITION WITH ACCELERATION ON THE NEURAL COMPUTE STICK

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **MATEJ HORNÍK**
AUTHOR

VEDOUCÍ PRÁCE **Ing. TOMÁŠ GOLDMANN,**
SUPERVISOR

BRNO 2022

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav inteligentních systémů (UITS) Akademický rok 2021/2022

Zadání bakalářské práce

Student: **Horník Matej**
Program: Informační technologie
Název: **Rozpoznávání osob podle obličeje s využitím Neural Compute Stick**
Face Recognition with Acceleration on the Neural Compute Stick
Kategorie: Databáze

Zadání:

1. Seznamte se s problematikou rozpoznávání osob podle obličeje.
2. Zjistěte, jaké algoritmy se pro rozpoznávání obličeje v současné době používají a porovnejte jejich vlastnosti a v případě i výkonnost.
3. Navrhněte vestavěný systém, který se bude skládat z kamery, výpočetní jednotky a akcelerátoru Neural Compute Stick. Pro rozpoznávání osob podle obličeje zvolte volně dostupný algoritmus, který akcelerujte pomocí Neural Compute Stick.
4. Navržené řešení sestavte a implementujte obslužný software.
5. Proveďte experimenty zaměřené na zhodnocení úspěšnosti rozpoznávání osob a na vyhodnocení zrychlení algoritmu pomocí akcelerátoru. Na základě zjištěných poznatků navrhněte rozšíření.

Literatura:

- MASI, Iacopo, et al. Deep face recognition: A survey. In: *2018 31st SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)*. IEEE, 2018. p. 471-478.
- TRIGUEROS, Daniel Sáez; MENG, Li; HARTNETT, Margaret. Face recognition: From traditional to deep learning methods. *arXiv preprint arXiv:1811.00116*, 2018.
- DI NARDO, E.; PETROSINO, A.; SANTOPIETRO, V. Embedded Deep Learning for Face Detection and Emotion Recognition with Intel Movidius (TM) Neural Compute Stick. 2018.

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 a 2.

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

Vedoucí práce: **Goldmann Tomáš, Ing.**
Vedoucí ústavu: Hanáček Petr, doc. Dr. Ing.
Datum zadání: 1. listopadu 2021
Datum odevzdání: 29. července 2022
Datum schválení: 30. června 2022

Abstrakt

Táto bakalárska práca sa zaoberá súčasnými technikami na rozpoznávanie osôb podľa tváre. V súčasnosti sa používajú konvolučné neurónové siete na rozpoznávanie tvárí. V tejto práci budú konvolučné neurónové siete popísané a taktiež budú porovnané súčasne architektúry konvolučných sietí, ktoré sa využívajú na rozpoznávanie tvárí. Cieľom bude vytvoriť vstavaný systém, ktorý sa bude skladať z kamery, výpočetnej jednotky a akcelerátora Neural Compute Stick. Systém bude rozpoznávať osoby podľa tváre s voľne dostupným algoritmom.

Abstract

This bachelor thesis deals with current techniques for recognizing people by face. Convolutional neural networks are currently used for face recognition. In this work, convolutional neural networks will be described and also the architectures of convolutional networks used for face recognition will be compared. The goal will be to create a built-in system that will consist of a camera, a computing unit and a Neural Compute Stick accelerator. The system will recognize people by face with a freely available algorithm.

Kľúčové slová

rozpoznávanie tváre, neurónová sieť, konvolučná neurónová sieť, strojové učenie, umelá inteligencia, Neural Compute Stick.

Keywords

face recognition, neural network, convolutional neural network, machine learning, artificial intelligence, Neural Compute Stick.

Citácia

HORNÍK, Matej. *Rozpoznávání osob podle obličeje s využitím Neural Compute Stick*. Brno, 2022. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Tomáš Goldmann,

Rozpoznávání osob podle obličeje s využitím Neural Compute Stick

Prehlásenie

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

.....
Matej Horník
29. júla 2022

Poďakovanie

Týmto by som chcel veľmi poďakovať môjmu vedúcemu práce Ing. Tomášovi Goldmannovi, za skvelé vedenie, ochotu, cenné pripomienky a rady, ktoré mi poskytol pri riešení tejto práce.

Obsah

Kapitola 1

Úvod

Biometrický systém založený na rozpoznávaní tvári je v dnešnej dobe často využívaný v rôznych oblastiach. Najčastiejšie sa využíva ako bezpečnostný prvok pre prístup do zabezpečených systémov ako je napríklad prístup do mobilného zariadenia. Rozpoznanie osoby na základe jej tváre je pre človeka v celku ľahká úloha. Pre počítač je to komplexná úloha, ktorá vyžaduje niekoľko krokov a zložité techniky pre správne rozpoznanie tváre. V dnešnej dobe algoritmy na rozpoznávanie tvárí dosahujú lepšiu úspešnosť ako ľudia na niektorých úlohách pri rozpoznávaní tvárí. Algoritmy sú založené na konvolučných neurónových sieťach, ktoré sa stali populárnymi vďaka pokroku výpočetného hardwaru. V minulosti boli na rozpoznávanie tváre využívané techniky, ktoré boli založené ručnom vytváraní vlastností o danej tváre, no však tieto metódy nedosahujú kvalitné výsledky oproti metódam založených na neurónových sieťach. I napriek významnému pokroku v rozpoznávaní ôsob podľa tváre sa výsledky v experimentálnom prostredí podstatne líšia od výsledkov v reálnom prostredí[22]. Problémy sú spôsobené zle nasvietenými tvárami, nízkou kvalitou snímok, extrémnymi natočenými tvárami, variabilitou výrazov, zakrytými časťami tváre, a inými problémamy, ktoré datasety používane na trénovanie obsahujú. V poslednej dobe je publikovaných mnoho prác, ktoré sa snažia s týmito problémami nejakým spôsobom vysporiadať.

S prudkým rastom mobilnej výpočtovej techniky a aplikácií internetu vecí (angl. *Internet of things*) sa k internetu pripájajú miliardy mobilných zariadení a zariadení internetu vecí, ktoré generujú obrovské množstvo dát na okraji siete. Výsledkom je, že zhromažďovanie obrovských objemov dát v cloudových dátových centrách, spôsobuje extrémne vysokú latenciu a využitie šírky pásma siete. Preto vznikajú nové riešenia ako tento problém riešiť. Je potrebné posunúť hranice umelej inteligencie až na okraj siete, aby sa naplno využil potenciál veľkých dát. Tento nový prístup sa nazýva *Edge computing* a je mu v práci venovaná jedna kapitola.

Cieľom tejto práce je preštudovať moderné riešenia umožňujúce detekciu a rozpoznávanie osôb a využitím akcelerátora Neural Compute Stick 2 akcelerovať algoritmy na rozpoznávanie ôsob. Následne tieto algoritmy implementovať pri vytváraní aplikácie, ktorá bude schopná pomocou kamery detekovať a rozpoznať osoby vo vstupných dátach a zaslať informáciu o detekcii uživateľovi prostredníctvom internetu.

Rozdelenie kapitol je nasledujúce. V kapitole 2 je popísaný základ rozpoznávania osôb podľa tváre a metódy pre rozpoznávanie tváre. V kapitole 3 je popísaná teoretická časť neurónových sietí. Jedna podkapitola popisuje súčasné algoritmy pre rozpoznávanie tváre, ich vlastnosti a výkonnosť. Posledná podkapitola sa venuje datasetom, ktoré sú používané pri trénovaní a evaluácií modelov. V kapitole 4 je popísaný *Edge computing*, sú popísané vlastnosti, rozdiely medzi cloud computingom a príklady použitia v reálnom svete. V po-

slednej kapitole 5 je popísaný návrh aplikácie tak ako aj jej implemntácia. V tejto kapitole je popísaný aj akcelerátor Neural Compute Stick 2 a platforma OpenVINO. V kapitole 6 sa nachádzajú experimty. Experinty sú v prvej sekcii zamerané na zhodnotenie úspešnosti rozpoznávania osôb. Vyhodnotenie verifikácie algoritmov je realizované pomocou ROC (*Receiver Operating Characteristic*) kriviek. V poslednej sekcii tejto kapitoly sa v tejto kapitole nachádzajú experimenty zamerané na vyhodnotenie akcelerácie algoritmov pomocou akcelerátora Neural Compute Stick 2. V poslednej kapitole čož je záver je popísané zhodnotenie práce a taktiež navrhnuté rozšírenie aplikácie pre rozpoznávanie tvárí.

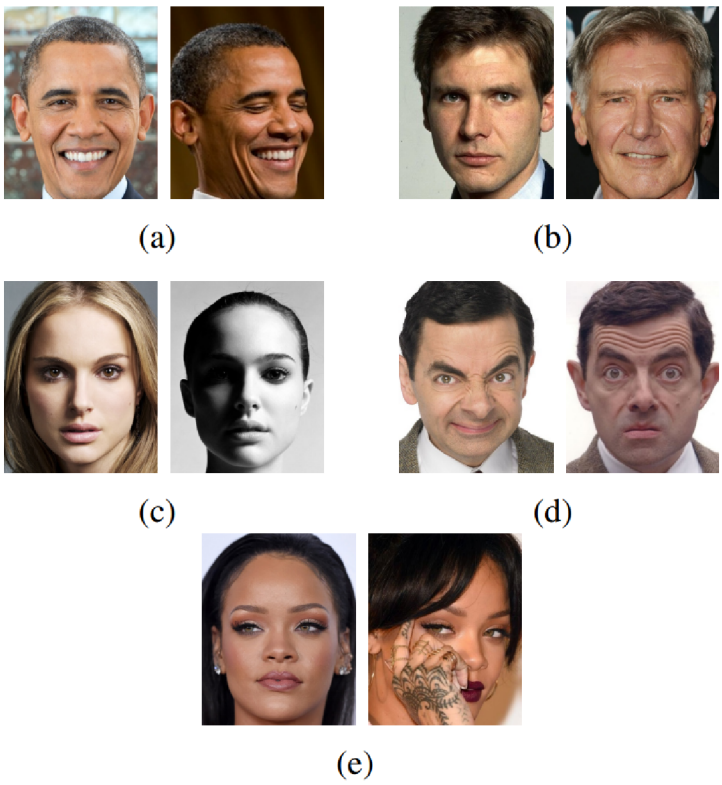
Kapitola 2

Rozpoznávanie osôb podľa tváre

Rozpoznávanie osôb na základe ich tváre je technológia schopná identifikovať alebo overiť ľudskú tvár z digitálneho obrazu alebo videa. Rozpoznávanie tváre bolo prominentnou biometrickou technikou overovania identity a je široko používaná v mnohých oblastiach, ako je armáda, financie, verejná bezpečnosť a každodenný život. Prvé algoritmy na rozpoznávanie tváre boli vyvinuté začiatkom sedemdesiatych rokov. Odvtedy sa ich presnosť zlepšila do tej miery, že v dnešnej dobe rozpoznávanie tváre sa často uprednostňuje pred inými biometrickými spôsobmi, ktoré sa tradične považovali za robustnejšie, ako napríklad odtlačky prstov alebo rozpoznávanie dúhovky. Jedným z rozdielnych vlastností oproti iným biometrickým spôsobom, vďaka ktorým je rozpoznávanie tváre príťažlivejšie ako iné biometrické spôsoby, je jeho nerušivý charakter. Napríklad rozpoznávanie odtlačkov prstov vyžaduje, aby používatelia vložili prst do snímača, rozpoznávanie dúhovky vyžaduje, aby sa používatelia výrazne priblížili k fotoaparátu a rozpoznávanie podľa hlasu vyžaduje, aby používatelia hovorili nahlas. Na rozdiel od toho moderné systémy pre rozpoznávania tváre vyžadujú, aby sa používatelia nachádzali iba v zornom poli kamery a v dostatočnej vzdialenosti od kamery pre požadujúcu kvalitu tváre. Vďaka tejto vlastnosti je rozpoznávanie tváre užívateľsky najpríjemnejšou biometrickou metódou. Znamená to tiež, že rozsah potenciálnych aplikácií rozpoznávania tvárí je širší, pretože môže byť nasadený v prostrediach, kde sa neočakáva, že používatelia budú spolupracovať so systémom, ako sú napríklad sledovacie a bezpečnostné systémy. Medzi ďalšie bežné aplikácie rozpoznávania tváre patrí kontrola prístupu, detekcia podvodov, overenie identity a sociálne médiá.

Rozpoznávanie tvárí je jednou z najnáročnejších biometrických spôsobov pri nasadení v reálnych prostrediach kvôli vysokej rôznorodosti fotografií tvárí v skutočnom svete, tieto typy obrazov tvárí sa bežne označujú ako tváre vo voľnej prírode (angl. *Faces in the wild*). Niektoré z týchto variácií zahŕňajú polohu hlavy, starnutie, ochlpenie tváre, vlasy, rôzne druhy osvetlenia a výrazy tváre. Príklady sú uvedené na obrázku 2.1.

Techniky rozpoznávania tváre sa v priebehu rokov výrazne posunuli. Tradičné metódy sa spoliehali na ručne vytvorené vlastnosti, ako sú okraje tváre a deskriptory textúr, v kombinácii s technikami strojového učenia, ako je analýza hlavných komponentov (angl. *Principal component analysis*), lineárna diskriminačná analýza alebo podporné vektorové modely (angl. *Support vector machines*). Náročnosť vytvárania ručných vlastností, ktoré boli odolné voči rôznym variáciám vyskytujúcim sa v reálnom prostrediach, prinútilo výskumníkom zamerať sa na špecializované metódy pre každý typ variácie, napr. vekovo-invariantné metódy, pózovo-invariantné metódy, iluminačne-invariantné metódy, atď. Tradičné metódy rozpoznávania tváre boli nahradené metódami hlbokého učenia založenými na konvolučných



Obr. 2.1: Typické variácie a) Póza hlavy. b) Vek. c) Osvetlenie. d) Výraz tváre. (e) Objekt pred tvárou[22].

neurónových sieťach. Hlavnou výhodou metód hlbokého učenia je, že ich je možno trénovať s obrovskými datasetmi, aby sa naučili najlepšie vlastnosti na reprezentáciu dát. Dostupnosť veľkého množstva obrázkov tvári voľne dostupných na internete, označovaných ako "vo voľnej prírode"(*faces in the wild*), umožnila zber rozsiahlych datasetov tvárí obsahujúcich rôzne variácie v reálnom svete. Metódy rozpoznávania tváre založené na CNN trénované s týmito datasetmi dosiahli veľmi vysokú presnosť, pretože sú schopné sa naučiť vlastnosti , ktoré sú odolné voči zmenám v reálnom svete. Okrem toho nárast popularity metód hlbokého učenia pre počítačové videnie urýchlil výskum rozpoznávania tváre, pretože konvolučné neurónové siete sa používajú na riešenie mnohých ďalších úloh počítačového videnia, ako sú detekcia a rozpoznávanie objektov, segmentácia obrazu, optické rozpoznávanie znakov, analýza výrazu tváre, vekový odhad atď.

Systém pre rozpoznávanie tváre

Väčšina systémov pre rozpoznávanie tváre sa skladá zo štyroch komponentov. Tieto systémy pracujú nad dvojrozmernými fotografiami tváre, ktoré systém získa buď z konkrétnej fotografie osoby alebo zo snímku z videa na ktorom sa nachádza osoba. Niektoré systémy pracujú aj s trojrozmernými dátami osôb aby zaručili väčšiu bezpečnosť. Jednotlivé časti rozpoznávania tváre:

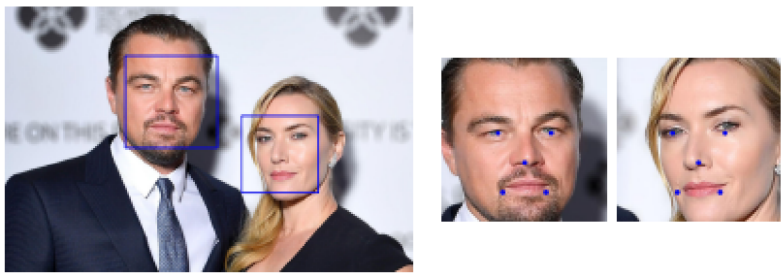
1. **Detekcia tváre** – Detektor tváre nájde polohu tvárí na obrázku a vráti súradnice ohraničujúceho rámčeka pre každú z nich. V prípade videa musí detektor vyhľadať

tváre osôb v jednotlivých snímkoch videa a tým je proces náročnejší. Toto je zobrazené na obrázku 2.2a

2. **Zarovnanie tváre** – Cieľom zarovnania tváre je zmeniť mierku a orezať obrázky pomocou unikátnych bodov umiestnených v obrázku. Tento proces zvyčajne vyžaduje nájdenie orientačných bodov tváre pomocou detektora unikátnych bodov a snaží sa o nájdenie najlepšej afinnej transformácie, ktorá vyhovuje orientačným bodom. Unikátne body tváre sú ústa, nos, oči, obrys atď. Obrázky 2.2b a 2.2c zobrazujú dva obrazy tváre zarovnané pomocou rovnakej sady orientačných bodov. Zložitejšie algoritmy pre 3D zarovnanie tváre môžu tiež dosiahnuť frontalizáciu tváre, to znamená. zmenu polohy tváre na čelnú.

3. **Reprezentácia tváre** – Vo fáze reprezentácie tváre sa hodnoty pixelov obrazu tváre transformujú do kompaktného a diskriminačného vektora. V ideálnom prípade by sa všetky tváre tej istej osoby mali mapovať na podobné vektory a mali by byť výrazne vzdialené od vektoru iných ôsob. Reprezentácia tváre je pravdepodobne najdôležitejšou súčasťou systému rozpoznávania tváre.

4. **Priraďovanie tváre osobe** – Získané príznaky sa porovnajú s ostatnými príznakmi v databáze a vytvorí sa skóre podobnosti, ktoré naznačuje pravdepodobnosť, že patria k rovnakému predmetu. Porovnávanie vektorov sa može vykonať dvoma spôsobmi, euklidovskou vzdialenosťou alebo cosínovou podobnosťou. Existujú dva spôsoby ako overiť danú tvár, verifikácia a identifikácia. Pri verifikácii sa daná tvár porovnáva s inou tvárou a určí sa či sa jedná o rovnakú tvár alebo nie. Pri identifikácii sa daná tvár porovnáva s celou databázou tvárí a výsledkom je identita danej tvári v databázi v prípade úspešnosti.



Obr. 2.2: a) Ohraňujúci rámec tváre. (*Region of interest*) b) a c) Detekované body tváre pre zarovnanie[22].

Metódy pre rozpoznávanie tváre

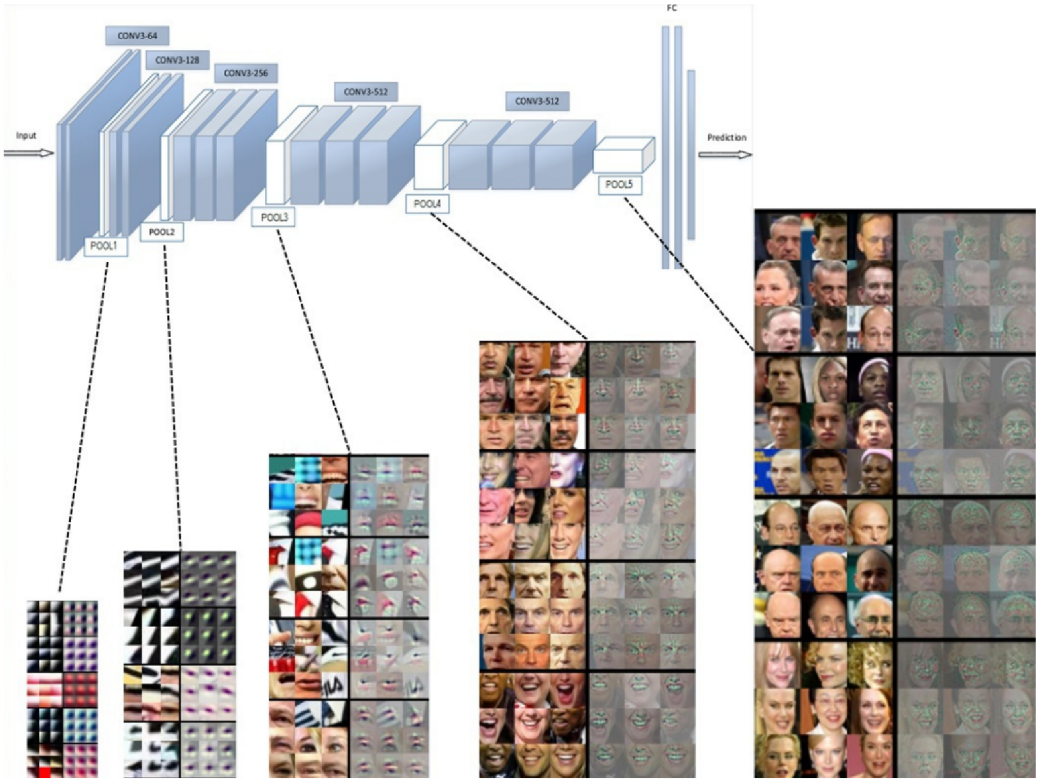
Od začiatku výskumu na ropoznávaní tváre boli použité rôzne prístupy na riešenie tochto problému. Jeden z prvých prístupov bol takzvaný Eigenface prístup, ktorý priniesli M. Turk a A. Pentland [27] v roku 1991. Tento prístup sa radí do kategorie metód založených na vlastnostiach, ktoré využívajú lokálne prvky extrahované na rôznych miestach obrazu tváre (angl. *Feature-based methods*). Holistický prístup bol ďalší, ktorý bol velmi populárny na začiatku 20. storočia. Prístup, ktorý sa v súčasnosti používá na rozpoznávanie tváre

je založený na hlbokých neurónových sieťach. Tento spôsob je vysvetlený v nasledujúcej kapitole.

Holistické metódy – Holistické rozpoznávanie tváre [27] [17] využíva globálne informácie z tvárí na rozpoznávanie. Globálna informácia z tváre je v podstate reprezentovaná malým počtom vlastností, ktoré sú priamo odvodené z pixelových informácií obrázkov tváre. Tento malý počet vlastností zreteľne zachytáva rozdiely medzi rôznymi individuálnymi tvárami. V týchto metódach je každý obraz tváre reprezentovaný ako vysokorozmerný vektor, ktorý zreťazí hodnoty šedej všetkých pixelov v obraze. Holistické metódy využívajú ako vstup celú tvár. Pri tejto metóde sa využívajú podporné vektorové modely, analýza hlavných komponentov a lineárna diskriminačná analýza.

Metódy založené na vlastnostiach – Metódy založené na vlastnostiach [22] sa týkajú metód, ktoré využívajú lokálne prvky extrahované na rôznych miestach obrazu tváre. Tieto metódy majú tendenciu byť robustnejšie. Uvažujme napríklad o dvoch snímkach tváre toho istého objektu, v ktorých je rozdiel medzi nimi len v tom, že na jednom z nich má osoba zatvorené oči. V metóde založenej na vlastnostiach sa medzi týmito dvoma obrázkami budú líšiť iba koeficienty vektorov vlastností, ktoré zodpovedajú vlastnostiam extrahovaným okolo očí. Na druhej strane v holistickej metóde sa môžu všetky koeficienty vektorov vlastností líšiť. Mnohé z deskriptorov používaných v metódach založených na vlastnostiach sú navrhnuté tak, aby boli invariantné k rôznym variáciám (napr. škálovanie, rotácia alebo posun).

Metódy založené na neurónových sieťach – Metódy hlbokého učenia [22] , ako sú konvolučné neurónové siete, využívajú kaskádu viacerých vrstiev procesných jednotiek na extrakciu a transformáciu vlastností. Učia sa viaceré úrovne reprezentácií, ktoré zodpovedajú rôznym úrovniam abstrakcie. Úrovne tvoria hierarchiu konceptov, ktoré vykazujú silnú invarianciu k póze tváre, osvetleniu a zmenám výrazu, ako je znázornené na obrázku 2.3. Prvá vrstva sa naučí jednoduché vlastnosti a štruktúry tváre. Druhá vrstva sa naučí zložitejšie vlastnosti textúry. Znaky tretej vrstvy sú zložitejšie a začali sa objavovať niektoré jednoduché štruktúry, ako napríklad vysoko premostený nos a veľké oči. Záverom možno povedať, že v hlbokých konvolučných neurónových sieťach sa nižšie vrstvy automaticky učia jednoduché vlastnosti a vyššie vrstvy sa ďalej učia abstrakciu vyššej úrovne. Nakoniec, kombinácia týchto vyšších úrovní abstrakcie predstavuje identitu tváre s bezprecedentnou stabilitou. Konvolučné neurónové siete sú najbežnejším typom metódy hlbokého učenia na rozpoznávanie tváre. Hlavnou výhodou metód hlbokého učenia je, že môžu byť trénované s veľkým množstvom dát, aby sa naučili reprezentáciu tváre, ktorá je odolná voči variáciám prítomným v tréningových dátach. Hlavnou nevýhodou metód hlbokého učenia je, že musia byť trénované s veľmi veľkými množstvami dát, ktoré obsahujú dostatok variácií na zovšeobecnenie pre príklady tvár, ktoré sa nenachádzajú v trénovacom datasete. Na obrázku 2.3 je zobrazená architektára konvolučnej neurónovej siete, ktorej výstupom je vektor vlastností.



Obr. 2.3: Hierarchická architektúra konvolučnej neurónovej siete. Výstupom je kompresovaný vektor vlastností, ktorý predstavuje danú tvár[29].

Kapitola 3

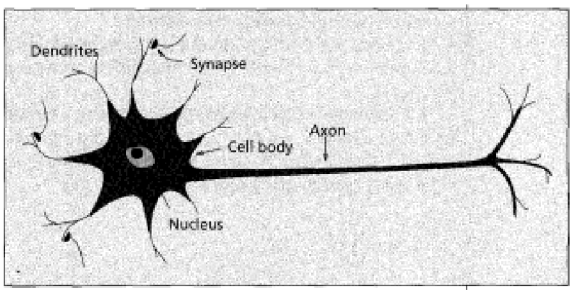
Neurónové siete a ich využitie pre rozpoznávanie osôb podľa tváre

Neurónové siete sú inšpirované biologickým nervovým systémom. Jedná sa o matematický model, ktorý transformuje množinu vstupných premenných do množiny výstupných premenných. Neúronové siete si svoje uplatnenie našli vo veľa oblastiach, v počítačovom videní sa používajú na detekciu a rozpoznávanie tváre v obraze. Neurónové siete majú podstatne menej neurónov oproti ľudskému mozgu, ktorý ich má okolo sto miliárd. Umelé neurónové siete sú schopné spracovať obrovské množstvo údajov a vytvárať predpovede, ktoré sú niekedy prekvapivo. Hlavnou vlastnosťou umelých neurónových sietí je, že sa dokážu „učiť“. Neurónová sieť sa skladá z umelých neuronóv a z hrán, ktoré spájajú jednotlivé neuróny. Neuróny a hrany majú typicky nejakú váhu, ktorá sa postupne upravuje s učením siete. Typicky sú neuróny agregované do vrstiev. Rôzne vrstvy môžu na svojich vstupoch vykonávať rôzne transformácie. Signály prechádzajú z prvej vrstvy (vstupnej vrstvy) do poslednej vrstvy (výstupnej vrstvy), môžu taktiež prechádzať skrytými vrstavami. Signálom sa rozumie hodnota, ktorá je výstupom jedného neuronu.

3.1 Základné prvky neurónovej siete

Informácie v tejto časti sekcie boli prezvaté z [13]. Ľudský mozog pozostáva z neurónov alebo nervových buniek, ktoré prenášajú a spracúvajú informácie prijaté našimi zmyslami. Mnoho takýchto nervových buniek je v našom mozgu usporiadaných spoločne, aby vytvorili sieť nervov. Tieto nervy prenášajú elektrické impulzy z jedného neurónu do druhého. Hlavné časti neurónu sú dendrity, jadro neurónu, axón a synapsie. Dendrity dostávajú impulz zo synapsie susedného neurónu. Dendrity prenášajú impulz do jadra nervovej bunky, ktorá sa nazýva soma. Tu sa elektrický impulz spracuje a potom sa prenesie do axónu. Axón je dlhšia vetva medzi dendritmi, ktorá prenáša impulz z jadra do synapsie. Synapsia potom odovzdá impulz dendritom druhého neurónu. V ľudskom mozgu tak vzniká komplexná sieť neurónov. Model neurónu je zobrazený na obrázku 3.1.

Rovnaký koncept siete neurónov sa používa v algoritmoch strojového učenia. V tomto prípade sú neuróny vytvorené umelo na počítači. Fungovanie umelého neurónu je podobné ako fungovanie neurónu prítomného v našom mozgu.

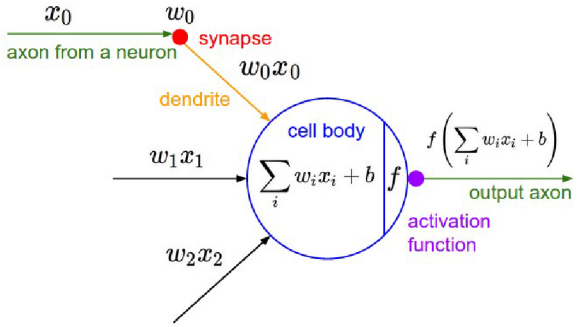


Obr. 3.1: Štruktúra biologického neurónu[13]

Umelý neurón [13] je abstrakciou toho biologického. Vstupom každého umelého neurónu je určitý počet vstupov, ktoré sa zapisujú ako vektor hodnôt x. Každý vstup je vynásobený odpovedajúcou váhou, ktorý sa tiež zapisuje ako vektor hodnôt w. Vynásobené hodnoty sú spočítané a k nim je pripočítaná hodnota prahu (angl. *bias*). Získaná hodnota je poslaná ako vstup aktivačnej funkcii, ktorá je výstupom neurónu a propagovaná ďalším neurónom, ktoré sa nachádzajú v ďalšej vrstve neurónovej siete. Model umelého neurónu je zobrazený na obrázku 3.2. Funkciu neurónu je možné matematicky vyjadriť ako:

$$y = f(\sum_{n=1}^{n} x_i w_i + b), \tag{3.1}$$

kde x_i predstavuje vstupnú hodnotu neurónu, w_i je odpovedajúca váha pre daný vstup, b označuje prah neurónu a funkcia f je aktivačnou funkciou. Hodnoty váh w_i a hodnota prahu b je postupne menená pri trénovaní neurónovej sieti.



Obr. 3.2: Model umelého neurónu[7].

Aktivačná funkcia alebo taktiež známa ako prechodová funkcia je funkcia určujúca výstup umelého neuronu. Používajú sa na transformáciu výstupu neurónu. Po tejto transformácii môže byť výstup použitý ako vstup pre ďalšiu vrstvu.

Existujú dva druhy aktivačných funkcií, jednou skupinou sú lineárne funkcie a druhou nelineárne funkcie. Pri viacvrstvových neurónových sieťach sa využívajú nelineárne aktivančné funkcie. Dôvodom je, že neurónová sieť obsahujúca len lineárnú aktivačnú funkciu by bola ekvivalentná jednému neurónu alebo jednej vrstve lineárnych neurónov.

ReLU (angl. *Rectified Linear Unit*) [1] je momentálne najpoužívanejšia aktivačná funkcia na svete. Používa sa takmer vo všetkých konvolučných neurónových sieťach alebo v hlbokom učení. Funkcia je definovaná vzorcom $f(x) = max(0, x)$, kde x značí výstup neurónu pred aktivačnou funkciou. Výhodou oproti ostatným aktivačným funckiám je to, že

zlepšuje rýchlosť trénovania u neurónových sietí s konvolučnými vrstvami. Problém je však v tom, že všetky záporné hodnoty sa okamžite stanú nulovými, čo znižuje schopnosť modelu správne sa prispôsobiť dátam pri trénovaní. ReLU sa bežne používa modifikována, často využívanou variantou je Leaky ReLU. Modifikuje výpočet pre záporné hodnoty, kde záporným hodnotám priraďuje malý pozitivný sklon. Existujú aj ostatné aktivačné funkcie ako sú napríklad sigmoid ($f(x) = \frac{1}{1+e^{-}x}$), softmax ($f(x) = \frac{e^x}{\sum e^x}$) alebo hyperbolický tangent($f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$), kde x značí výstup neurónu pred aktivačnou funkciou.

3.2 Trénovanie neurónových sietí

Informácie v tejto podsekcie boli prezvaté z [14]. Neurónové siete sú trénované (alebo sa učia) spracovaním príkladov, z ktorých každý obsahuje známy „vstup“ a „výsledok“. Medzi nimi sa tvoria pravdepodobnostné vážené asociácie, ktoré sú uložené v dátovej štruktúre samotnej siete. Trénovanie neurónovej siete z daného príkladu sa zvyčajne vykonáva určením rozdielu medzi spracovaným výstupom siete (často predpoveďou) a cieľovým výstupom. Tento rozdiel je chyba. Sieť potom upraví svoje váhy podľa pravidla trénovania a pomocou tejto chybovej hodnoty, ktorá sa vypočíta pomocou chybovej funkcie ktorá je popísaní v sekcii 3.2.2. Postupné úpravy spôsobia, že neurónová sieť bude produkovať výstup, ktorý sa čoraz viac podobá cieľovému výstupu. Po dostatočnom počte týchto úprav je možné trénovanie na základe určitých kritérií ukončiť. Toto je známe ako učenie pod dohľadom (angl. *supervised learning*).

Tieto systémy sa „učia“ vykonávať úlohy zvažovaním príkladov, zvyčajne bez toho, aby boli naprogramované pravidlami špecifickými pre danú úlohu. Napríklad pri rozpoznávaní obrázkov sa môžu naučiť identifikovať obrázky obsahujúce mačky analyzovaním vzorových obrázkov, ktoré boli manuálne označené ako „mačka“ alebo „žiadna mačka“, a pomocou výsledkov identifikovať mačky na iných obrázkoch. Robia to bez toho, aby vopred vedeli o mačkách, napríklad, že majú srsť, chvosty, fúzy a atď. Namiesto toho automaticky generujú identifikačné charakteristiky z príkladov, ktoré spracúvajú. Ostatné metódy učenia sú učenie bez učiteľa (unsupervised learning) a posilňovacie učenie (angl. *reinforcement learning*).

3.2.1 Inicializácia váh

Pod inicializáciou váh sa rozumie proces, keď pre novú neurónovú sieť sú určené hodnoty váh jednotlivých neurónov pred samotným trénovaním. Inicializácia váh je dôležitým krokom pri vývoji modelov neurónových sietí s hlbokým učením. Historicky inicializácia váh zahŕňala použitie malých náhodných čísel, hoci v poslednom desaťročí boli vyvinuté špecifickejšie metódy, ktoré využívajú informácie, ako je typ aktivačnej funkcie, ktorá sa používa, a počet vstupov neurónu. Tieto lepšie prispôsobené metódy môžu viesť k efektívnejšiemu trénovaniu modelov neurónových sietí.

Xavierová inicializácia je súčasný štandardný prístup k inicializácii váh vrstiev a neurónov neurónovej siete, ktoré využívajú aktivačnú funkciu Sigmoid alebo TanH. Je pomenovaná po Xavierovi Glorotovi, v súčasnosti výskumníkovi v Google DeepMind. Je popísaná v článku *Understanding the difficulty of training deep feedforward neural networks*[8]. Existujú dve verzie tejto metódy, xavierová metóda a normalizovaná xavierová metóda. Obe metódy boli odvodené za predpokladu, že aktivačná funkcia je lineárna a napriek tomu sa stali štandardom pre nelineárne aktivačné funkcie ako sigmoid a hyperbolický tangent, ale nie ReLU. Je definovaná ako: $w = U(-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}})$, kde U je rovnomerné rozdelenie a n je počet vstupov daného neurónu.

Zistilo sa, že xavierová inicializácia má problémy, keď sa používa na inicializáciu sietí, ktoré používajú aktivačnú funkciu ReLU. Modifikovaná verzia prístupu bola vyvinutá špeciálne pre neuróny a vrstvy, ktoré využívajú aktiváciu ReLU, populárnu v skrytých vrstvách väčšiny viacvrstvových modelov a konvolučných neurónových sietí.

HE inicializácia je súčasný štandardný prístup k inicializácii váh vrstiev neurónovej siete a neurónov, ktoré využívajú ReLU aktivačnú funkciu. Je pomenovaná po Kaimingovi He, v súčasnosti výskumníkovi vo Facebooku. Je popísaná v článku *Delving Deep into Rectifiers*[10]. Je definovaná ako: $w = N(0.0, \sqrt{\frac{2}{n}})$, kde N je normálne alebo Gaussovo rozdelenie a n je počet vstupov daného neurónu.

3.2.2 Chybová funkcia

Chybová funkcia taktiež známa ako *loss function* alebo *cost function* udáva mieru množstva chýb, ktoré robí neurónová sieť na zadanom datasete. Získana chyba sa pri procese učenia postupne znižuje až dokýn nedosiahne určitý minimum. Hoci existuje veľa chybových funkcií, všetky penalizujú na základe vzdialenosti medzi predpovedanou hodnotou a jeho skutočnou hodnotou v zadanom datasete. Niektoré typy chybových funkcií sú vhodné len na určité typy problémov, napríklad regresné problémy alebo klasifikačné problémy do ktorých spadá rozpoznávanie tváre.

Overfitting alebo pretrénovanie sa týka modelu, ktorý príliš dobre modeluje tréningové dáta. Pretrénovanie nastane, keď sa model naučí detaily a šum v trénovacích dátach do takej miery, že to negatívne ovplyvní výkon modelu na nových dátach. To znamená, že šum alebo náhodné fluktuácie v trénovacích dátach sú zachytené a naučené ako koncepty modelom. Problémom je, že tieto koncepty sa nevzťahujú na nové dáta a negatívne ovplyvňujú schopnosť modelov zovšeobecňovať.

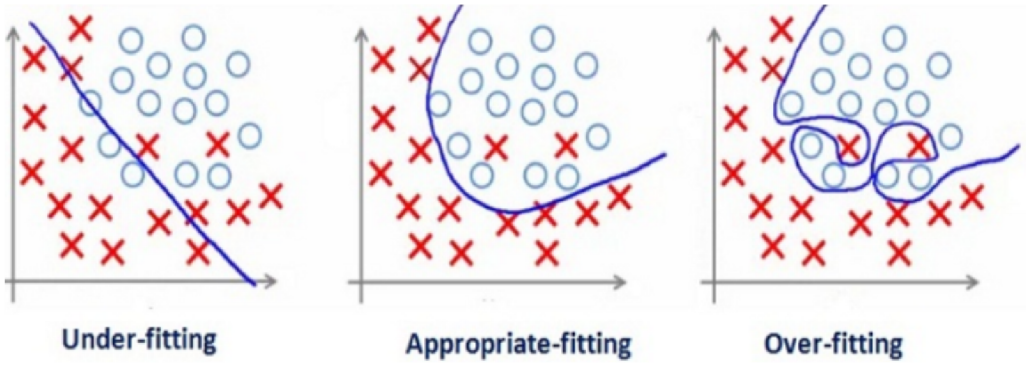
Underfitting alebo podtrénovanie sa týka modelu, ktorý nedokáže modelovať tréningové dáta, ani ich zovšeobecňovať na nové dáta. Podtrénovaný model nie je vhodným modelom a bude to zrejmé, pretože bude mať slabú úspešnosť na tréningových dátach. O podtrénovaní sa často nehovorí, pretože je ľahké ho odhaliť. Tento problém sa dá riešiť vyskúšaním alternatívnych algoritmov hlbokého učenia. Napriek tomu poskytuje dobrý kontrast k pretrénovaniu.

V ideálnom prípade je najlepší model taký, ktorý je medzi podtrénovaním a pretrénovaním. Tento cieľ, je ale v praxi veľmi ťažké dosiahnuť. Na obrázku 3.3 je zobrazený problém podtrénovania a pretrénovania a ideálny model. Existujú dôležité techniky, ktoré pomáhajú pri hodnotení neurónových sietí na obmedzenie pretrénovania:

- Použitie techniky prevzorkovania na odhad presnosti modelu.
- Použitie validačného datasetu.

Najpopulárnejšou technikou prevzorkovania je k-násobná krížová validácia (angl. *k-cross validation*). Umožňuje trénovať a testovať model k-krát na rôznych podskupinách trénovacieho datasetu.

Validačný dataset je jednoducho podmnožinou trénovacieho datasetu, ktorá sa použije až po vyladení neurónovej siete na trénovacom datasete. Natrénovaná neurónová sieť sa vyhodnotí na validačnom datasete a podľa neho sa získa objektívna predstava o tom, ako bude model fungovať na nových dátach.



Obr. 3.3: Problém podtrénovania, ideálny model a problém pretrénovania[2].

3.2.3 Backpropagation

Backpropagation [21] alebo spätné šírenie chyby je algoritmus na výpočet gradientu chybovej funkcie vzhľadom k váham neurónom neurónového modelu. Metóda backpropagation tvorí podstatu učenia hlbokých neuronových sietí. Backpropagation bol prvýkrát aplikovaný na optimalizáciu neurónových sietí gradientovým zostupom v dokumente z roku 1986 od D. Rumelharta, G. Hintona a R. J. Williamsa. Následnú prácu vykonal v 80. a 90. rokoch Yann LeCun, ktorý ju prvýkrát aplikoval na konvolučné siete.

Základnou myšlienkou algoritmu backpropagation je, že umožňuje vypočítať všetky prvky gradientu v jedinom prechode dopredu a dozadu cez sieť, namiesto toho, aby bolo potrebné vykonať jeden prechod dopredu pre každý jednotlivý parameter neurónove siete. Prechodom dopredu je myslený, smer výpočtu od vstupu k výstupu neurónovej siete taktiež známi forward propagation (dopredná propagácia) a prechodom dozadu je mylsený, smer výpočtu od výstupu k vstupu siete. Umožňuje to využitie reťazového pravidla v kalkuluse, ktoré nám umožňuje rozložiť derivát ako súčin jeho jednotlivých funkčných častí. Matematický zápis reťazového pravidla:

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x}. \tag{3.2}$$

Sledovaním rozdielov v priechode dopredu pozdĺž každého spojenia a ich ukladaním je možné vypočítať gradient tak, že sa vezme výsledok chybovej funkcie nájdený na konci prechodu dopredu a „prenesie chybu späť“ cez každú z vrstiev. To spôsobí, že spätná propagácia zaberie približne rovnaké množstvo práce ako dopredná propagácia. To dramaticky urýchľuje trénovanie a robí zostup gradientu na hlbokých neurónových sieťach realizovateľným problémom.

3.2.4 Gradientný zostup

Trénovanie je proces minimalizácie chybovej funkcie, čo je typ optimalizačnej úlohy. Proces optimalizácie je potom definovaný ako hľadanie minima alebo maxima určitej funkcie. U neurónových sietí je touto funkciou chybová funkcia a algoritmus pre jej minimalizáciu sa volá gradientný zostup.

Gradientný zostup [20] je iteratívny optimalizačný algoritmus, ktorý využíva parciálne derivácie k nájdeniu gradientu chybovej funkcie. Keďže cieľom je minimalizovať chybovú funkciu a chybová funkcia sa zvyšuje v smere gradientu, algoritmus gradientného zo-

stupu iteratívne aktualizuje váhy neurónovej siete, aby sa pohybovala proti smeru gradientu. Algoritmus beží v iteráciách, kým chybová funkcia nedosiahne bod, v ktorom prestane klesať, alebo po určitom počte opakovaní. Algoritmus gradientného zostupu je zobrazený na obrázku 3.4.

$$\begin{aligned}&\text{repeat until convergence } \{\\ &\quad \theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)\\ &\quad \text{(for } j = 1 \text{ and } j = 0)\\ &\}\end{aligned}$$

Obr. 3.4: Algoritmus gradientného zostupu, kde θ_j je aktualizovaná váha, α je learning rate a $\frac{\partial}{\partial \theta_j} J$ je parciálna derivácia chybovej funkcie.

Gradient udáva smer minimalizácie funkcie, ale nie dĺžku kroku, o ktorý sa posunú váhy v jednej iterácii v smere gradientu. Dĺžka kroku sa označuje termínom *learning rate*. Jedná sa o hyperparameter, ktorý je treba vhodne nastaviť. Tento hyperparameter je vo všeobecnosti dôležitým a citlivým hyperparametrom na nastavenie a často sa označuje ako α. Ak je *learning rate* nastavený príliš nízko, môže trvať neprijateľne dlho, kým sa dostanete nadol. Ak je príliš vysoko, môže presiahnuť správnu cestu alebo dokonca stúpať nahor. *Learning rate* sa počas trénovania typicky mení.

Gradientný zostup má jeden hlavný problém: je príliš pomalý. Neurónová sieť môže mať stovky miliónov parametrov, to znamená, že jeden príklad z datasetu si vyžaduje stovky miliónov operácií na vyhodnotenie. Následne zostup gradientu vyhodnotený na všetkých príkladoch v datasete – známy aj ako *batch gradientný zostup* (angl. *batch gradient descent*)– je to veľmi nákladná a pomalá operácia. Okrem toho je možné ukázať, že dostatočne veľká podmnožina príkladov môže aproximovať úplný gradient, čím je *batch gradient descent* zbytočne nákladný na odhadnutie gradientu.

Tento problém aj problém lokálnych miním je možné riešiť pomocou modifikovanej verzie gradientného zostupu nazývaného stochastický gradientný zostup (angl. *Stochastic Gradient Descent - SGD*). SGD najprv zamieša dataset a potom prejde každý príklad jednotlivo, gradient sa vypočíta vzhľadom na tento jeden príklad a pre každý sa aktualizujú váhy. Na prvý pohľad sa to môže zdať ako zlý nápad, pretože jeden príklad môže byť odľahlý od ostatných a nemusí nevyhnutne poskytovať dobrú aproximáciu skutočného gradientu. Ukazuje sa však, že ak sa to vykoná pre každý príklad datasetu v nejakom náhodnom poradí, celkové kolísanie cesty aktualizácie gradientu sa spriemeruje a bude konvergovať k dobrému riešeniu. Okrem toho, SGD pomáha, sa dostať z miestnych miním a sedlových bodov tým, že aktualizácie váh sú „trhľavejšie“ a nepravidelnejšie. SGD je obzvlášť užitočné v prípadoch, keď je povrch chybovej funkcie nepravidelný. Ale vo všeobecnosti je zvyčajným prístupom použitie mini-batch gradientového zostupu (angl. *Mini Batch Gradient Descent - MB-GD*), v ktorom je celý dataset náhodne rozdelený do N rovnako veľkých mini-batchov, z ktorých každý pozostáva z K príkladov. K môže byť malé kladné číslo alebo môže byť v desiatkach alebo stovkách, závisí od konkrétnej architektúry a aplikácie neurónovej siete.

3.3 Konvolučné neurónové siete

Konvolučná neurónová sieť (angl. *Convolutional neural networks - CNN*) [9][18] je dobre známa architektúra hlbokého učenia inšpirovaná prirodzeným mechanizmom vizuálneho vnímania živých tvorov. Významný bod v práci na konvolučných sieťach sa stal v roku 2012 keď Alex Krizhevsky, Ilya Sutskever a Geoffrey E. Hinton publikovali a následne vyhrali súťaž ImageNet (konkétne ILSVRC-2012) vytvorením jedinečnej hlbokej konvolučnej neurónovej siete, ktorá dokázala klasifikovať 1,3 milióna obrázkov s vysokým rozlíšením do tisíc rôznych tried a mala 15,3 % chybovosť na testovacích dátach, ktorá bola oveľa lepšia ako model na druhom mieste (26,2 %).

Inováciou konvolučných neurónových sietí je schopnosť sa paralelne učiť veľké množstvo filtrov špecifických pre trénovací dataset pod obmedzeniami špecifického problému, ako je napríklad klasifikácia obrázkov. Výsledkom sú vysoko špecifické vlastnosti, ktoré je možno zistiť kdekoľvek na vstupných obrázkoch.

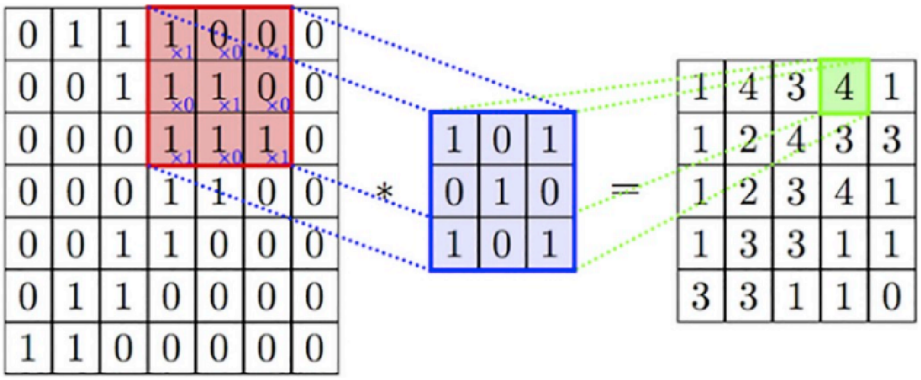
Významný rozdiel medzi Konvolučnými neurónovými sieťami a tradičnými neurónovými sieťami je ten, že CNN sa primárne používajú v oblasti počítačového videnia hlavne na rozpoznávania objektov v rámci obrázkov. To umožňuje zakódovať vlastnosti špecifické pre obrázok do architektúry, vďaka čomu je sieť vhodnejšia pre úlohy zamerané na obrazové dáta – a zároveň ďalej znižuje parametre potrebné na nastavenie modelu. Konvolučné neurónové siete sa učia pomocou backpropagation algoritmu. Vďaka konvolúciám, ktoré sú vykonávané v konvolučných vrstvách, je sieť invariantná voči posunom alebo iným deformáciám vstupného obrazu. CNN sa skladajú zo vstupnej vrstvy, skrytých vrstiev a výstupnej vrstvy, rozdiel oproti tradičným sieťam je v tom, že používajú iné skryté vrstvy. Skryté vrstvy obsahujú konvolučné vrstvy, ktoré tvoria jadro konvolučných neurónových sietí, združujúcich alebo pooling vrstiev a plne prepojených vrstiev. Jednotlivé vrstvy budú popísane v nasledujúcej sekcii. Na konci CNN sa vezme výstupný vektor obsahujúci informácie o vstupnom obraze a predá sa klasifikátoru.

Konvolučná vrstva

Táto vrstva je založená na operácii konvolúcie. Konvolúcia je lineárna operácia, ktorá zahŕňa násobenie množiny váh so vstupom. Násobenie sa vykonáva medzi vstupným poľom dát a dvojrozmerným poľom váh, nazývaným filter alebo jadro (angl. *kernel*). Filter je menší ako vstupné dáta a typ násobenia aplikovaný medzi filtrom a časťou obrázka s rovnakou veľkosťou ako je filter je skalárny súčin. Skalárny súčin je násobenie hodnotou medzi časťou obrázka a filtrom, ktoré sa potom sčítavajú a výsledkom je vždy jedna hodnota. Použitie filtra menšieho ako je vstup je zámerné, pretože umožňuje, aby sa rovnaký filter vynásobil vstupným poľom dát viackrát v rôznych bodoch obrázka. Konkrétne sa filter aplikuje systematicky na každú prekrývajúcu sa časť, zľava doprava, zhora nadol. Výstopom tejto operácie je dvojrozmerné výstupné pole, ktoré sa nazýva mapa vlastností (angl. *feature map*). Operácia konvolúcie je matematicky definovaná ako:

$$y(i,j) = \sum_{m=-k}^{k} \sum_{n=-k}^{k} h(m,n) * x(i-m, j-n), \tag{3.3}$$

kde y je výstupná mapa vlastností, h je filter, x reprezentuje vstupné dáta a $*$ je operácia konvolúcie. Na obrázku 3.5 je zobrazený proces konvolúcie.



Obr. 3.5: Výpočet konvolúcie[26].

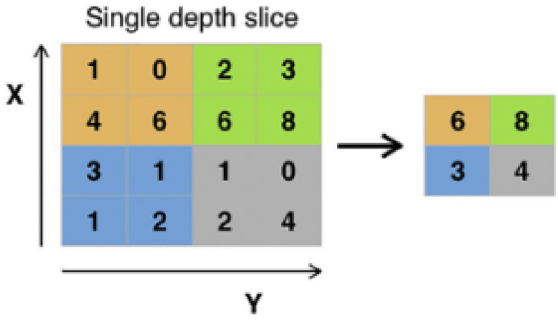
Rozmery použitého filtru sú rôzne. Väčšinou sa v praxi používajú veľkosti s nepárnymi hodnoty. Okrem veľkosti filtru je možné pri konvolucií nastaviť aj *padding* a *stride*. *Padding* popisuje pridanie prázdnych pixelov okolo okraja obrázka. Účelom výplne je zachovať pôvodnú veľkosť obrázka pri použití konvolučného filtra a umožniť filtru vykonávať úplné konvolúcie na okrajových pixeloch. *Stride* v kontexte konvolučných neurónových sietí popisuje proces zväčšenia veľkosti kroku, ktorým sa filter bude posúvať cez vstupný obrázok. Ak bude *stride* nastavený na 2 filter sa v každom kroku posunie o dva pixely.

Pooling vrstva

Kedže konvolučné neurónové siete typicky používajú veľké počet konvolučných filtrov, čo spôsobuje lepšiu presnosť siete ale aj nárast v počte dát, ktoré musí sieť spracovať. Preto sa používajú *pooling* alebo združovacie vrstvy, ktorých úlohou je zmenšiť, podvzorkovať dáta. *Pooling* vrstva sa pridáva po konvolučnej vrstve. Konkrétne po aplikovaní aktivačnej funkcie (napr. ReLU) na výstup konvolučnej vrstvy. *Pooling* vrstva pracuje s každou mapou vlastností samostatne, aby vytvorila novú množinu rovnakého počtu máp vlastností. *Pooling* zahŕňa výber *pooling* filtra, ktorý sa má použiť na mapy vlastností. Veľkosť *pooling* filtra je menšia ako veľkosť mapy vlastností. Konkrétne ide takmer vždy o filter veľkosti 2×2 s krokom 2. To znamená, že *pooling* vrstva vždy zmenší veľkosť každej mapy vlastností o faktor 2. *Pooling* operácia je špecifikovaná, nie natrénovaná. Dve bežné operácie sú:

- *Average Pooling*: Vypočíta priemernú hodnotu pre každú oblasť na mape vlastností.
- *Maximum Pooling*: Vypočíta maximálnu hodnotu pre každú oblasť na mape vlastností.

Na obrázku 3.6 je zobrazená operácia *maximum pooling*.



Obr. 3.6: Operácia maximum pooling[26].

Plne prepojená vrstva

Označovaná aj ako FC (*Fully-Connected*) vrstva. Vstupom do tejto vrstvy je výstup z konečnej pooling alebo konvolučnej vrstvy. Vstup je prevedený na vektor pomocou sploštenia. Každý neurón predchádzajúcej vrstvy je napojený na každý vstup plne prepojenej vrstvy. Vzhľadom na vysoký počet spojení sa zvyšujú aj výpočtové nároky. Preto sa táto vrstva používa až v posledných vrstvách neurónovej siete, kde má úlohu klasifikácie objektov.

3.4 Detektory tváre

Detekcia a zarovnanie tváre sú nevyhnutné pre mnohé aplikácie tváre, ako je rozpoznávanie tváre a analýza výrazu tváre. Avšak veľké vizuálne variácie tvárí, ako sú oklúzie, veľké variácie póz a extrémne osvetlenie, predstavujú pre tieto úlohy veľké výzvy v aplikáciách v reálnom svete. Kaskádový detektor tváre navrhnutý Viola a Jones[28] využíva *Haar-Like* vlastnosti a *AdaBoost* na trénovanie kaskádových klasifikátorov, ktoré dosahujú dobrý výkon s efektívnosťou v reálnom čase. *Haar-Like* vlastnosti sú škálovateľné, obdĺžnikové rámce, ktoré sa používajú na porovnanie vzájomného vzťahu pixelov, konkrétne, aký tmavý je jeden ku druhému. *AdaBoost* je veľmi populárna zosilňovacia technika, ktorej cieľom je skombinovať viacero slabých klasifikátorov na vytvorenie jedného silného klasifikátora. Pomerne veľa vedeckých prác však naznačuje, že tento detektor môže výrazne degradovať v reálnych aplikáciách s väčšími vizuálnymi variáciami ľudských tvárí, dokonca aj s pokročilejšími vlastnosťami a klasifikátormi. V poslednej dobe konvolučné neurónové siete dosahujú pozoruhodný výsledky v rôznych úlohách počítačového videnia, ako je klasifikácia obrazu a rozpoznávanie tváre. Pred publikáciou práce MTCNN boli navrhnuté niektoré prístupy na detekciu tváre založené na neurónových sieťach avšak niektoré mali zložitú štruktúru konvolučnej siete a boli nevhodné kvôli časovej nákladnosti trénovania, iné dokázali len detekovať tváre v obraze bez lokalizácie orientačných bodov tváre.

3.4.1 MTCNN

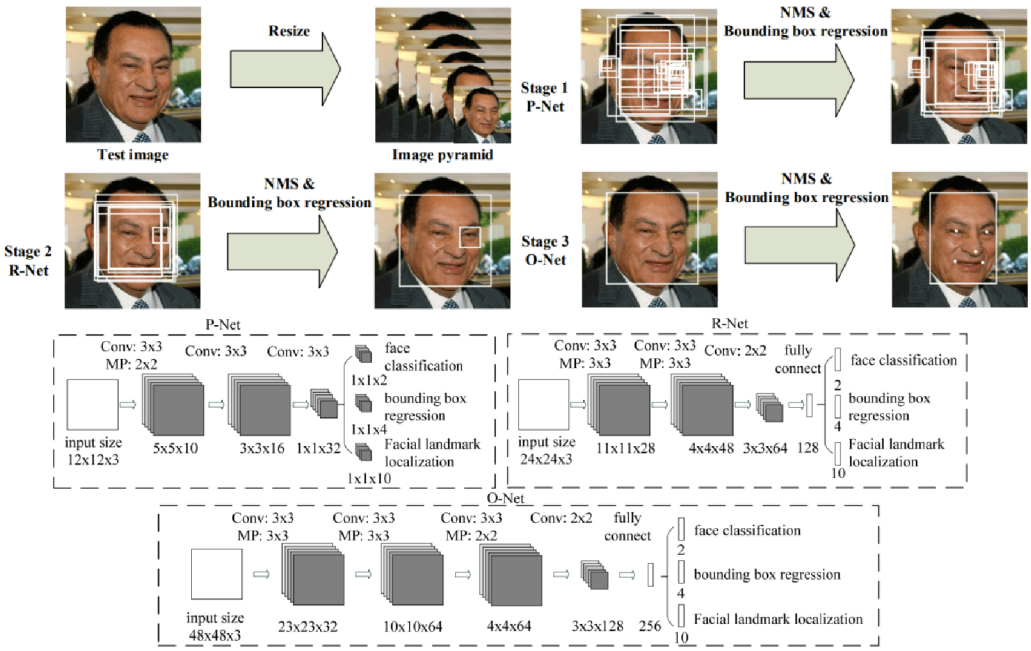
Algoritmus MTCNN[35] alebo *Multi-task Cascaded Convolutional Network* je kaskádová konvolučná neurónová sieť určená pre detekciu tváre v obraze a detekciu orientačných bodov tváre. Architektúra modelu sa skladá z troch konvolučných neurónových sietí. V prvej fáze používa plytkú konvolučnú neurónovú sieť na rýchle vytváranie kandidátskych okien (časť obrázka, kde sa pravdepodobne nachádza tvár). V druhej fáze spresňuje navrhované kandidátske okná prostredníctvom komplexnejšej konvolučnej neurónovej siete. A napo-

kon, v tretej fáze používa tretiu konvolučnú neurónovú sieť, zložitejšiu ako ostatné, na ďalšie spresnenie výsledku a výstupných pozícií orientačných bodov tváre. Ešte pred použitým prvej neurónovej siete sa zoberú vstupné obrázky a zmenia sa ich veľkosti na rôzne mierky, aby sa vytvorila obrazová pyramída, ktorá je vstupom nasledujúcej trojstupňovej kaskádovej siete.

The Proposal Network (P-Net) je prvá sieť MTCNN algoritmu. Táto prvá sieť je plne konvolučná sieť (*fully convolutional network*). Rozdiel medzi normálnou konvolučnou sieťou a plnou je v tom, že plne konvolučná sieť nepoužíva hustú vrstvu ako súčasť architektúry. Táto sieť sa používa na získanie kandidátskych okien a ich regresných vektorov ohraničenia. Bounding box regresia je populárna technika na predpovedanie lokalizácie boxov, kde je cieľom detekovať objekt nejakej vopred definovanej triedy, v tomto prípade tváre. Po získaní vektorov ohraničenia sa vykoná určité spresnenie, aby sa spojili prekrývajúce sa oblasti. Konečným výstupom tejto fázy sú všetky okná kandidátov po spresnení, aby sa zmenšil objem kandidátov.

The Refine Network (R-Net) je druhá sieť MTCNN algoritmu. Všetci kandidáti z P-Net sú zaradení do siete R-Net. Narozdiel od P-Net, táto sieť nieje plne konvolučná sieť, pretože v poslednej fáze architektúry siete je hustá vrstva. R-Net ďalej znižuje počet kandidátov, vykonáva kalibráciu s regresiou bounding boxu a využíva nemaximálne potlačenie (*non-maximum suppression*) na zlúčenie prekrývajúcich sa kandidátov. Výstupom R-Net je odpoveď, či vstup je tvár alebo nie. Ak áno, výsledkom je 4-prvkový vektor, ktorý je bounding boxom pre tvár, a 10-prvkový vektor na lokalizáciu orientačných bodov tváre.

The Output Network (O-Net) je tretia sieť MTCNN algoritmu. Táto sieť je podobná R-Net, ale cieľom tejto výstupnej siete je podrobnejšie opísať tvár a poskytnúť päť pozícií orientačných bodov pre tvár, pre oči, nos a ústa. Na obrázku 3.8 je zobrazený princip algoritmu a detaily jednotlivých konvolučných sietí.



Obr. 3.7: Vizualizácia algoritmu MTCNN a detaily jednotlivých architektúr konvolučných neurónových sietí[35].

3.5 Datasety

Výber správneho datasetu je jedným z hlavných kľúčov k úspešnému výslednemu modelu na rozpoznávanie tvárí. Existujú datasety obrázkov tváre rôznej kvality a veľkosti obrázkov. Datasety, ktoré sa používajú na rozpoznávanie tvárí je možné rozdeliť na dve hlavné skupiny, voľne dostupné (*in-the-wild*) datasety a špeciálne zozbierané datasety. Špeciálne zozbierané datasety sa bežnejšie používali v skorších dňoch rozpoznávania tváre pred používaním neurónových sietí, tieto datasety boli často profesionálne nasnímané v štúdiách, kde bolo možné ovládať osvetlenie a pózu osoby. Nevýhodou tejto skupiny datasetov je však to, že zvyčajne boli dosť malé v počte subjektov a neponúkali dostatočnú rozmanitosť ako ich vek, etnickú príslušnosť alebo pohlavie. Najbežnejšie datasety sú v šúčasnej dobe voľne dostupné datasety. Tieto datasety obsahujú obrázky ľudí v nedokonalých podmienkach. Tvár môže byť otočená akýmkoľvek smerom, osoba môže mať slnečné okuliare, šiltovku alebo iný objekt, ktorý robí rozpoznanie ťažším. Moderné voľne dostupné datasety vo všeobecnosti obsahujú fotografie ľudí rôzneho veku, etnickej príslušnosti a pohlavia. Tieto datasety sú tiež vo všeobecnosti dosť veľké, v desiatkach, stovkách tisíc fotografií. Existujú aj datasety s obrovským počtom fotografií avšak tieto datasety vlastnia komerčné firmy a často niesú verejné. Trénovanie modelu na voľne dostupných datasetoch je náročnejšie, pretože väčšina fotografií je nedokonalá, avšak takýto natrénovaný model bude robustnejší a taktiež lepší.

Rozmanitosť datasetu je veľmi dôležitá. Dataset by mal obsahovať dostatočne rôznorodý obsah ľudí rôznej etnicity, veku, pohlavia a polohy hlavy. Ak by dataset napríklad obsahoval iba fotografie mužov, potom by v prípade rozpoznávia tvárí žien, natrénovaný model nedokázal vytvoriť presné vektory vlastností pre ženské obrázky. Takýto extrémny prípad je príkladom takzvaného algoritmického bias-u, sklonu a môže sa vyskytnúť aj v menej extrémnych prípadoch, keď je jedna z týchto skupín jednoducho nedostatočne zastúpená. Ďalšou dôležitou vlastnosťou datasetu je všeobecná kvalita fotografií, ak sú všetky fotografie dobre osvetlené a objekty sú otočené priamo pred kameru, natrénovaný model nemusí byť schopný správne rekonštruovať fotografie, ktoré nie sú nasnímané za dokonalých podmienok. Kľúčom k výberu dobrého datasetu je vybrať taký, ktorý obsahuje niektoré nedokonalé obrázky, v datasetoch je veľmi dôležitá rozmanitosť a rovnováha. Trénovanie modelu na vyváženom datasete spôsobí, že výsledný model bude dosť robustný a dá sa dobre zovšeobecniť na rôzne typy obrázkov. V ďalšej časti je popísaných pár datasetov, ktoré sa často používajú na trénovanie a verifikáciu úspešnosti modelu.

3.5.1 Labeled Faces in the Wild

Labeled Faces in the Wild (LFW)[11] je verejne dostupný dataset zameraný na verifikáciu osôb pomocou obrázkov tváre. Dataset obsahuje 13233 obrázkov. Každý obrázok obsahuje jednu osobu so známou identitou. Celkovo sa v datasete nachádza 5749 osôb, z toho 1680 osôb sa vyskytuje na viac než jednom obrázku. Tento dataset autori poskytujú v dvoch verziách s neupravenými tvárami alebo vo verzii so zarovnanými tvárami. Oproti starším datasetom, LFW sa nesnaží o normalizíáciu a unifikáciu obrázkov. Tie tak obsahujú obrázky osôb s prirodzenou variáciou natočenia, pózy, osvetlenia, zaostrenia a výrazu tváre. Všetky obrázky sú k dispozícii v rozlišení 250x250 pixelov, väčšina obrázkov je vo farebnom prevedení, malá časť je černobiela. Keďže vlastné rozdelenie na trénovaciu, validačnú a testovaciu časť môže mať vplyv na kvalitu testovaného algoritmu, autori datasetu poskytujú vopred definované rozdelenie. Tento dataset sa svojou veľkosťou radí skôr medzi menšie, nie je často využívaný na trénovanie modelov. Slúži však ako jeden z najviac využívaných

datasetov na porovnanie presnosti rôznych modelov. Na webových stránkach datasetu je tiež prístupný rebríček najúspešnejších modelov. Tento dataset bol vytvorený vedcami na univerzite v Massachusetts v roku 2018.

3.5.2 CelebFaces Attributes Dataset

CelebFaces Attributes Dataset (CelebA)[16] je verejne dostupný dataset, ktorý sa radí do skupiny *in the wild* datasetov. Obsahuje 202,599 obrázkov tvárí celebrít. V datasete sa nachádza 10 177 rozdielnych identít. Tento dataset stavia na staršom datasete s názvom CelebFaces pridaním anotácií popisujúcich jednotlivé charakteristiky každej z obrázkov. Anotácie popisujú rôzne črty obrázku, ako je farba vlasov, či osoba nosí alebo nemá okuliare a iné. K dispozícii sú aj orientačné body na tvári. Rozlíšenie obrázkov je 178×218 pixelov vo farebnom prevedení. Dataset je pomerne rôznorodý, pokiaľ ide o etnickú príslušnosť aj pohlavie, ale nieje dobre rôznorodý ak sa jedná o vek. Kvalita obrázkov je rôzna a vyskytujú sa v ňom aj obrázky nízkej kvality, pretože upravený dataset bol vytvorený extrahovaním tvárí z väčších fotografií rôznej kvality, ich orezaním a následnou zmenou veľkosti. Niektoré z obrázkov boli orezané a orezané okraje sú potom vyplnené rovnakými pixelmi ako okraj orezania. Autori poskytujú presné rozdelenie obrázkov na trénovaciu, validačnú a testovaciu časť. Tento dataset bol vytvorený vedcami na čínskej univerzite v Hong Kongu v roku 2015.

3.5.3 VGGFace2

Dataset VGGFace2[5] nadväzuje na dataset VGGFace a je jedním z aktuálne najväčších verejne dostupných datasetov, ktoré neobsahujú príliš veľa šumu (chybných detekcií, zlých anotacií, apod.). Celkom obsahuje 3,31 miliónov obrázkov, na ktorých je zachytených 9131 osôb. Priemerný počet obrázkov na osobu je 362. Tento dataset obsahuje bounding boxy okolo tvárí overené ľuďmi a päť orientačných bodov na tvári, podobne ako dataset CelebA. Okrem toho sa nachádzajú v datasete informácie o polohe tváre (vybočenie, sklon a prevrátenie) a veku, ktorý je odhadovaný vopred natrénovaným klasifikátormi pózy a veku. Obrázky celebrít boli stiahnuté z vyhľadávania obrázkov Google. Dataset je čiastočne manuálne vyčistený a autori udávajú, že dataset obsahuje maximálne 4% šumu. Oproti svojmu predchádzajúcemu datasetu je väčší, obsahuje menej šumu a modely natrénované na tomto datasete dosahujú vyššej presnosti na testovacej sade. Dataset bol vytvorený vedcami na univerzite Oxford v roku 2018.

3.5.4 YouTube Faces Database

Youtube Faces Database (YDB)[30] je dataset zameriavajúci sa na identifikáciu osôb vo videu. Skladá sa z celkom 3425 video súborov, na ktorých sa vyskytuje 1595 rôznych osôb. Všetky videá boli stiahnuté z Youtube, zhotovené záznamy teda obsahujú vysoko rozdielne ukážky natočenia tváre, rozlíšenia a svetelných podmienok. Extrakciou snímok z videa tiež môže dôjsť k ďalšiemu zníženiu kvality kvôli kompresii a rozmazaniu snímky pohybom. Osoby vo videách boli identifikované pomocou kombinácie automatických a manuálnych techník, priemerne sa každá osoba vyskytuje v 2,15 videách. Najkratšia dĺžka videa je 48 obrázkov, najdlhšia 6 070 obrázkov a priemerná dĺžka videa je 181,3 obrázka. Autori datasetu sa pri vytváraní riadili príkladnym datasetom LFW. Tento dataset je velmi populárny na evaluáciu kvality modelu rozpoznávania tváre. Dataset bol vytvorený vedcami na univerzite Tel Aviv v roku 2011. Na obrázku 3.13 je zobrazená tabulka datasetov a ich vlastností, ktoré sa používajú na trénovanie modelu rozpoznávania tvárí.

THE COMMONLY USED FR DATASETS FOR TRAINING

Datasets	Publish Time	#photos	#subjects	# of photos per subject [1]	Key Features
MS-Celeb-1M (Challenge 1)[45]	2016	10M 3.8M(clean)	100,000 85K(clean)	100	breadth; central part of long tail; celebrity; knowledge base
MS-Celeb-1M (Challenge 2)[45]	2016	1.5M(base set) 1K(novel set)	20K(base set) 1K(novel set)	1/-/100	low-shot learning; tailed data; celebrity
MS-Celeb-1M (Challenge 3) [163]	2018	4M(MSv1c) 2.8M(Asian-Celeb)	80K(MSv1c) 100K(Asian-Celeb)	-	breadth;central part of long tail; celebrity
MegaFace [44], [164]	2016	4.7M	672,057	3/7/2469	breadth; the whole long tail;commonalty
VGGFace2 [39]	2017	3.31M	9,131	87/362.6/843	depth; head part of long tail; cross pose, age and ethnicity; celebrity
CASIA WebFace [120]	2014	494,414	10,575	2/46.8/804	celebrity
MillionCelebs [165]	2020	18.8M	636K	29.5	celebrity
IMDB-Face [124]	2018	1.7M	59K	28.8	celebrity
UMDFaces-Videos [166]	2017	22,075	3,107	–	video
VGGFace [37]	2015	2.6M	2,622	1,000	depth; celebrity; annotation with bounding boxesand coarse pose
CelebFaces+ [21]	2014	202,599	10,177	19.9	private
Google [38]	2015	>500M	>10M	50	private
Facebook [20]	2014	4.4M	4K	800/1100/1200	private

[1] The min/average/max numbers of photos or frames per subject

Train/ Test	Database	Race (%)				Gender (%)	
		Caucasian	Asian	Indian	African	Female	Male
train	CASIA-WebFace [120]	84.5	2.6	1.6	11.3	41.1	58.9
	VGGFace2 [39]	74.2	6.0	4.0	15.8	40.7	59.3
	MS-Celeb-1M [45]	76.3	6.6	2.6	14.5	-	-
test	LFW [23]	69.9	13.2	2.9	14.0	25.8	74.2
	IJB-A [41]	66.0	9.8	7.2	17.0	-	-

Obr. 3.8: Tabulka obsahujúca vlastnosti a štatistiky datasetov používaných na trénovanie modelu rozpoznávania tvárí[29].

3.6 Neurónové siete pro rozpoznávanie ôsob podľa tváre

3.6.1 FaceNet

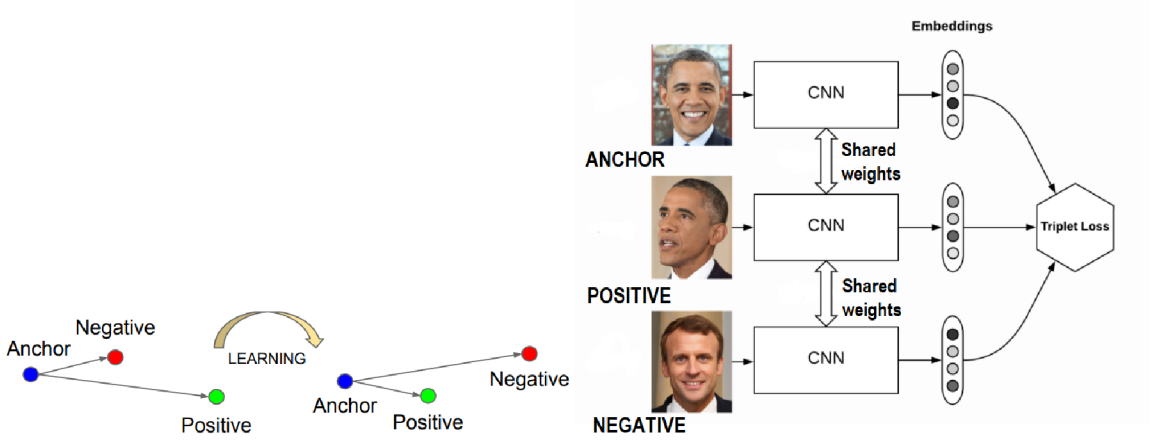
FaceNet[23] je systém rozpoznávania tváre, ktorý popísal Florian Schroff a jeho kolegovia zo spoločnosti Google v ich dokumente z roku 2015. Je to systém, ktorý na základe obrázku tváre vytiahne z tváre vysokokvalitné vlastnosti a predpovedá vektorovú reprezentáciu týchto vlastností so 128 prvkami, taktiež nazývanú *face embedding*.

FaceNet poskytuje jednotnú vektorová reprezentáciu na rozpoznávanie tváre, verifikáciu a klasifikačné úlohy. Mapuje každý obraz tváre do euklidovského priestoru tak, že vzdialenosti v tomto priestore zodpovedajú podobnosti tváre, to znamená obraz osoby A bude umiestnený bližšie ku všetkým ostatným obrazom osoby A v porovnaní s obrazmi akejkoľvek inej osoby prítomnej v datasete. Hlavným rozdielom medzi týmto prístupom a inými technikami je v tom, že FaceNet sa učí mapovanie z obrázkov a vytvára vektor vlastností, namiesto toho, aby na rozpoznávanie alebo na verifikáciu používal akúkoľvek alšiu vrstvu. Akonáhle sú vytvorené vektory vlastností, všetky ostatné úlohy, ako je verifikácia, rozpoznávanie atď., môžu byť vykonané pomocou štandardných techník. Napríklad môžeme použiť algoritmus k-najbližších-susedov na rozpoznávanie tvárí pomocou vektora vlastností a po-

dobne môžeme použiť akúkoľvek techniku zhlukovania na zoskupenie tvárí a na verifikáciu stačí definovať prahovú hodnotu.

FaceNet používa hlbokú konvolučnú neurónovú sieť. Sieť je trénovaná tak, že euklidovská vzdialenosť medzi vektormi vlastností zodpovedá podobnosti tváre. Obrázky používané na trénovanie sú zmenšené, transformované a sú tesne orezané okolo oblasti tváre. Ďalším dôležitým aspektom je chybová funkcia. Používa funkciu *triplet loss*. Na vypočítanie chyby sú potrebné tri obrázky, a to vstupný, pozitívny a negatívny.

Intuícia za funkciou *triplet loss* spočíva v tom, že je potrebné, aby bol vstupný obrázok (obraz konkrétnej osoby A) bližšie k pozitívnym obrázkom (všetky obrazy osoby A) v porovnaní s negatívnymi obrázkami (všetky ostatné obrazy). Inými slovami, je potrebné, aby vzdialenosti medzi vektorom vlastností vstupného obrázku a vektormi vlastností pozitívnych obrázkov boli menšie v porovnaní so vzdialenosťami medzi vektorom vlastností vstupného obrázku a vektormi vlastností negatívnych obrázkov. Na obrázku 3.10 je zobrazený princip *triplet loss* funkcie.



Obr. 3.9: *Triplet loss* minimalizuje vzdialenosť medzi vstupným obrázkom a pozitívnym obrázkom, pričom oba obrázky majú rovnakú identitu a maximalizuje vzdialenosť medzi vstupným obrázkom a záporným obrázkom s odlišnou identitou.

Funkcia *triplet loss* je matematicky definovaná ako:

$$L = \sum_{i}^{N} \left[||f(x_i^a) - f(x_i^p)||_2^2 - ||f(x_i^a) - f(x_i^n)||_2^2 + \alpha \right]_+, \tag{3.4}$$

kde x_i reprezentuje daný obrázok, $f(x_i)$ reprezentuje vektor vlastností daného obrázku, parametr α značí hranicu (angl. *margin*), medzi pozitivnými a negatívnymi pármi. Horné indexy (a, p, n) značia vstupné, pozitívne a negatívne obrázky v uvedenom poradí. Alpha (α) je v podstate prahová hodnota, ktorá určuje rozdiel medzi pármi obrázkov.

Výber správnych párov obrázkov je mimoriadne dôležitý, pretože bude veľa párov obrázkov, ktoré budú spĺňať hranicu, a preto sa z nich model veľa nenaučí a bude sa tiež pomaly približovať ku optimálnemu výsledku. Preto je vhodné vybrať pozitívne obrázky (x_i^p) pre daný vstupný obrázok (x_i^a) ako $argmax||f(x_i^a) - f(x_i^p)||_2^2$, čo znamená, že pre daný vstupný obrázok osoby A, je potrebné nájsť pozitívny obrázok osoby A taký, aby vzdialenosť medzi týmito dvoma obrázkami bola najväčšia. Negatívne obrázky (x_i^n) pre daný vstupný obrázok je vhodné vybrať ako $argmin||f(x_i^a) - f(x_i^n)||_2^2$, čo znamená, že pre daný vstupný obrázok osoby A, je potrebné nájsť negatívny obrázok osoby A taký, aby vzdialenosť medzi týmito

dvoma obrázkami bola najmenšia. Touto metódou získavame ťažké pozitívne a negatívne obrázky (angl. *hard positives*, *hard negatives*). Tento prístup pomáha urýchliť konvergenciu modelu, pretože model sa učí užitočné reprezentácie. Ale s týmto prístupom je spojený problém, je výpočtovo nemožné vypočítať tvrdé pozitíva a tvrdé negatíva pre celý dataset. Šikovným riešením, na ktoré prišli autori FaceNetu, je vypočítať tvrdé pozitíva a negatíva pre mini-batch. Vo väčšine experimentov bola veľkosť mini-batchu okolo 1800 obrázkov.

FaceNet používa 2 typy konvolučných neurónových sietí, a to architektúru Zeiler & Fergus[34] a Inception model v štýle GoogLeNet[25]. Architektúra Zeiler & Fergus sa používa na vizualizáciu tréningového procesu konvolučných neurónových sietí. Hlavnou myšlienkou architektúry siete Inception je použitie viacerých filtrov rôznych veľkostí súčasne. V akejkoľvek inej tradičnej sieťovej architektúre sa zvyčajne volí filter veľkosti 3×3, 5×5 atď., ale v architektúre Inception sa používa viacero filtrov súčasne a spájajú sa ich výsledky. Táto architektúra modelu Inception použitá v článku FaceNet má medzi 6,6 až 7,5 milióna parametrov a približne 0,5 až 1,6 miliárd operácií s pohyblivou rádovou čiarkou za sekundu (FLOPS). Vo FaceNete sa používajú rôzne variácie modelu Inception, niektoré z nich sú optimalizované na mobilné telefóny, a preto majú porovnateľne menej parametrov a filtrov. Najvyššia pesnosť bola nameraná u siete Inception 224 × 224 (89,4 %). Na obrázku 3.10 je zobrazený detailne model FaceNetu.

Pri hodnotení modelu sa autorom podarilo dosiahnuť presnosť 99,63 % na datasete *Labeled Faces in the Wild*[11] a na datasete *YouTube Faces DB*[30] dosiahol presnosti 95,12 %.

type	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj (p)	params	FLOPS
conv1 (7×7×3, 2)	112×112×64	1							9K	119M
max pool + norm	56×56×64	0						m 3×3, 2		
inception (2)	56×56×192	2		64	192				115K	360M
norm + max pool	28×28×192	0						m 3×3, 2		
inception (3a)	28×28×256	2	64	96	128	16	32	m, 32p	164K	128M
inception (3b)	28×28×320	2	64	96	128	32	64	L_2, 64p	228K	179M
inception (3c)	14×14×640	2	0	128	256,2	32	64,2	m 3×3,2	398K	108M
inception (4a)	14×14×640	2	256	96	192	32	64	L_2, 128p	545K	107M
inception (4b)	14×14×640	2	224	112	224	32	64	L_2, 128p	595K	117M
inception (4c)	14×14×640	2	192	128	256	32	64	L_2, 128p	654K	128M
inception (4d)	14×14×640	2	160	144	288	32	64	L_2, 128p	722K	142M
inception (4e)	7×7×1024	2	0	160	256,2	64	128,2	m 3×3,2	717K	56M
inception (5a)	7×7×1024	2	384	192	384	48	128	L_2, 128p	1.6M	78M
inception (5b)	7×7×1024	2	384	192	384	48	128	m, 128p	1.6M	78M
avg pool	1×1×1024	0								
fully conn	1×1×128	1							131K	0.1M
L2 normalization	1×1×128	0								
total									7.5M	1.6B

Obr. 3.10: Architektúra siete FaceNet[23].

3.6.2 ArcFace

Hlavnou výzvou pri konštrukcii modelu na rozpoznávanie obrázkov vo veľkom rozsahu je implementácia jej chybovej funkcie, ktorá môže účinne zvýšiť rozlišovaciu schopnosť medzi zhlukmi obrázkov. Autori článku ArcFace[6] identifikovali dôležitosť chybovej funkcie pri modeloch rozpoznávania tváre a navrhli novú chybovú funkciu, ktorá je inšpirovaná často používanou klasifikačnou chybovou funkciou *soft-max loss*. Funkcia *soft-max loss* má nasledujúcu definíciu:

$$L_1 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{W_{y_i}^T x_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T x_i + b_{y_j}}}, \tag{3.5}$$

kde $x_i \in \mathbb{R}^d$ značí vektorovú reprezentáciu i-tého obrázku patriaceho do y-tej skupiny, velkosť vektoru vlastností d je v ArcFace článku daná na 512 hodnôt. $W_j \in \mathbb{R}^d$ označuje j-tý stĺpec váh $W \in \mathbb{R}^{d \times n}$ a $b_j \in \mathbb{R}^n$ je hodnota *bias*. Velkosť trénovacej dávky (angl. *batch*) je označená N a n označuje počet tried v datasete.

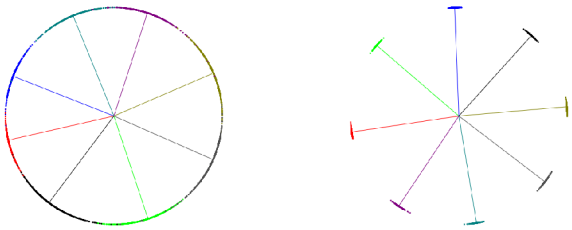
Funkcia *soft-max loss* však dostatočne neoptimalizuje výstupné vektory vlastností, aby sa presadila vyššia podobnosť pre obrázky v rámci jednej triedy a vyššia rozmanitosť pre obrázky medzi jednotlivými triedami. To vedie k výkonnostnej medzere pri rozpoznávaní tvárí, kde sa nachádza veľa variácií vzhľadu v rámci jednej triedy (napr. variácie póz a vekový rozdiel) a taktiež pri rozsiahlych testovacích scenároch.

Problém funkcie *soft-max loss* začali riešiť v podobných prácach pomocou uhlovej (kosínusovej) vzdialenosti upravovaním rovnice (3.5). V článku ArcFace upravili rovnicu určením hodnoty prahu (angl. *bias*) $b_j = 0$, zmenou $W_j^T x_i$ za $||W_j||||x_i|| \cos\theta_j$, kde θ_j je uhol medzi váhou W_j a vektorom vlastností x_i. Individuálna váha bola zmenená na $||W_j|| = 1$ podľa L_2 normalizácie. Pomocou L_2 normalizácie bol zmenený aj vektor vlastností (angl. *embedding feature*) a preškálovaný ho na hodnotu s. Tento krok normalizácie vektorov a váh spôsobuje, že predpovede závisia iba od uhla medzi vektorom a váhou. Naučené vektory vlastností (angl. *embedding features*) sú teda rozmiestnené na hypersfére s polomerom s. K týmto úpravám autori článku pridali penalizáciu m *additive angular margin penalty* medzi vektorom vlastností x_i a váhou W_{y_i} aby súčasne zvýšili kompaktnosť v rámci triedy a nesúlad medzi triedami. Výsledná chybová funkcia má nasledujúcu definíciu:

$$L_2 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))} + \sum_{j=1, j\neq y_i}^{n} e^{s\cos\theta_j}}, \tag{3.6}$$

kde m je penalizácia *additive angular margin penalty* a ostatné členy sú popísané v odstavci nad definíciou (3.6).

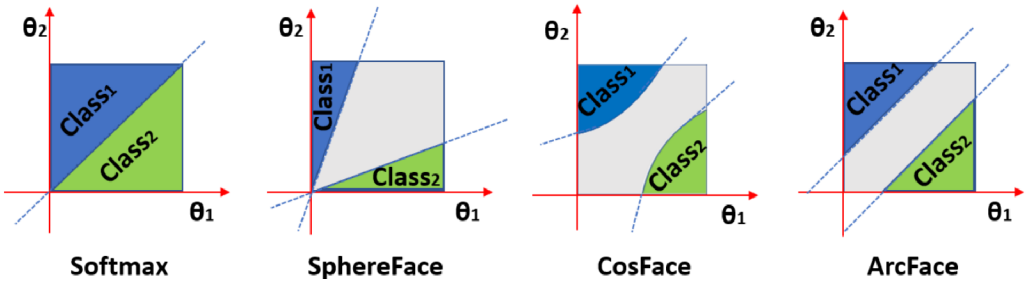
Na obrázku 3.11 je príklad porovnania chybovej funkcie *soft-max loss* a *ArcFace* pri trénovaní siete 2-D vektora vlastností pri ôsmych identitách obsahujúcich približne 1500 obrázkov na 1 identitu. Ako je znázornené na obrázku 3.11, strata softmax poskytuje zhruba oddeliteľné vektory vlastností, ale vytvára značnú nejednoznačnosť v hraniciach rozhodovania, zatiaľ čo chybová funkcia *ArcFace* vynúti zreteľnejšiu medzeru medzi blízkymi triedami.



(a) Softmax (b) ArcFace

Obr. 3.11: Porovnanie chybovej funkcie *soft-max loss* a *ArcFace* na 8 identitách. Bodky označujú jednotilivé obrázky a čiary sa vzťahujú na stredový smer každej identity[6].

Napriek numerickej podobnosti medzi *ArcFace* a predchádzajúcimi prácami má penalizácia *additive angular margin penalty* lepší geometrický atribút, pretože *angular margin* presne zodpovedá geodetickej vzdialenosti. Na obrázku 3.12 sú porovnané hranice rozhodovania v prípade binárnej klasifikácie. Autori článku ArcFace argumentujú, že presnosť algoritmu vychádza práve z lineárneho geometrického rozdelenia tried.



Obr. 3.12: Rozhodovacie hranice rôznych chybových funkcií v prípade binárnej klasifikácie. Prerušovaná čiara predstavuje rozhodovaciu hranicu a sivé oblasti sú okraje rozhodovania[6].

V článku ArcFace, autori využili široko používanú architektúru ResNet50 a ResNet100. Po poslednej konvolučnej vrstve využili štruktúru: batch normalizácie[12], dropout vrstvy[24], plne prepojenej vrstvy a znova batch normalizácie na získanie výstupného vektoru vlastností s 512 hodnotami. Ako trénovacie data samostatne využili datasety CASIA[33], VGGFace2[5], MS1MV2 a DeepGlint-Face.

Pri hodnotení modelu sa autorom podarilo dosiahnuť presnosť 99,83 % na datasete *Labeled Faces in the Wild*[11] a na datasete *YouTube Faces DB*[30] dosiahol presnosti 98,02 %.

3.6.3 SphereFace

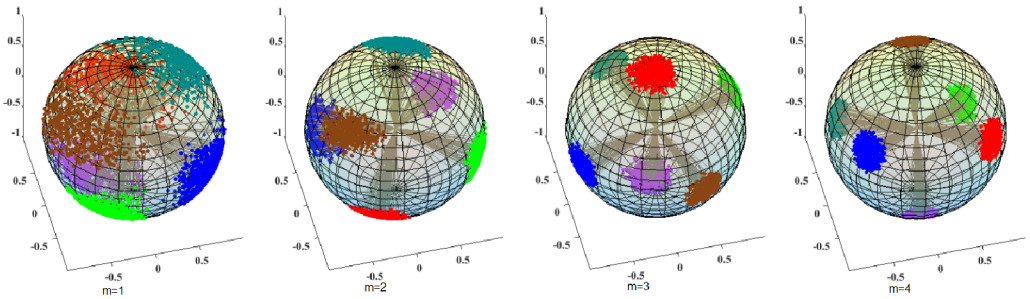
V dokumente SphereFace[15] sa autori zaoberali problémom hlbokého rozpoznávania tváre v rámci otvoreného protokolu. V tomto type rozpoznávania tváre sa očakáva, že ideálne vlastnosti tváre budú mať menšiu maximálnu vzdialenosť v rámci jednej triedy ako minimálnu vzdialenosť medzi jednotlivými triedami pri vhodne zvolenom metrickom priestore a taktiež tento typ obsahuje triedy, ktoré neboli prítomné v tréningových dátach. V čase publikácie práce, len málo existujúcich algoritmov však dokázalo efektívne splniť toto kritérium. Autori práce navrhli novú chybovú funkciu *angular softmax loss* (A-Softmax), ktorá umožňuje konvolučným neurónovým sieťam naučiť sa uhlovo diskriminačné vlastnosti. Geometricky je možné A-Softmax chybovú funkciu vnímať ako zavedenie diskriminačných obmedzení na hypersférický manifold, ktorý sa vnútorne zhoduje s predchádzajúcim, že na manifolde ležia aj jednotlivé tváre. Okrem toho je možné veľkosť *angular margin* kvantitatívne upraviť parametrom.

Autori prepracovali definíciu pre chybovú funkciu softmax ((3.5)) z hľadiska uhla medzi vektorom vlastností a váhovým vektorom zodpovedajúcim jeho triede v matici váh. Potom zjednodušili definíciu tým, že každý váhový vektor (W_j) normalizovali pomocou L_2 normalizácie a ignorovali hodnotu *bias* (b_j). Ako posledné pridali parameter m na riadenie citlivosti výrazu na uhol θ. Konečná definícia A-Softmax je nasledujúca:

$$L_{a_{ng}} = \frac{1}{N}\sum_{i} -\log\left(\frac{e^{||x_i||\psi(\theta_{y_i,i})}}{e^{||x_i||\psi(\theta_{y_i,i})} + \sum_{j\neq y_i} e^{||x_i||\cos(\theta_{j,i})}}\right), \tag{3.7}$$

kde $\psi(\theta_{y_i,i})$ je definované ako $(-1)^k \cos(m\theta_{y_i,i}) - 2k, \theta_{y_i,i} \in [\frac{k\pi}{m}, \frac{(k+1)\pi}{m}]$ a $k \in [0, m-1]$. $m \geq 1$ je celé číslo, ktoré riadi veľkosť uhlových okrajov (*angular margin*).

Niektoré z výhod tejto chybovej funkcie sú, že pracuje priamo s uhlovými premennými, čo je viac v súlade s vnútorným vlastnosťami dát, nevyžaduje komplexnú rekombináciu tréningových príkladov do dvojíc alebo trojíc a pomáha znižovať vzdialenosť medzi podobnými obrázkami a súčasne zvyšovať vzdialenosť medzi odlišnými triedami. Nevýhodou tochto článku je, že obsahuje niekoľko aproximácií a predpokladov (napr. m je obmedzené na celé číslo), taktiež keď sa m zvyšuje, lokálne minimá funkcie kosínus sa tiež dostanú do rozsahu možných θ, po ktorom je funkcia nemonotónna. Na obrázku 3.13 je zobrazená vizualizácia vektorov vlastností naučených s rôznymi m. 3-D vlastnosti sú premietnuté na guľe. Premietané body sú priesečníky vektorov vlastností a gule



Obr. 3.13: Vizualizácia vektorov vlastností naučených pomocou chybovej funkcie *angular margin loss* s rôznymi hodnotami m[15].

Autori na trénovanie modelu použili verejne dostupný webový tréningový dataset CASIA-WebFace, z ktorého odstránili obrázky identít vyskytujúcich sa v testovacích dátach. Konvolučná architektúra mala 64 vrstiev a z plne prepojenej vrstvy extrahovala vektory vlastností. Na verifikáciu modelu použili hodnotu *angular margin* $m = 4$. Pri hodnotení modelu sa autorom podarilo dosiahnuť presnosť 99,42 % na datasete *Labeled Faces in the Wild*[11] a na datasete *YouTube Faces DB*[30] dosiahol presnosti 95,00 %.

Kapitola 4

Edge computing

Edge computing [4][32][19] je nová výpočtová paradigma, ktorá vykonáva výpočty "na okraji siete" (angl. *at the edge of the network*). Okraj siete je miesto, kde zariadenia komunikujú údaje v reálnom čase do siete. Na rozdiel od *cloud computingu* sa *edge computing* posúva bližšie k používateľovi a bližšie k zdroju údajov. Na okraji siete je toto paradigma ľahké na lokálne ukladanie a spracovanie údajov v malom rozsahu.

S rýchlym rozvojom internetu všetkého (angl. *Internet of Everything*), rýchlo rastie počet inteligentných zariadení pripojených k internetu, ktoré generujú rozsiahle dáta na okraji siete. Obrovské množstvo údajov spôsobuje problémy, ako je zaťaženie šírky pásma, pomalá rýchlosť odozvy, slabé zabezpečenie a slabé súkromie v tradičných modeloch *cloud computingu*. Tradičný *cloud computing* už teda nestačí na podporu dnešných požiadaviek na inteligentné služby a inteligentné spracovanie údajov, a preto sa objavili technológie *edge computing*. *Edge computing* smeruje výpočtové údaje, aplikácie a služby preč z cloudových serverov na okraj siete. Výsledkom je, že vývojári aplikácií môžu využívať výpočtové systémy na zariadení tým, že ponúkajú používateľom služby, ktoré sú im bližšie. *Edge computing* sa vyznačuje vysokou šírkou pásma, ultranízkou latenciou a prístupom k sieťovým informáciám v reálnom čase, ktoré môžu využívať viaceré aplikácie. *Edge computing* je preto základom novej generácie *edge intelligence*, nasadenia algoritmov strojového učenia na edge zariadenia, kde sa generujú údaje.

Keďže sa *edge computing* stal hotspotom výskumu, existujú rôzne definície. Satyanarayanan, profesor na Carnegie Mellon univerzite, definuje *Edge computing* ako „nový výpočtový model, ktorý nasadzuje výpočtové a úložné zdroje (ako sú cloudlety, mikro dátové centrá alebo hmlové uzly atď.) na okraji siete bližšie k mobilným zariadeniam."Čínske konzorcium *Edge Computing* definuje *edge computing* ako „blízko okraja siete alebo zdroja údajov, otvorená platforma, ktorá integruje základné funkcie, ako sú siete, výpočtová technika, úložisko, aplikácie, a poskytuje špičkové inteligentné služby v blízkosti, aby splnila agilitu odvetvia. kľúčové požiadavky v oblasti pripojenia, podnikania v reálnom čase, optimalizácie údajov, aplikačnej inteligencie, bezpečnosti a ochrany osobných údajov." Firma *Zigbee Home Automation* poskytuje nasledujúcu definíciu: „*Edge computing* je nový výpočtový model, ktorý zjednocuje zdroje, ktoré sú blízko k používateľovi v geografickej vzdialenosti alebo vzdialenosti siete, aby poskytoval výpočtové, úložné a sieťové služby pre aplikácie."

Dnešná inteligentná spoločnosť je poháňaná potrebou inteligentných a prepojených služieb v rôznych odvetviach. *Edge* zariadenia sa rozšírili do všetkých aspektov spoločnosti, ako sú inteligentné domácnosti, sledovacie kamery, autonómne vozidlá, inteligentné výrobné roboty a ďalšie. V dôsledku toho sa počet pripojených zariadení neustále zvyšuje. Preto

množstvo dát generovaných zariadeniami na celom svete drasticky narastá. Na základe neustáleho a masívneho rastu objemu údajov a rôznych požiadaviek na spracovanie údajov sa ukázalo , že cloudové spracovanie veľkých údajov má mnohé nedostatky:

1. **V reálnom čase**: Prenos veľkého množstva údajov do cloudu má za následok veľké zaťaženie prenosového pásma siete, čo má za následok oneskorenie prenosu údajov. *Cloud computing* nebude schopný splniť obchodné požiadavky v reálnom čase.

2. **Spotreba energie**: Spotreba energie dátových centier sa výrazne zvýšila. *Cloud computing* nie je schopný uspokojiť rastúci dopyt po optimalizovanej spotrebe energie.

3. **Bezpečnosť a súkromie**: Nahrávanie údajov do cloudu a ich ukladanie v centralizovanom prostredí prináša riziká úniku alebo útokov na súkromie. Pravdepodobnosť útokov na dátových cestách je vyššia pri *cloud computingu* ako pri *edge computingu*, čo je spôsobené dlhšou cestou k serveru.

Preto sa technológia *edge computing* zameriava na poskytovanie služieb, a generovanie údajov a vykonávanie výpočtov na okraji siete. Cieľom *edge computing* je migrovať cloudovú sieť, výpočtové kapacity, úložné kapacity a zdroje na okraj siete a poskytovať inteligentné služby na okraji siete. Vyžaduje sa to na splnenie kritických potrieb odvetvia IT v podnikaní v reálnom čase, aplikačnej inteligencii, optimalizácii údajov, bezpečnosti a súkromí a na splnenie požiadaviek na nízku latenciu a veľkú šírku pásma v sieti.

4.1 Charakteristiky Edge Computing

Edge computing má niekoľko vlastností podobných *cloud computingu*, ale rozširuje cloud svojou špecifickou architektúrou:

- **Geografická distribúcia**: Aplikácie internetu vecí založené na senzorových sieťach majú veľký úžitok zo spracovania údajov lokálne prostredníctvom okrajových počítačových platforiem. Analýzu veľkých dát možno vykonávať rýchlo a s vyššou presnosťou. Systémy *Edge* umožňujú analýzu v reálnom čase a spracovanie umelej inteligencie vo veľkej mierke. Príklady zahŕňajú senzorové siete na monitorovanie prostredia, napríklad monitorovanie potrubí alebo systémy na predchádzanie kolíziám.

- **Podpora mobility**: Keďže počet mobilných zariadení rýchlo rastie, *edge computing* podporuje aj mobilitu na priamu komunikáciu s mobilnými zariadeniami.

- **Povedomie o polohe**: Sledovanie polohy umožňuje použitie technológií, ako je GPS, na nájdenie polohy zariadení. Povedomie o polohe teda môžu využívať počítačové aplikácie *Edge*, ako sú aplikácie pre bezpečnosť vozidiel a správa katastrof založených systémoch Edge.

- **Blízkosť**: Dostupnosť výpočtových zdrojov a služieb v miestnej blízkosti umožňuje užívateľom využiť informácie o kontexte siete na rozhodovanie o vyložení a rozhodovaní o používaní služieb.

- **Nízka latencia**: Nízka latencia *Edge computingu* umožňuje používateľom spúšťať ich aplikácie náročné na zdroje a citlivé na oneskorenie na *edge* zariadení. Takéto aplikácie zahŕňajú prepojené vozidlá, vzdialené monitorovanie zdravotného stavu, skladovú logistiku a priemyselné riadiace systémy.

- **Prípady intenzívneho používania**: Rastúce množstvo údajov generovaných nasadením internetu vecí je dnes náročných na šírku pásma, najmä údaje z videa z monitorovacích kamier. Umiestnenie výpočtových zdrojov čo najbližšie k zdrojom údajov s veľkou šírkou pásma znamená, že do vzdialených dátových centier v cloude je potrebné posielať oveľa menej údajov. Napríklad videá a údaje zo senzorov z nebezpečných miest môžu byť spracované lokálne, aby poskytli informácie v reálnom čase záchranárom v aplikáciách verejnej bezpečnosti.

Rozdiely medzi Edge Computing a Cloud Computing

Tradičný *cloud computing* prenáša všetky dáta do *cloud computingového* centra cez sieť a rieši problémy s výpočtovou technikou a úložiskom centralizovaným spôsobom. Vznik *edge computingu* však nenahradí *cloud computing*. Tieto dva koncepty sa navzájom dopĺňajú a umožňujú vo väčšej miere digitálnu transformáciu odvetví. *Edge computing* je pokročilá verzia *cloud computingu*, ktorá znižuje latenciu približovaním služieb ku koncovým používateľom. Okrem toho minimalizuje zaťaženie cloudu poskytovaním zdrojov a služieb v sieti *Edge*. V niektorých internetových službách je potrebné niektoré údaje po spracovaní *edge computingom* vrátiť do cloudu na spracovanie, ako je napríklad hĺbková analýza dolovania a zdieľania údajov, čo si vyžaduje spoluprácu oboch konceptov.

Hlavné rozdiely medzi *cloud computingom* a *edge computingom* sú nasledovné:

- **Cloud**: Hlavnou črtou *cloud computingu* je to, že dokáže spracovať veľké množstvo údajov, vykonávať hĺbkovú analýzu a zohráva dôležitú úlohu pri spracovaní údajov, ktoré nie je v reálnom čase, ako je napríklad obchodné rozhodovanie.
- **Edge**: Na druhej strane, *edge computing* sa zameriava lokálne. Preto môže zohrávať lepšiu úlohu v inteligentnej analýze malého rozsahu v reálnom čase, ako je napríklad uspokojovanie potrieb distribuovaných služieb v reálnom čase.

Edge computing je potrebný na zdieľanie tlaku cloudu a na prevzatie úloh v rámci jeho pôsobnosti. Okrem toho lokálne spracovanie údajov znižuje riziko straty veľkého rozsahu údajov. Tieto faktory prinášajú stabilitu do služieb, ktoré zahŕňajú pripojené zariadenia v sieti tkzv. internet vecí.

Výhody Edge Computingu

Systémy *Edge Computing* ukladajú a spracúvajú zozbierané údaje lokálne na okrajových zariadeniach bez ich nahrávania na platformu *cloud computingu*. To prichádza s niekoľkými dôležitými výhodami na zníženie tlaku na šírku pásma siete a ďalšie nevýhody *cloud computingu*.

- **Výkon**: Výpočtová technológia *edge computing* ponúka rýchle spracovanie a analýzu údajov a rýchlu odozvu umožňujúcu služby v reálnom čase. Používateľom poskytuje množstvo služieb rýchlej odozvy, najmä v oblasti automatického riadenia, inteligentnej výroby, monitorovania videa a ďalších, kde je rýchla spätná väzba kritická. Napríklad *Edge computing* umožňuje aplikácie počítačového videnia v reálnom čase.
- **Súkromie a bezpečnosť**: Centralizovaný prístup tradičného *cloud computingu* vyžaduje, aby sa všetky dáta nahrávali do cloudu na jednotné spracovanie. Spracovanie v cloude preto vystavuje značné riziká straty a úniku údajov. *Edge computing* je založený na lokálnych dátach a nie je potrebné nahrávať dáta do cloudu, čím sa *edge*

computing vyhne rizikám, ktoré prináša sieťový prenos. Výsledkom je, že útoky a zlyhania ovplyvňujú iba lokálne dáta, nie všetky dáta. Výpočet na zariadení umožňuje zaručiť bezpečnosť údajov. Okrem toho výpočtová technika na zariadení umožňuje distribuované trénovanie modelov neurónových sietí.

- **Efektívnosť**: Výpočet na zariadení znižuje množstvo dát prenášaných v sieti, znižuje náklady na prenos a tlak na šírku pásma siete, znižuje spotrebu energie miestnych zariadení a zlepšuje výpočtovú efektivitu. *Edge computing* poháňa vývoj vysoko efektívneho hardvéru umelej inteligencie, ako sú akcelerátory hlbokého učenia (napr. *Vision Processing Units*).

- **Spoľahlivosť**: Technológia *edge computing* poskytuje metódy na to, aby boli služby stabilnejšie, robustnejšie a vysoko dostupné. Vysoká spoľahlivosť pripojených systémov je obzvlášť dôležitá v kritických aplikáciách, ktoré sú citlivé na odpojenie siete, čo môže viesť k fatálnym následkom (napr. lekárske monitorovanie alebo dopravné systémy).

4.2 Príklady Edge Computingu

S *Edge Computingom* je možné poháňať škálovateľné, kritické a súkromné aplikácie umelej inteligencie. Keďže *Edge Computing* je stále veľmi nová technológia, v blízkej budúcnosti možno očakávať oveľa viac aplikácií. V tejto sekcii budú popísané príklady *Edge computingu*.

Aplikácie chytrého počítačového videnia

Aplikácie počítačového videnia, ako je napríklad živá video analytika, autonómne vozidlá alebo sledovanie premávky s pomocou umelej inteligencie sú skvelým príkladom možností, ktoré ponúka *edge computing*. Koprocesory vyvinuté od Intel s názvom *Visual Processing Units* na pohánanie vysokovýkonných aplikácií počítačového videnia pre *edge* zariadenia sú skvelé práve pre tieto typy aplikácii.

Autonómne vozidlá sú schopné bezpečne jazdiť z jedného bodu do druhého bez akéhokoľvek ľudského zásahu. Umožňuje to celý rad pokročilých technológií, ktorých jadrom je umelá inteligencia na *edge* zariadení. Automobily Tesla sú dobrým príkladom toho, ako môže kombinácia umelej inteligencie a internetu vecí drasticky zmeniť automobilový priemysel. Tie využívajú senzory, prístroje, kamery a množstvo pokročilých technológií, ktoré spolupracujú na zaistení bezpečnej automatizovanej jazdy. Počítače autonómnych vozidiel dokážu zhromažďovať a analyzovať informácie z niekoľkých zdrojov v reálnom čase, aby mohli robiť čo najpresnejšie jazdné rozhodnutia. V súlade s tým sú autonómne vozidlá hlavným príkladom *Edge Computingu*.

Senzory a zariadenia založené na umelej inteligencii sú veľmi obľúbené pre analýzu premávky v reálnom čase v chytrých mestách. Takéto systémy umelej inteligencie zahŕňajú inteligentné drony na posúdenie úrovne preťaženia premávky alebo vykonanie analýzy davu. V doprave systémy *edge computingu* založené na počítačovom videní automaticky identifikujú nehody, dopravné priestupky, zastavené vozidlá a ďalšie iné veci.

Zdravotnícke aplikácie

Zdravotnícke aplikácie založené na umelej inteligencii, ako je vzdialená chirurgia a diagnostika, ako aj monitorovanie vitálnych funkcií pacienta, sú väčšinou založené na *edge* zariadeniach, ktoré vykonávajú umelú inteligenciu na hranici. Lekári môžu použiť vzdialenú platformu na ovládanie chirurgických nástrojov na diaľku, kde sa cítia bezpečne a pohodlne.

Zábava

Aplikácie pre zábavu zahŕňajú virtuálnu realitu, rozšírenú realitu a zmiešanú realitu, ako je napríklad streamovanie video obsahu do okuliarov virtuálnej reality. Veľkosť takýchto okuliarov je možné zmenšiť prenesením výpočtov z okuliarov na *edge* servery v blízkosti koncového zariadenia. Napríklad Microsoft nedávno predstavil HoloLens, holografický počítač zabudovaný do náhlavnej súpravy pre zážitok z rozšírenej reality. Cieľom spoločnosti Microsoft je navrhnúť štandardné počítačové nástroje, analýzu údajov, lekárske zobrazovanie a špičkové nástroje s využitím HoloLens.

4.3 Akcelerátory pre neurónové siete

Inferencia pri umelej inteligencie je proces prevzatia natrénovaného modelu neurónovej siete, zvyčajne vytvoreného pomocou hlbokého učenia, a jeho následného nasadenia do výpočtového zariadenia ako je napríklad Neural Compute Stick 2. Toto zariadenie potom spracuje prichádzajúce dáta (zvyčajne obrázky alebo video), aby vyhľadalo a identifikovalo akýkoľvek vzor,ktorý bolo naučené rozpoznávať.

Inštalácia počítača s nízkou spotrebou energie s integrovaným akcelerátorom inferencie AI blízko zdroja dát má za následok oveľa rýchlejšie časy odozvy a efektívnejšie výpočty. Okrem toho vyžaduje menšiu šírku pásma internetu a grafický výkon. V porovnaní s cloudovou inferenciou môže inferencia na pri zdroji dát potenciálne skrátiť čas výsledku z niekoľkých sekúnd na zlomok sekundy. Akcelerátory umelej inteligencie môžu výrazne zvýšiť rýchlosť inferencie alebo spracovávania modelu na zariadení a možno ich použiť aj na vykonávanie špeciálnych úloh založených na neurónových sieťach, ktoré nemožno vykonávať na bežnom CPU. Keďže sa umelá inteligencia stáva kľúčovou hnacou silou *edge computingu*, kombinácia hardvérových akcelerátorov a softvérových platforiem sa stáva dôležitou súčasťou pre beh modelov na inferenciu. Okrem Neural Compute Sticku, existujú ešte populárne akcelerátori od NVidie nazývaný *NVIDIA Jetson* a od Googlu *Google Coral Edge TPU*. Použitie týchto akcelerátorov má svoje výhody najmä v oblasti počítačového videnia, kde je pracovná záťaž vysoká a úlohy, ktoré sa majú vypočítať, sú veľmi náročné na dáta. Používanie hardvérovej akcelerácie umelej inteligencie pre zariadenia *edge computingu* má preto mnoho výhod, z ktorých hlavnými sú rýchlosť a efektívnosť.

4.3.1 Neural Compute Stick 2

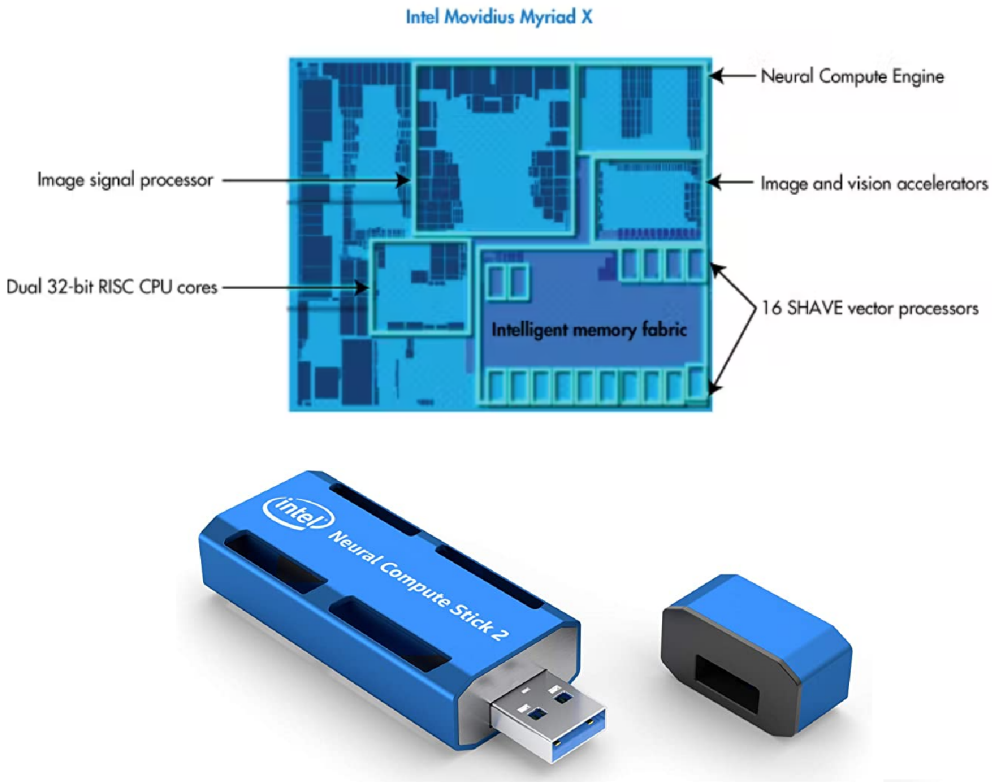
Neural Compute Stick 2 (NCS2) je malé zariadenie v tvare USB kľúča na inferenciu modelou neurónových sietí. Intel Neural Compute Stick 2 je poháňaný procesorom Intel Movidius X VPU, ktorý poháňa aplikácií umelej inteligencie na zariadení pri vysokom výkone s mimoriadne nízkou spotrebou energie. Tento typ procesoru je špeciálne navrhnutý na akceleráciu úloh strojového učenia, ktorý prináša počítačové videnie a aplikácie umelej inteligencie na špičkové zariadenia, ako sú drony, inteligentné kamery, inteligentná domácnosť, zabezpečenie, VR/AR headsety a 360-stupňové kamery. Akcelerátory, ako je procesor NCS2, sú užitočné na urýchlenie dátovo-náročnej inferencie hlbokého učenia na zariadeniach použitých pri *edge computingu* veľmi nákladovo efektívnym spôsobom. Tieto akcelerátory fungujú tak, že pomáhajú procesorovej jednotke tým, že preberajú matematickú záťaž potrebnú na spustenie modelov hlbokého učenia. Akcelerátory umožňujú spustenie modelov hlbokého učenia pri nízkych nákladoch, nízkej spotrebe energie a vyšších rýchlostiach inferencie. Prínosy sa primárne merajú pomocou priepustnosti, latencie a efektívnosti. Na základe maximálneho výkonu operácií za sekundu vo všetkých dostupných výpočtových jednotkách dosahuje Intel

Neural Compute Stick 2 celkový výkon viac ako 4 bilióny operácií za sekundu. Ako riešenie *edge computingu* ponúka všetky výhody *edge* zariadenia, ako je súkromie prostredníctvom lokálneho spracovania, výkon vďaka nízkej latencii a spoľahlivosť.

Intel Movidius Myriad X Vision Processing Unit (VPU) je prvým VPU od Intelu, ktorý obsahuje Neural Compute Engine – vyhradený hardvérový akcelerátor pre inferenciu hlbokých neurónových sietí. Movidius Myriad X VPU sa predáva ako čip vložený do Neural Compute Sticku, je špeciálne vytvorený na spracovanie obrázkov a video vstupov. Uloženie čipu v USB obale umožňuje, aby bol ľahko kompatibilný so zariadením Raspberry Pi alebo v zariadení Intel NUC (akákoľvek populárna počítačová architektúra, ako sú počítače x86). Engine Intel Movidius Neural Compute Stick je hardvérová súčasť vo forme čipu navrhnutá na prevádzkovanie hlbokých neurónových sietí pri vysokej rýchlosti a nízkej spotrebe bez kompromisov v presnosti, čo umožňuje procesy počítačového videnia v reálnom čase. Tento *engine* je implementovaný USB obale spolu s procesorom Movidius Myriad X. Tento procesor taktiež obsahuje stereo hĺbkový akcelerátor, ktorý dokáže súčasne spracovať 6 kamerových vstupov (3 stereo páry), z ktorých každý beží v rozlíšení 720 pixelov pri snímkovej frekvencii 60 Hz. Pri súbežnom používaní *Neural Compute Engine-u*, 16 výkonných jadier SHAVE a vysoko priepustnej pamäte robí neurónovú výpočtovú jednotku Intel Movidius Myriad X ideálnu na tréning a nasadenie hlbokých neurónových sietí a aplikácií počítačového videnia.

Referenčné hodnoty ukazujú relatívnu mieru úspešnosti rôznych algoritmov strojového učenia na rôznych inferenčných *enginoch*. Pre populárny ssd300-CF (Caffe backend) je priepustnosť priamo úmerná presnosti, zatiaľ čo miera latencie je nepriamo úmerná kvalite. V súlade s tým akcelerátor Movidius dosahuje vyššiu priepustnosť pre ssd300-CF ako ktorýkoľvek z Core i3 až i9 (okrem Intel Core i9-10920X) procesorov od Intelu. Okrem toho má nižšiu latenciu ako ktorýkoľvek inferenčný *engine*, čo z neho robí relatívne robustnú platformu na spúšťanie modelov počítačového videnia.

Intel poskytuje sadu nástrojov pre vývojárov OpenVINO na vývoj a nasadenie aplikácii, ktoré sa najlepšie používajú v spojení s Movidius Myriad X. Sada nástrojov umožňuje rýchlejšiu inferenciu modelov hlbokého učenia vytváraním efektívnych a robustných aplikácií počítačového videnia. OpenVINO bude podrobnejšie popísaný v nasledujúcej časti. Na obrázku 4.1 je zobrazená architektúra procesoru Intel Movidius Myriad X VPU a taktiež obrázok akcelerátora Neural Compute Stick 2.



Obr. 4.1: Architektúra procesoru Intel Movidius Myriad X VPU a Neural Compute Stick 2[31].

4.3.2 Použitý obslužný software

OpenVINO

OpenVINO je multiplatformová sada nástrojov na hlboké učenie vyvinutá spoločnosťou Intel. Názov znamená „*Open Visual Inference and Neural Network Optimization*“. OpenVINO sa zameriava na optimalizáciu inferencie neurónových sietí pre hardvérové platformy Intel. Sada nástrojov je zadarmo na použitie pod licenciou *Apache License* verzie 2.0 a má dve verzie: OpenVINO toolkit, ktorý je podporovaný open-source komunitou a Intel Distribúcia sady nástrojov OpenVINO, ktorú podporuje Intel. Pomocou sady nástrojov OpenVINO je možné vybrať a nasadiť modely, ktoré sú vopred natrénované (YOLOv3, ResNet50 atď.) prostredníctvom vysokoúrovňového *C++ Inference Engine API* integrovaného s aplikačnou logikou. OpenVINO ponúka integrované funkcionality na urýchlenie vývoja aplikácií a riešení, ktoré riešia niekoľko úloh pomocou počítačového videnia, automatického rozpoznávania reči, spracovania prirodzeného jazyka, odporúčacích systémov, strojového učenia a ďalších.

Hlboké neurónové siete urobili v posledných rokoch značný pokrok v mnohých priemyselných doménach, čím posunuli presnosť algoritmov počítačového videnia na novú úroveň. Nasadenie a vytvorenie takýchto presných a užitočných modelov si však vyžaduje prispôsobenie hardvéru a výpočtových metód. OpenVINO umožňuje optimalizáciu modelov hlbokých neurónových sietí pre inferenciu efektívnym procesom prostredníctvom integrácie rôznych nástrojov. Sada nástrojov OpenVINO je založená na najnovších generáciách neurónových sietí, ako sú konvolučné neurónové siete, ako aj na rekurentných sieťach a

sieťach založených na pozornosti.OpenVINO pokrýva úlohy počítačového videnia aj nepočítačového videnia v rámci hardvéru Intel. Maximalizuje výkon a urýchľuje vývoj aplikácií. OpenVINO má za cieľ urýchliť pracovné zaťaženie umelej inteligencie a zrýchliť čas uvedenia na trh pomocou knižnice vopred určených funkcií, ako aj vopred optimalizovaných jadier.

Výhody OpenVINO

- **Zrýchlenie výkonu**: Zrýchlenie pracovnej záťaže počítačového videnia povolením jednoduchých vykonávacých metód na rôznych procesoroch a akcelerátoroch Intel, ako sú CPU, GPU/*Intel Processor Graphics*, VPU (Intel Neural Compute Stick s Myriad X) a FPGA.
- **Zjednodušenie nasadenia aplikácii s hlbokými neurónovými sieťami**: Využitie funkcií hlbokého učenia založených na konvolučných neurónových sieťach pomocou jednej spoločnej API. Okrem toho OpenVINO poskytuje viac ako 30 vopred pripravených modelov a zdokumentovaných vzoriek kódu. S viac ako 100 verejnými a vlastnými modelmi OpenVINO zefektívňuje prácu s hlbokým učením tým, že poskytuje jednu centralizovanú metódu na implementáciu desiatok modelov hlbokého učenia.
- **Možnosť rozšírenia a prispôsobenia**: Kernely OpenCL (*Open Computing Language*) a ďalšie nástroje ponúkajú otvorený štandardný spôsob bez licenčných poplatkov, ako pridať vlastné časti kódu priamo do aplikácie, prispôsobiť vrstvy modelu neurónových sietí bez nutnosti využitia konkrétneho frameworku a implementovať paralelné programovanie rôznych akcelerátorov.
- **Inovácia aplikácií umelej inteligencie**: Kompletná sada nástrojov *Deep Learning Deployment Toolkit* v rámci OpenVINO umožňuje používateľom rozšíriť a optimalizovať aplikácie umelej inteligencie pomocou nástrojov, ako je *Model Optimizer*, *Intermediate Representation*, *nGraph Integration* a ďalších.

Pracovný postup pri OpenVINO

Ešte pred použitím OpenVINO je potrebné si určiť prostredie a typ modelu a frameworku pre strojové učenie. OpenVINO podporuje operačné systémy ako Linux, Windows, macOS a Raspbian. Taktiež podporuje frameworky pre strojové učenie, ako sú TensorFlow, Caffe, MXNet a Kaldi, ako aj formát modelu *Open Neural Network Exchange* (ONNX). Podpora tiež zahŕňa väčšinu vrstiev v rámci týchto frameworkov. Okrem toho je možné OpenVINO rozšíriť o podporu vlastných vrstiev.

1. **Príprava a trénovanie modelu**: Nájdenie open-source predtrénovaného modelu alebo vytvorenie vlastného modelu. *Open Model Zoo* v rámci OpenVINO poskytuje open-source repozitár optimalizovaných, vopred pripravených modelov pre rôzne všeobecné úlohy, ako je rozpoznávanie objektov, odhad pózy osoby, detekcia textu a rozpoznávanie pohybu. Overená podpora pre verejné modely a zbierka vzoriek kódu a ukážok sú tiež open source v rámci repozitáru. Ďalej je potrebné nakonfigurovať nástroj *Model Optimizer*, buď použitím dostupných skriptov alebo manuálnym procesom pre framework, ktorý bol použitý na trénovanie modelu.

2. **Konvertovanie, optimalizácia modelu a príprava na inferenciu**: Pomocou nástroja *Model Optimizer* je potrebné konvertovať model na prechodnú reprezentáciu (angl. *Intermediate representation* - IR), ktorá je reprezentovaná dvojicou súborov (.xml a .bin). Tieto súbory popisujú topológiu siete a obsahujú váhy a hodnoty bias modelu. Spolu s dvojicou súborov (.xml a .bin) poskytuje nástroj *Model Optimizer* aj diagnostické správy, ktoré pomáhajú pri ďalšom ladení. Okrem toho *open-source* nástroj *Accuracy Checker*, môže pomôcť pri overovaní presnosti modelu. Pre urýchlenie inferencie a konvertovanie modelov na reprezentácie vhodné pre hardvér (napríklad s nižšou presnosťou, ako je INT8), ktoré nevyžadujú pretrénovanie, je možné použiť nástroj na optimalizáciu po trénovaní (angl. *Post-Training Optimization Tool*).

3. **Inferencia a optimalizácia výkonu**: Spustenie nástroja *Inference Engine*. Načítanie a kompilácia optimalizovaného modelu a spustenie inferencie na vstupných dátach a následný výstup výsledkov nástrojom. *Inference Engine* je vysokoúrovňové inferenčné API implementované v C++ alebo v jazyku Python s rozhraním, ktoré je implementované ako dynamicky načítané moduly pre každý typ hardvéru. Poskytuje optimálny výkon pre každý hardvér bez potreby implementácie a údržby viacerých častí kódu. OpenVINO poskytuje ďalšie nástroje, ktoré pomáhajú zlepšiť výkon modelu:

    - *The Benchmark App* - analyzuje výkon modelu.
    - *The Cross-Check Tool* - porovnáva presnosť a výkon medzi dvoma po sebe nasledujúcimi inferenciami.
    - *Deep Learning Workbench* - umožňuje vizualizovať, dolaďovať a porovnávať výkon modelov.

4. **Nasadenie aplikácie**: Na nasadenie aplikácie sa používá nástroj *Inference Engine*. *Inference Engine* pri nasadení aplikácie je možné spustiť ako hlavný objekt s rozšíreniami, ktoré potom načítajú optimalizovaný sieťový model *nGraph* do konkrétneho cieľového zariadenia. S načítanou sieťou môže *Inference Engine* akceptovať dáta a požiadavky na spustenie inferencie a poskytovať výstupné dáta. Nasadenie do reálneho prostredia je možné pomocou nástroja *Deployment Manager*, ktorý vytvorí balík zostavením modelu, IR súborov, aplikácie a súvisiacich závislostí pre cieľové zariadenie.

OpenVINO ponúka množstvo funkcií, ktoré sú užitočné. Jednou s hlavných vlastností je spustenie inferencie na viacerých zariadeniach. Procesory Intel obsahujú silné jadrá x86 poháňané širokou škálou integrovaných grafických kariet čož je špičkový hardvér, ktorý umožňuje zníženie zaťaženia výpočtov. Príklady zníženia výpočtovej záťaže zahŕňajú prípady, keď integrovaná grafika umožňuje používateľom presunúť výpočty na integrovaný procesor Intel, ktorý je už vstavaný, pričom procesor sa používa na malé, interaktívne funkcie alebo funkcie s nízkou latenciou. *Runtime* v sade nástrojov Intel OpenVINO je možné použiť na spustenie inferencie na integrovanej grafike ako aj na akomkoľvek inom podporovanom cieľovom zariadení (napríklad CPU alebo Neural Compute Stick 2). Ako bolo spomenuté jednou z charakteristických vlastností OpenVINO je spustenie inferencie na viacerých zariadeniach. Kompatibilita s viacerými zariadeniami umožňuje vývojárom beh inferencie na kombinácii počítačových zariadeniach na jednom systéme. Toto umožňuje maximalizovať výkon inferencie. *Multi-device* modul ďalej umožňuje používateľom plne využívať výhody kombinácie hardvérových a softvérových doplnkov. Modul využíva dostupné CPU a integrované GPU na plné využitie systému.

Kapitola 5

Návrh a implementácia

Táto kapitola sa venuje návrhu, implementácii aplikácie rozpoznávania ôsob na vstavanom systéme s využitím akcelerátoru Neural Compute Stick 2 a použitým hardwarom a softwarom k vytvorení aplikácie.

5.1 Návrh aplikácie

5.1.1 Analýza problému

Problém rozpoznávania ôsob podľa tváre sa delí na tri podpodproblémy, Detekcia tváre v obraze, identifikácia významných bodov v tvári a následná identifikácia osoby na základe tváre. Aplikácia bude používať tri modely, ktoré budú riešiť vyššie spomenuté podproblémy. Vstupom aplikácie bude video záznam alebo vstup z kamery pripojenej ku vstavanému systému. Tento vstavaný systém s kamerou bude podobný inteligentej domácej kamere použitej napríklad pred vchodovými dverami na monitorovanie ôsob nachádzajúcich sa pred dverami. Keďže detekovaná tvár osoby sa môže nachádzať v rôznej vzdialenosti od kamery, aplikácia by mala dokázať identifikovať osobu v rozumnej vzdialenosti od kamery. Detekované a rozpoznané osoby budú odoslané pomocou internetu ako notifikácie uživateľovi. Pri rozpoznávaní ôsob na videu je potrebné spracovať každý jeden snímok a vykonať detekciu a rozpoznávanie na každom snímku. Tento proces je výpočtovo náročný pre vstavané systémy a preto tento proces bude akcelerovaný pomocou Neural Compute Stick-u. Dôležitou časťou systému pre rozpoznávanie tvári je predom vytvorená databáza ôsob. Na základe tejto databáze systém dokáže určiť meno detekovanej osoby.

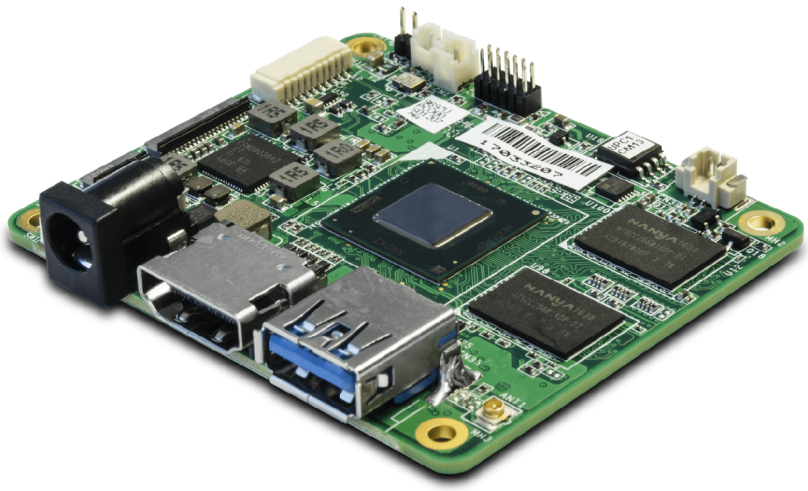
5.1.2 Použité nástroje

Vstavaný systém

Ako vstavaný systém bol použitý *single-board* počítač od spoločnosti *UP! Bridge the gap*[1]. Bol použitý počítač vo verzii UP Core, čo je základná verzia , ktorú ponúkajú. Táto miniatúrna doska v rozmeroch približne 10x10 centimetrov ponúka vysoký výkon s nízkym spotrebom energie. Doska obsahuje procesor Intel® Atom™ x5 Z8350 s veľkosťou pamäte RAM 2GB a s vnútornou pamäťou až 32GB. Tento procesor, taktiež obsahuje internú grafickú kartu 8. generácie Intel Gen 8 HD 400. Doska obsahuje konektor USB 3.0 a HDMI, taktiež je v doske zabudovaná WiFi. Na dosku bol nainštalovaný operačný systém Linux

[1] https://up-board.org/

v distribúcii Ubuntu vo verzii 20.04. Doska je napájaná pomocou 5V konektora. Kamera použítá spolu so vstavaným systémom sa pripája pomocou USB. Kamera podporuje širokú škálu rozlíšení v rýchlosti snímania obrazu 30 snímkov za sekundu.



Obr. 5.1: *Single-board* počítač od spoločnosti *UP! Bridge the gap.*

Python

Ako implementačný jazyk bol zvolený jazyk Python[2]. Tento jazyk podpruje knižnica OpenVINO, ktorá je potrebná pre prácu s akcelerátorom. V dnešnej dobe je tento jazyk veľmi populárny. Python má jednoduchú syntax, vďaka ktorej je práca s ním jednoduchšia. Taktiež je to open-source softvér a dá sa bezsplatne používať. Ponúka vysokú úroveň abstrakcie. Taktiež ponúka mnoho balíkov, ktoré boli vytvorené tretími stranami pre rozšírenie možnosti práce s týmto jazykom.

Knižnica **ZeroMQ**[3] bola použitá pre zasielanie správ cez sieť. ZeroMQ je asynchrónna knižnica pre posielanie sieťových správ známa svojim vysokým výkonom. Jej zamýšľané použitie je pre distribuované systémy, ako aj pre konkurentné systémy. Stručne povedané, ZMQ umožňuje posielať správy (binárne dáta, serializované dáta, jednoduché reťazce atď.) cez sieť rôznymi metódami, ako je TCP alebo multicast, ako aj medzi procesmi. ZeroMQ poskytuje celý rad aplikačných jazykových rozhraní, ktoré bežia na väčšine operačných systémov a umožňujú bezproblémovú komunikáciu medzi všetkými druhmi programov. Poskytuje tiež kolekciu vzorov, ako napríklad žiadosť-odpoveď (angl. *request-reply*) a publikovanie-odoberanie (angl. *publish-subscribe*). Vzor publikovanie-odoberanie bol použitý pri implementácii zasielania notifikácii o detekcii ôsob v aplikácii.

Sockety sú hlavnou časťou knižnice ZeroMQ. ZMQ obsahuje niekoľko rôznych socketov, z ktorých každý má svoje vlastné vlastnosti a prípady použitia. Sockety možno kombinovať mnohými rôznymi spôsobmi, aj keď existuje množstvo kombinácií socketov, ktoré sú nekompatibilné. Sockety PUB a SUB využívajú model publikovania-odoberania. Základnou myšlienkou tochto modelu je, že socket PUB zasiela správy a všetky súvisiace sockety SUB tieto správy prijímajú. Táto komunikácia je striktne jednosmerná, SUB sockety neposielajú žiadne odpovede ani potvrdenia.

Knižnica **OpenCV**[4] bola použitá pre prácu s vstupnými dátami, ich spracovaním a zobrazením. OpenCV je obrovská open-source knižnica pre počítačové videnie, strojové učenie a spracovanie obrazu. OpenCV podporuje širokú škálu programovacích jazykov ako Python, C++, Java atď. Dokáže spracovať obrázky a videá na identifikáciu objektov, tvárí alebo dokonca aj rukopisu človeka. Python verzia OpenCV nie je nič iné ako len 'obalená' trieda pre pôvodnú knižnicu, ktorá je napísana v C++.

5.1.3 Použité modely

Modely neurónových sietí použitých v aplikácii pochádzajú z *Open Model Zoo* pre OpenVINO. Open Model Zoo[5] pre OpenVINO poskytuje širokú škálu bezplatných, vopred natrénovaných modelov hlbokého učenia a demo aplikácií, ktoré poskytujú šablóny pre vytváranie aplikácií. Modely a ukážky aplikácii sú dostupné v repozitári *Open Model Zoo GitHub* a sú licencované na základe licencie *Apache version 2.0*. Pre identifikáciu tváre je možné použiť 3 modely. Tieto modely sú popísané v sekcii 3.6. Modely boli prekonvertované do prechodnej reprezentácie (angl. *Intermediate Representation*) pomocou nástroja *Model Converter*. Túto reprezentáciu využíva OpenVINO. Model je reprezentovaný dvoma súbormi: súborom XML a binárnym súborom (.bin). Súbor XML popisuje topológiu siete pomocou značiek *<layer>* pre operačný uzol a značiek *<edge>* pre spojenia toku dát. Každá operácia má pevný počet

[2] https://www.python.org/
[3] https://zeromq.org/
[4] https://opencv.org/
[5] https://docs.openvino.ai/latest/model_zoo.html

atribútov, ktoré definujú typ operácie, ktorá je použitá pre daný uzol. Napríklad operácia Konvolúcie má také atribúty ako *dilation*, *stride*, *pads-begin* a *pads-begin*. Súbor XML nemá veľké konštantné hodnoty, ako sú konvolučné váhy. Namiesto toho odkazuje na časť sprievodného binárneho súboru, ktorý uchováva takéto hodnoty v binárnom formáte. Modely použité v aplikácii sú dostupné v dvoch presnostiach s pohyblivou rádovou čiarkou (FP16 a FP32). Modely s polovičnou presnosťou (FP16) majú menší rozsah. FP16 môže viesť k lepšiemu výkonu tam, kde stačí polovičná presnosť. Neural Compute Stick 2 podporuje iba modely preložené s FP16. U každého použitého modelu je zobrazená hodnota *FLOPS* (počet operácií s pohyblivou rádovou čiarkou za sekundu) a počet parametrov daného modelu (*Params*). FLOP sa často používa na popis toho, koľko operácií je potrebných na spustenie jednej inferencie daného modelu

Model pre detekciu tváre

- **face-detection-retail-0044**: Detektor tváre, ktorý je založený na architektúre *SqueezeNet light* s jedným detektorom *Single Shot MultiBox Detector* pre vnútorné a vonkajšie scény nasnímané prednou kamerou. Architektúra pozostáva z takzvaných *Fire* modulov na zníženie počtu výpočtov. Výstupom sú hraničné boxy detekových tvári v obrázku (angl. *Bounding Boxes*). GFLOPS: 1,067 MParams: 0,588

Model pre detekciu významných bodov tváre

- **landmarks-regression-retail-0009**: Jedná sa o ľahký regresor významných bodov tváre. Má klasický konvolučný dizajn: naskladané 3x3 konvolúcie, batch normalizácie, aktivácie PRELU a pooling vrstvy. Finálna regresia sa vykonáva pomocou plne prepojených vrstiev. Model predpovedá päť bodov tváre: dva body pre oči, jeden pre nos a dva pre kútiky pier. Všetky výstupné súradnice sú normalizované tak, aby boli v rozsahu 0 až 1. GFLOPS: 0,021 MParams: 0,191

Modely pre rozpoznávania tváre Použité modely pre rozpoznávanie tváre sú popísané v sekcii 3.6. Výstupom týchto modelov je vektor príznakov danej tváre s 512 hodnotami.

- **facenet-20180408-102900**: Model bol konvertovaný z frameworku *TensorFlow*. GFLOPS: 2,846 MParams: 23,469
- **Sphereface**: Model bol konvertovaný z frameworku *Caffe*. GFLOPS: 3,504 MParams: 22,671
- **face-recognition-resnet100-arcface-onnx**: Model bol konvertovaný z frameworku *MXNet*. GFLOPS: 24,2115 MParams: 65,1320

5.2 Implementácia

V tejto sekcii je popisaný príncip fungovania aplikácie. V rámci zadania nebolo vyžadované aby aplikácia mala uživatelské rozhranie, preto vytvorená aplikácia je ovládaná pomocou príkazovej riadky. Modely hlbokého učenia, ktoré budú použité pre detekciu tváre, detekciu významných bodov a následnu identifikáciu tváre sa zadávajú prostredníctvom vstupného parametru aplikácie. Vstupom aplikácie môže byť samostatný obrázok, priečinok obrázkov,

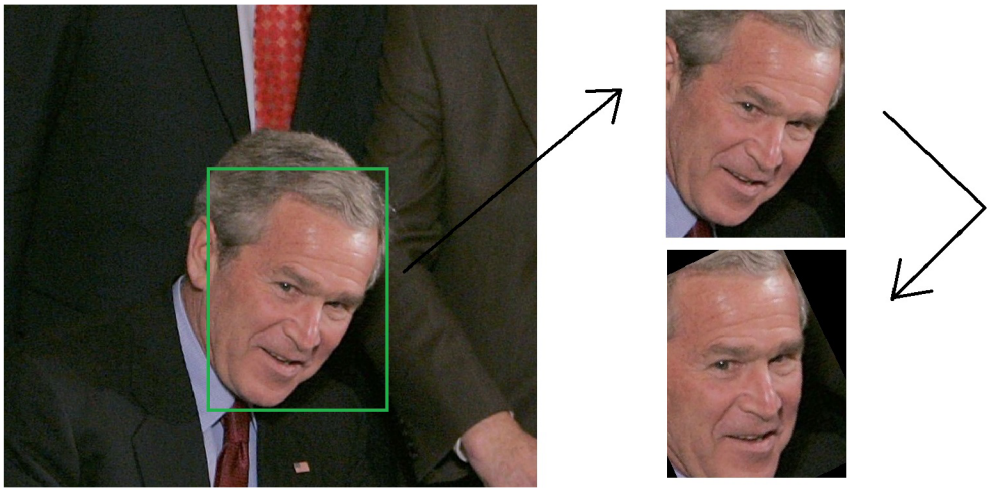
video záznam alebo kamera. Kamera je podporovaná len na operačnom systéme Linux a to z dôvodu toho, že linux umožnuje jednoduchšie odkázanie sa na kameru prostredníctvom okdazu */dev/video0*. Na čítanie vstupu sa používa knižnica OpenCV, ktorá poskytuje rozhrania pre prácu s obrazom. Pri spustení aplikácie sa vytvorí inštancia *Inference Enginu*, čo tvorí hlavný komponent knižnice OpenVINO pre prácu s modelmi. *Inference enginu* su poskytnuté inicializačné nastavenia ako sú potrebné modely, ich vstupné parametre, výstupné parametre a taktiež zariadenie, na ktorom sa inferencia daného modelu bude vykonávať. Inferencia môže byť vykonávaná na viacerych zariadeniach, to znamená že detektory, ktoré su menej náročné možú využívať CPU a rozpoznávanie tváre sa môže vykonávať na akcelerátore. Nasleduje inicializácia *Inference Enginu* a následné načítanie jednotlivých modelov. Dôležitým aspektom aplikácie je databáza osôb.

Databáza ôsob je navrhnutá ako priečinok obrázkov osôb. Databáza je načítana pri každom spustení aplikácie. Pri vytváraní databáze je využitý model pre rozpoznávanie tváre, ktorý z obrázkov tvárí zistí vektory príznakov danej osoby a tie si aplikácia spolu s označením vnútorne uloží. Databáza osôb môže obsahovať viacero osôb tej istej identity, v tomto prípade sa uložia viacere vektory príznakov. Pre zhodu tvári je použitý algoritmus hungarian a pre spočítanie zhody vektorov príznakov sa používa cosínová vzdialenosť. Každý obrázok v databáze môže mať ľubovolnú veľkosť a musí obsahovať jednu najlepšie čelne orientovanú tvár v primeranej kvalite. Túto databázu je možné vytvoriť prostredníctvom vstupného argumentu aplikácie. Tento argument spôsobí to, že pri každej detekovanej neznámej osobe na vstupných dátach sa uživateľa aplikácia opýta, či neznámu detekovaná osobu bude chcieť uložiť do databáze. Tento proces vytvorí nové okno s orezaným obrázkom danej osoby. Uživateľ bude následne vyzvaný na zadanie pomenovania tejto osoby.

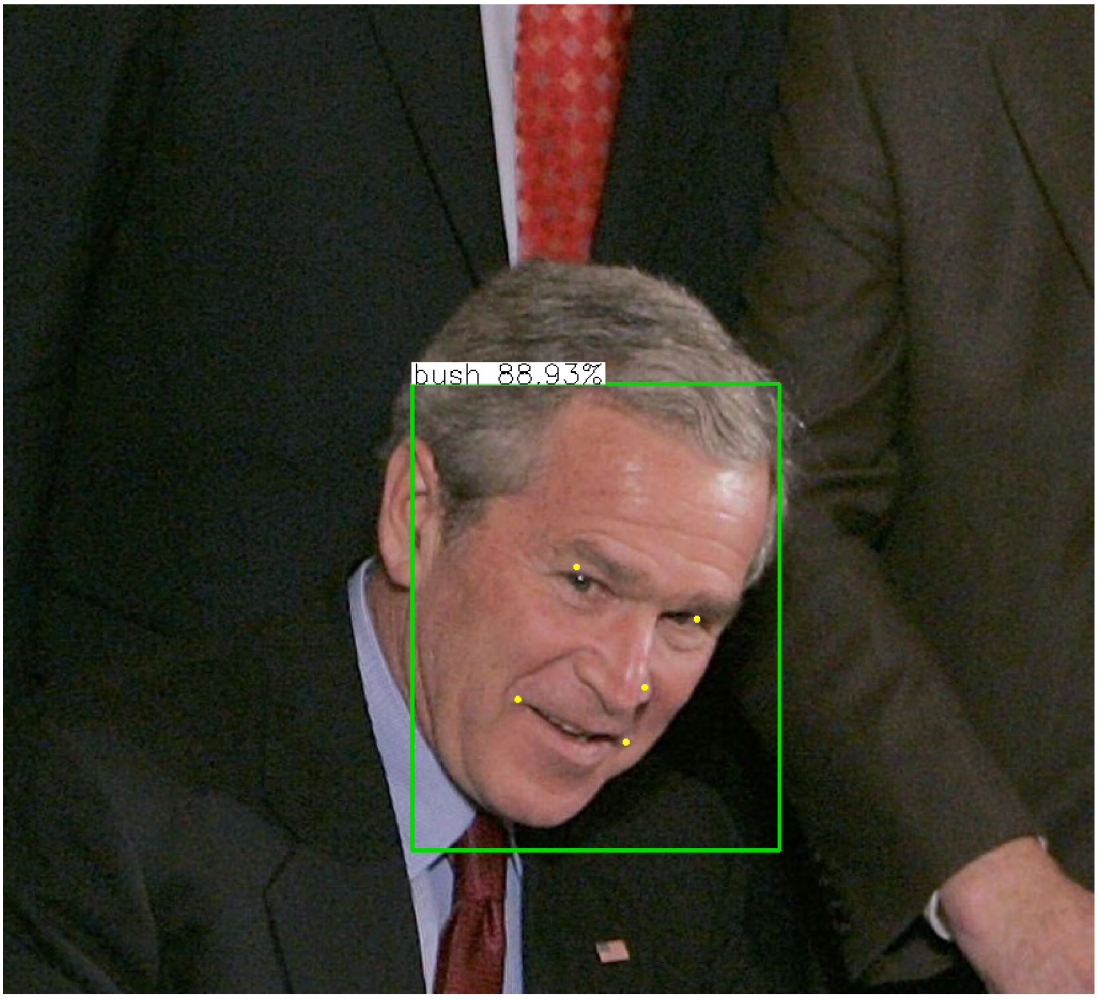
Aplikácia podporuje zasielanie detekcií ôsob na vstupných dátach prostredníctvom siete. Túto možnosť je možné povoliť vstupných argumentom. Pre posielanie týchto oznamení bola použitá knižnica ZeroMQ, ktorá je popísaná v časti 5.1.2. Oznámenia o detekcií tváre su zasielané v prípade detekcie osoby. Pri známej osoby je odoslané pomenovanie danej osoby a percentuálna zhoda. V prípade detekcie neznámej osoby sa odošle oznámenie o počte detekovaných neznámych ôsob. Ak sa osoba nachádza v dlhšom čase na vstupe, sú zasielané pravidelné oznámenia s novou percentuálnou zhodou.

Hlavnú časť aplikácie samozrejme tvorí spracovanie vstupu dát. Pri video vstupe sú spracovávané jednotlivé snímky videa. Ako prvé je na obrázok použitý detektor tvári, detektor vracia zoznam potencionálnych oblastí s tvárou takzvaných oblastí záujmu (angl. *region of interest*). Tieto oblasti sú následne poslané do detektora významných bodov tváre. Výstupom sú súradnice pre 5 významných bodov tváre pre každú oblasť. Tieto dva výstupy z detektorou sú zaslané nakoniec modelu pre rozpoznávanie tváre. Obrázok tváre sa najprv zarovná podľa referenčných bodov a získaných bodov. Na zarovnanú tvár je použitý model rozpoznávania tváre, ktorý vráti vektor príznakov a podľa neho sa zistí identita osoby. Na obrázku 5.2 je zobrazená detekcia a zarovnanie tváre, ktorá aplikácia vykonáva pred použitím modelu pre rozpoznávanie tváre.

Aplikácia umožňuje uloženie výslednej inferencie na vstupe videa poprípade obrázkov do video súboru poprípade jednotlivých obrázkov s detekciami tvárí a identitami ôsob. Podstatná časť kódu tejto aplikácie bola prevzatá z *Open Model Zoo* pre OpenVINO, čo je poskytovaný pod licenciou *Apache License 2.0*. Na obrázku 5.3 je zobrazený výstup aplikácie so zobrazenou detekciou tváre, významnými bodmi tváre a identitou detekovanej osoby. Vstup bol obrázok z ukážky 5.2.



Obr. 5.2: Ukážka detekcie a zarovnania tváre pred použitím modelu pre rozpoznávania tváre.



Obr. 5.3: Ukážka výstupu aplikácie so zobrazenou detekciou tváre, detekovanými významnými bodmi tváre a detekovanou identitou.

Kapitola 6

Experimenty

V tejto kapitole sú popísané experimenty zamerané na zhodnotenie úspešnosti rozpoznávania osôb a experimenty zamerané na zhodnotenie zrýchlenia algoritmu pomocou akcelerátra Neural Compute Stick 2. Jednotlivé experimenty s výsledkami sú bližšie popísané v nasledujúcich sekciách.

6.1 Zhodnotenie úspešnosti rozpoznávania osôb

V tejto sekcii sú popísané experimenty zamerané na zhodnotenie úspešnosti rozpoznávania osôb. Experimenty boli vykonané na troch modeloch pre rozpoznávanie tváre, FaceNet[23], ArcFace[6] a SphereFace[15]. Tieto neurónové siete sú bližšie popísané sekcii 3.6. Dátová sada, ktorá bola použitá na verifikáciu rozpoznávania ôsob, bola zvolená dátová sada *Labeled Faces In The Wild* popísaná v sekcii 3.5.1. Tento dataset je na verifikáciu vhodný z dvoch významných dôvodov. Prvým dôvodom je jeho veľmi rozsiahle využitie vo výskumných článkoch. Dataset je často využívaný ako vyhodnocovací dataset pre úlohy zamerané na rozpoznávanie ôsob podľa tváre. Druhým dôvodom je variabilita snímok v datasete. Obrázky ôsob v datasete sú zachytené v rôznorodých podmienkach, tváre sú tiež nasnímané s rôznym sklonom hlavy. Aplikácia navrhnutá v tejto práci je vnímaná ako bezpečnostná kamera a v tom prípade je dôležité aby algoritmus dokázal dobre identifikovať aj tváre, ktoré majú rôzny sklon hlavy. Pri verifikácii algoritmov bola použitá len časť datasetu, kvôli náročnosti na výpočet. Z celkového datasetu boli použité len osoby, ktoré mali v datasete viac ako 5 obrázkov. Celkovo bolo použitých 5424 obrazkov na ktorých sa nachádzalo 311 ôsob.

6.1.1 Metriky pre hodnotenie

Vyhodnotenie verifikácie algoritmov je realizované pomocou ROC (*Receiver Operating Characteristic*) kriviek[3]. ROC krivka je nástroj pre hodnotenie a optimalizáciu binárneho klasifikačného systému, ktorý zobrazuje vzťah medzi špecifickosťou a senzitivitou daného testu pre všetky prípustné hodnoty prahu vzdialenosti. Metrika vzdialenosti medzi 2 vektormi príznakov bola vypočítaná pomocou kosínusovej vzdialenosti. Senzitivita je mierou pravdepodobnosti, že skutočný pozitívny prípad bude klasifikovaný ako pozitívny. Senzitivita (angl. *True Positive Rate*) je definované ako:

$$TPR = Senzitivita = \frac{TP}{TP + FN} \tag{6.1}$$

Špecifickosť (angl. *Specificity*) sa pri ROC krivkách počítá ako 1 − *Specificity* čož vyadruje FPR (*False Positive Rate*). Miera FPR je v podstate mierou toho, ako často sa vyskytne „falošný poplach“ – alebo ako často bude skutočný negatívny prípad klasifikovaný ako pozitívny. FPR je definovaná následovne:

$$FPR = 1 - Specificity = \frac{FP}{FP + TN} \tag{6.2}$$

Skratky v rovniciach vyjadrujú následovné hodnoty:
True positives (TP) - pozitívne prvky, ktoré boli predikované ako pozitívne
False positives (FP) - negatívne prvky, ktoré boli predikované ako pozitívne
True negatives (TN) - negatívne prvky, ktoré boli predikované ako negatívne
False negatives (FN) - pozitívne prvky, ktoré boli predikované ako negatívne

Na obrázku 6.1 je zobrazený graf ROC krivky s hodnoteniami algoritmov pre rozpoznávanie tváre. Pri každom algoritme je zobrazená hodnota AUC (angl. *Area Under Curve*). Na druhom grafe je zobrazený detail hornej časti ROC krivky.

Pri verifikácii algoritmov pre rozpoznávanie tváre bolo taktiež spočítané F1 skóre a presnosť algoritmu. Na základe týchto metrík bola zvolená hodnota prahu vzdialenosti pre aplikáciu. *Precision* a *Recall* sú dve najbežnejšie metriky, ktoré zohľadňujú nerovnováhu triedy. Sú tiež základom skóre F1.

Metrika *Precision* vyjadruje, koľko prvkov z celej množiny pozitívnych prvkov bolo predikovaných pozitívne. Táto metrika je definovaná následovne:

$$Precision = \frac{TP}{TP + FP} \tag{6.3}$$

Metrika *Recall* vyjadruje, koľko prvkov z celej množiny prvkov predikovaných pozitívne je skutočne pozitívnych. Táto metrika je definovaná následovne:

$$Recall = \frac{TP}{TP + FN} \tag{6.4}$$

Spojením týchto dvoch vyššie uvedených metrík vzniká F1 skóre, ktoré je definované ako ich harmonický priemer. F1 skóre je definované nasledovne:

$$F1\ score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{6.5}$$

Na obrázku 6.2 je zobrazený graf na ktorom sú zobrazené F1 hodnoty a presnosti modelu FaceNet pre rôzne hodnoty prahu vzdialenosti. Bod dotyku týchto dvoch metrik v určitej hodnote prahu vzdialenosti značí, že toto je optimálna hodnota prahu vzdialenosti. V tomto prípade je optimálna hodnota prahu 0,71.

6.2 Vyhodnotenie zrýchlenia algoritmu pomocou akcelerátoru

V experimentoch som sa zameral na vyhodnotenia zrýchlenia pomocou akcelerátoru použítého spolu so vstavaným systémom. Akcelerátor bol použitý aj na modernom notebooku pre porovnanie výsledkou dosiahnutých na *single-board* počítači popísanom v sekcii 5.1.2.

Notebook obsahuje procesor od spoločnosti Intel 8. generácie (Intel i5 - 8265U), v čase experimentoch bolo v notebooku nainštalovaných 8GB RAM pamäte. Ako vstupné dáta na experimentovanie boli použité prednahraté video úseky. V prvom videu úseku sa nachádzala iba jedna osoba. V druhom video úseku bolo súčasne 7 ôsob. Video úseky boli pôvodne nahraté v 4K rozlíšení (3840 × 2160) a z nich boli vytvorené ďalšie videá v nižších rozlíšeniach, videá boli snímané v rýchlosti 30 snímkov za sekundu. V experimentoch bola merená odozva medzi jednotlivými spracovaniami obrázkov a taktiež odozva medzi jednotlivých časťami spracovania jedného snímku. Prvá časť spracovania zahŕňa načítanie jedného snímku z video úseku do aplikácie a pripravenie snímku ku spracovaniu. Druhá časť spracovania zahŕňa prípravu snímku na inferenciu, čož je orezanie a zarovnanie snímku a následné zaslanie snímku do zariadenia na ktorom bude prebiehať inferencia. Posledná a tretia časť spracovania zahŕňa inferenciu a získanie výstupov zo zariadenia, na ktorom prebiehala inferencia. Model pre rozpoznávanie tvare bol použitý Facenet popísaný v časti 3.10. Časy spracovania jednotlivých obrázkov videa boli spriemerované. Výstupom experimentov je graf a tabuľky s nameranými hodnotami pre oba video úseky. Prvý stĺpec v tabuľkách vyjadruje v akej kvalite bolo video pri experimentoch a nasledujúce stĺpce sú namerané hodnoty pre celkové spracovanie jedného snímku, prvú časť spracovania, druhú časť spracovania a tretiu časť spracovania. Jednotlivé časti sú popísané vyššie v tomto odstavci.

Ako prvé bolo experimentované s videom, kde sa náchadza iba jedna osoba. Výsledkom je graf, ktorý je zobrazený na obrázku 6.3 a 4 tabuľky (6.1, 6.2, 6.3, 6.4) s nameranými hodnotami pre jednotlivé zariadenie. Z tabuliek je viditeľné, že použitím akcelerátora Neural Compute Stick 2 sa podarilo trojnásobne zrýchliť čas inferencie. Priemerný čas inferencie na počítači UPboard pri kvalite videa 1280 × 720 bez použitia akcelerátora bol 150,7 ms, s použitím akcelerátora sa čas inferencie zlepšil na hodnotu 47,4 ms.

V druhom experimente bolo experimentované s videom, kde sa naraz náchadzalo sedem ôsob, čím proces spracovania a inferencie bol podstatnejšie dlhší. Výsledkom je graf, ktorý je zobrazený na obrázku 6.4 a 4 tabuľky (6.5, 6.6, 6.7, 6.8) s nameranými hodnotami pre jednotlivé zariadenie. Z tabuliek je vidieť, že akcelerátor Neural Compute Stick 2 zlepšil čas inferencie niekoľko násobne pri výskyte viacerych osôb naraz na jednom obrázku. Priemerný čas inferencie na počítači UPboard pri kvalite videa 1280 × 720 bez použitia akcelerátora bol až 972 ms, s použitím akcelerátora sa čas inferencie zlepšil na hodnotu 140,7 ms. Toto zrýchlenie je možné, kvôli architektúre procesora Intel Movidius Myriad X, ktorý poskytuje 16 vektorových procesorov čo umožňuje súčasne spúšťať viacero inferenčných kanálov.

Video s jednou osobou - Intel i5				
	Spracovanie obrázka	Načítanie obrázka	Preprocessing	Inferencia
HD	26	3,1	2,9	19
FULL HD	37	6,7	5	22,7
4K	82	29,2	14	29,2

Tabuľka 6.1: Tabuľka s hodnotami nameranými pri video úseku, na ktorom sa nachádzala len 1 osoba. Hodnoty boli namerané na notebooku Intel i5. Všetky hodnoty sú uvedené v milisekundách.

Video s jednou osobou - Intel i5 + NCS2				
	Spracovanie obrázka	Načítanie obrázka	Preprocessing	Inferencia
HD	64	4,8	6,9	45,9
FULL HD	74	13,4	8,5	45,3
4K	95	28,3	12,9	41,2

Tabuľka 6.2: Tabuľka s hodnotami nameranými pri video úseku, na ktorom sa nachádzala len 1 osoba. Hodnoty boli namerané na notebooku Intel i5 spolu s akcelerátorom Neural Compute Stick 2. Všetky hodnoty sú uvedené v milisekundách.

Video s jednou osobou - UPboard Intel Atom				
	Spracovanie obrázka	Načítanie obrázka	Preprocessing	Inferencia
HD	201	13,9	13,1	150,7
FULL HD	220	28,1	20,9	154,7
4K	387	119,6	62,2	163,3

Tabuľka 6.3: Tabuľka s hodnotami nameranými pri video úseku, na ktorom sa nachádzala len 1 osoba. Hodnoty boli namerané na počítači UPboard. Všetky hodnoty sú uvedené v milisekundách.

Video s jednou osobou - UPboard Intel Atom + NCS2				
	Spracovanie obrázka	Načítanie obrázka	Preprocessing	Inferencia
HD	100	24,3	19,1	47,4
FULL HD	120	45,2	22,1	47,7
4K	219	115,5	58,5	42,9

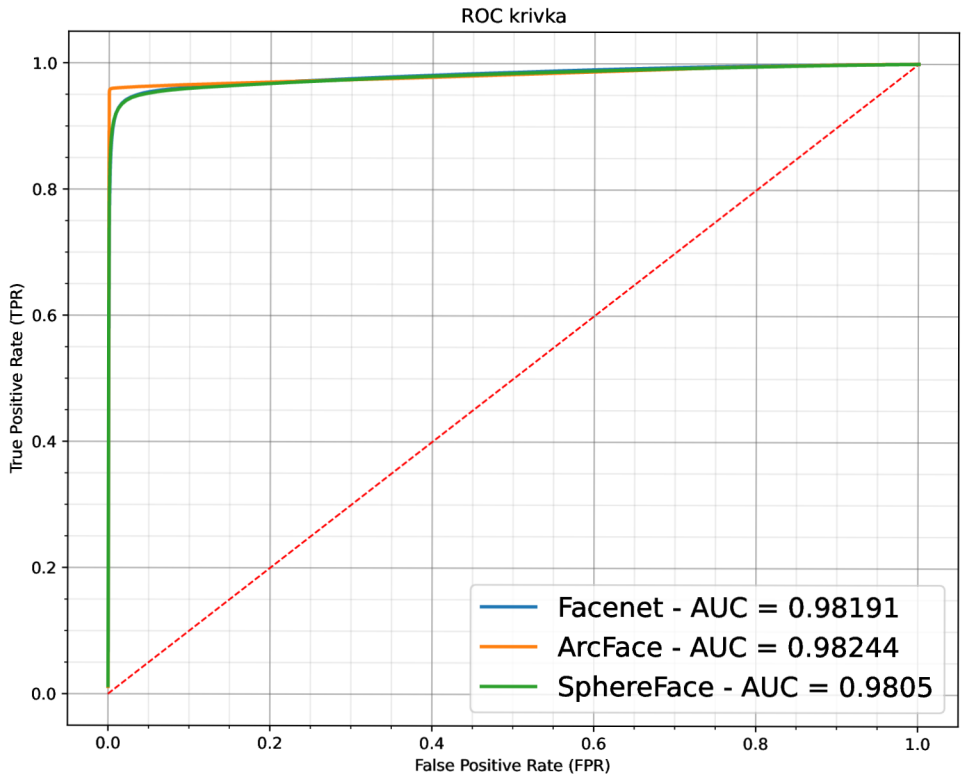
Tabuľka 6.4: Tabuľka s hodnotami nameranými pri video úseku, na ktorom sa nachádzala len 1 osoba. Hodnoty boli namerané na počítači UPboard spolu s akcelerátorom Neural Compute Stick 2. Všetky hodnoty sú uvedené v milisekundách.

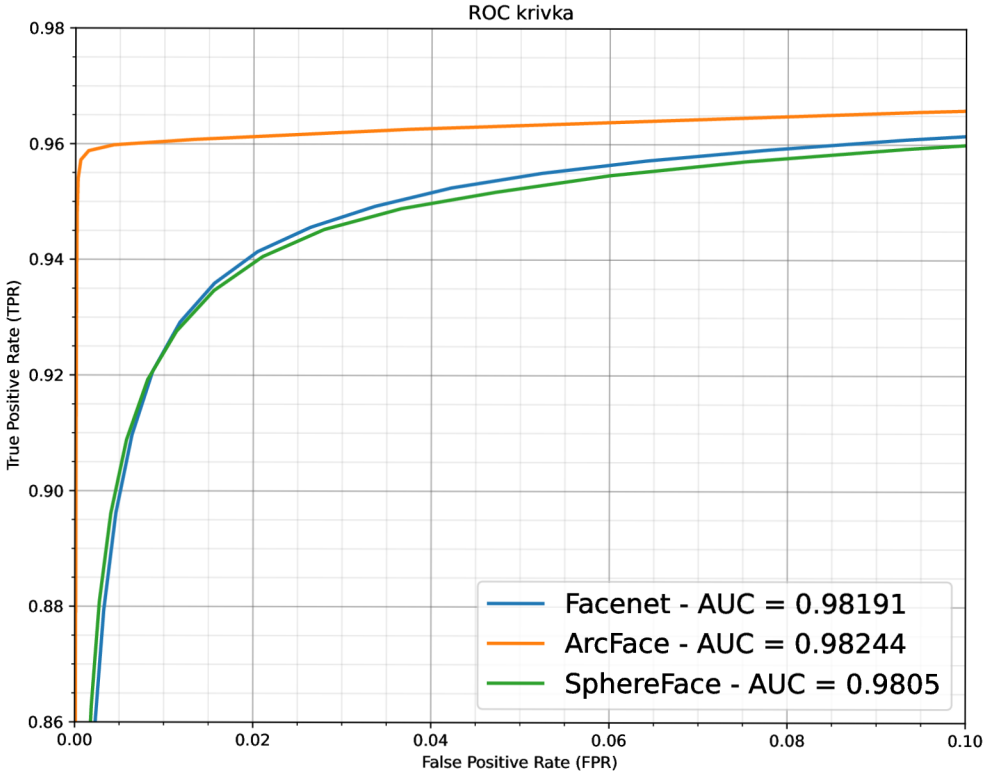
Video so 7 osobami - Intel i5				
	Spracovanie obrázka	Načítanie obrázka	Preprocessing	Inferencia
HD	118	2,7	6,8	106,9
FULL HD	134	6,4	10,2	107,3
4K	164	27,2	19,8	108,2

Tabuľka 6.5: Tabuľka s hodnotami nameranými pri video úseku, na ktorom sa nachádzalo súčasne 7 tvárí. Hodnoty boli namerané na notebooku Intel i5. Všetky hodnoty sú uvedené v milisekundách.

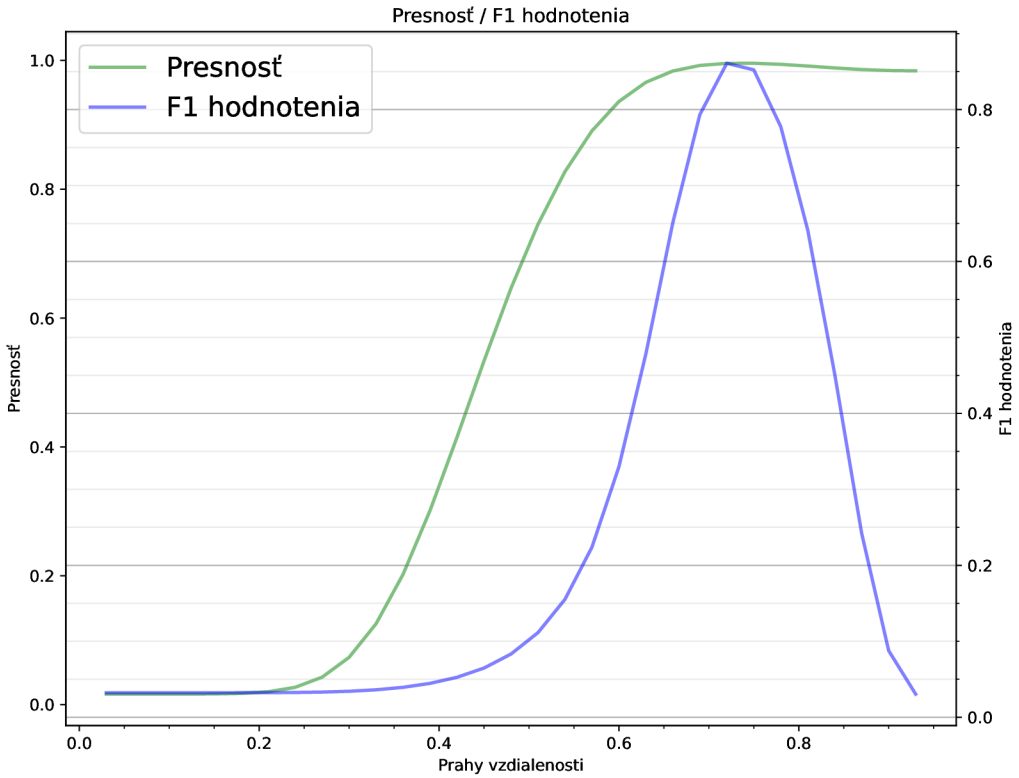
Video so 7 osobami - Intel i5 + NCS2				
	Spracovanie obrázka	Načítanie obrázka	Preprocessing	Inferencia
HD	164	6,6	17,9	134,4
FULL HD	177	16	21,4	134
4K	199	28,9	37,1	120

Tabuľka 6.6: Tabuľka s hodnotami nameranými pri video úseku, na ktorom sa nachádzalo súčasne 7 tvárí. Hodnoty boli namerané na notebooku Intel i5 spolu s akcelerátorom Neural Compute Stick 2. Všetky hodnoty sú uvedené v milisekundách.





Obr. 6.1: ROC krivka zobrazujúca hodnotenie algoritmov popísaných v sekcii 3.6. Druhý graf je detail hornej časti ROC krivky.



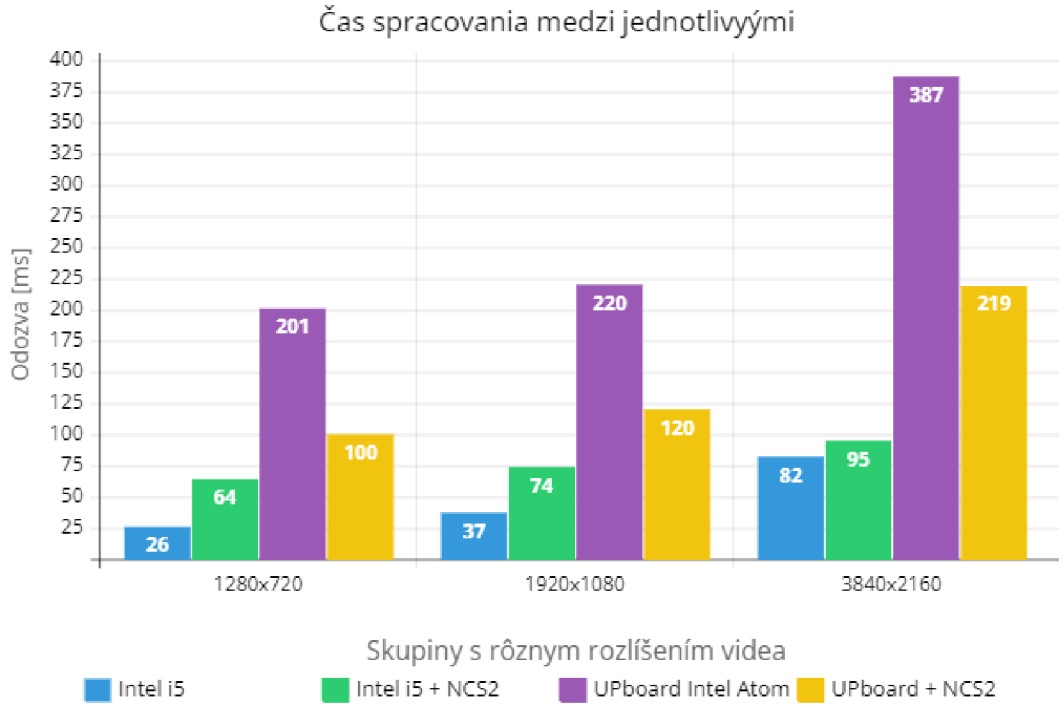
Obr. 6.2: Na grafe je zobrazené F1 skóre a presnosť modelu FaceNet pre rôzne hodnoty prahu vzdialenosti. Optimálna hodnota prahu vzdialenosti je 0,71.

Video so 7 osobami - UPboard Intel Atom				
	Spracovanie obrázka	Načítanie obrázka	Preprocessing	Inferencia
HD	1040	13	32,5	972
FULL HD	1107	27,2	44,72	975
4K	1233	119,1	74	990

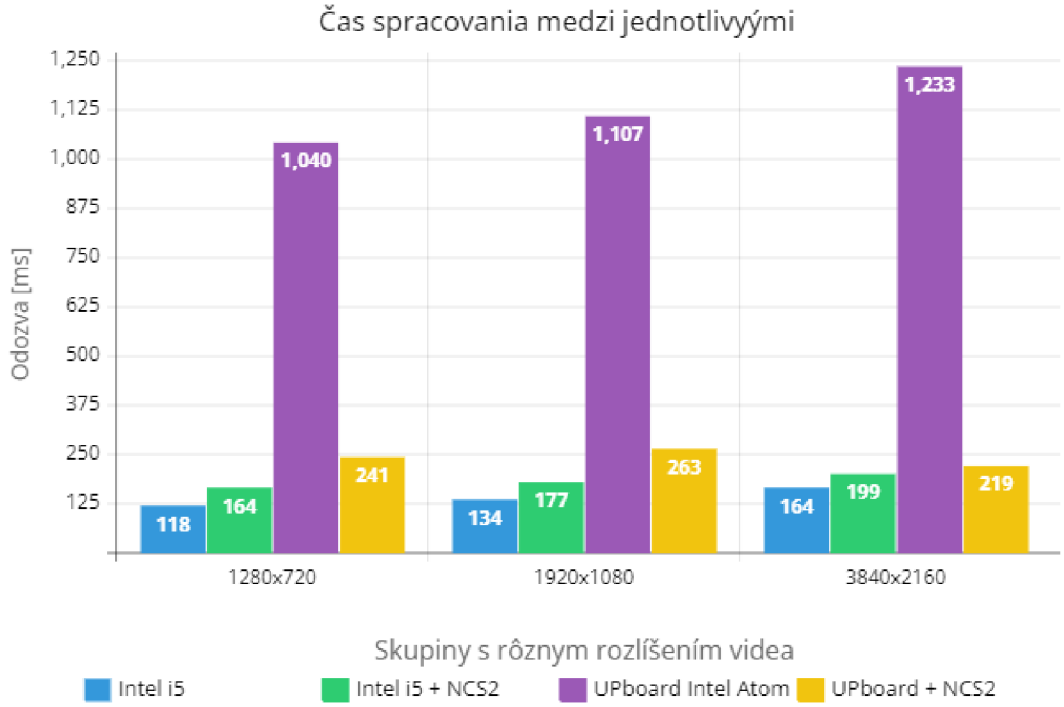
Tabuľka 6.7: Tabuľka s hodnotami nameranými pri video úseku, na ktorom sa nachádzalo súčasne 7 tvárí. Hodnoty boli namerané na počítači UPboard. Všetky hodnoty sú uvedené v milisekundách.

Video so 7 osobami - UPboard Intel Atom + NCS2				
	Spracovanie obrázka	Načítanie obrázka	Preprocessing	Inferencia
HD	241	25,4	66,2	140,7
FULL HD	263	45,2	67	140,6
4K	369	118,5	98,5	126,2

Tabuľka 6.8: Tabuľka s hodnotami nameranými pri video úseku, na ktorom sa nachádzalo súčasne 7 tvárí. Hodnoty boli namerané na počítači UPboard spolu s akcelerátorom Neural Compute Stick 2. Všetky hodnoty sú uvedené v milisekundách.



Obr. 6.3: Graf znázorňujúci priemerný čas spracovania a inferencie medzi jednotlivými obrázkami videa vykonaný na rôznych zariadeniach. Prvý stĺpec v skupine zariadení je notebook s procesorom i5 (modrý), druhý stĺpec je notebook s procesorom i5 spolu s akcelerátorom Neural Compute Stick 2, na ktorom bola vykonávaná inferencia (zelený), tretí stĺpec je vstavané zariadenie popísané v časti 5.1.2 (fialový) a posledný stĺpec je vstavané zariadenia spolu akcelerátorom Neural Compute Stick 2 (žltý). Výška stĺpcov odpovedá priemernému času spracovania a inferencie jediného obrázku vo videu. V každom stĺpci je zobrazená presná hodnota, ktorá odpovedá priemernému času spracovania a inferencie na danom zariadení. Video bolo použité v rôznych rozlíšeniach. Video obsahovalo jednu osobu.



Obr. 6.4: Graf znázorňujúci priemerný čas spracovania a inferencie medzi jednotlivými obrázkami videa vykonaný na rôznych zariadeniach. Prvý stĺpec v skupine zariadení je notebook s procesorom i5 (modrý), druhý stĺpec je notebook s procesorom i5 spolu s akcelerátorom Neural Compute Stick 2, na ktorom bola vykonávaná inferencia (zelený), tretí stĺpec je vstavané zariadenie popísané v časti 5.1.2 (fialový) a posledný stľpec je vstavané zariadenia spolu akcelerátorom Neural Compute Stick 2 (žltý). Výška stĺpcov odpovedá priemernému času spracovania a inferencie jediného obrázku vo videu. V každom stĺpci je zobrazená presná hodnota, ktorá odpovedá priemernému času spracovania a inferencie na danom zariadení. Video bolo použité v rôznych rozlíšeniach. Video obsahovalo naraz sedem ôsob.

Kapitola 7

Záver

Práca uviedla čitateľa do problematiky strojového učenia a hlbokých neurónových sietí, vysvetlila základné pojmy a koncepty. Bližšie sa zamerala na konvolučné siete, ktoré boli využívané v praktickej časti. Sú taktiež zmienené a popísané aktuálne najnovšie a najlepšie prístupy pre rozpoznávanie tváre, ktoré dosahujú state-of-the-art výsledkov. Na konci teoretickej časti bol popísaný edge computing a k nemu uvedené charakteristiky a nakoniec príklady edge computingu v reálnom svete.

V praktickej časti práce bolo treba navrhnúť vstavaný systém na rozpoznávanie tváre, ktorý sa skladá z kamery, výpočetnej jednotky a akcelerátoru Neural Compute Stick 2. Algoritmus na rozpoznávanie tváre mohol byť zvolený voľne dostupný a predtrénovaný. Neural Compute Stick 2 bol teoretický spracovaný a taktiež platforma, ktorá umožnuje prácu s týmto akcelerátorom, OpenVINO. Boli použité tri rôzne modely na rozpoznávanie tváre, model pre detekciu tváre v obraze, model pre detekciu významných bodov tváre a model pre rozpoznávanie tváre. Aplikácia umožnuje použitie troch rôzných modelov pre rozpoznávanie tváre: Facenet, Sphereface a Arcface, tieto modely sú popísané v sekcii 3.6. V aplikácii bol implementovaný server pre zasielanie notifikácii o detekciách ôsob na vstupe. Súčasťou riešenie je aj demonstračný klient, ktorý číta prichádzajúce notifikácie.

Experimenty ukázali, že použitie akcelerátora spolu so vstavaným systémom vie zlepšiť spracovanie a inferenciu snímkov a tým odľahčiť zaťaženie procesora. V experimentoch sa ukázala sila paralérneho spracovania obrázkov akcelerátorom, ktorý zrýchlil inferenciu a spracovanie niekoľko násobne, keď sa na jednom obrázku nachádzalo viacero ôsob. V budúcnosti by bolo vhodné aplikácii navrhnúť uživatelské rozhranie a taktiež zlepšiť prácu s vytváraním databáze. Bolo by dobré vyskúšať zlepšiť spracovanie obrazu v aplikácii, čo by mohlo viesť ešte k lepším výsledkom. Pri experimentoch bolo taktiež zistené, že spomaľovať inferenciu može prenos dát do akcelerátora cez USB, toto je možné vyriešiť použitím akcelerátora s pripojením PCIe.

Literatúra

[1] Agarap, A. F. Deep Learning using Rectified Linear Units (ReLU). *CoRR*. 2018, abs/1803.08375. Dostupné z: http://arxiv.org/abs/1803.08375.

[2] Bhande, A. *What is underfitting and overfitting in machine learning and how to deal with it.* [online]. GreyAtom School, marec 2018 [cit. 2022-01-11]. Dostupné z: https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76.

[3] Bradley, A. P. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*. 1997, zv. 30, č. 7, s. 1145–1159. DOI: https://doi.org/10.1016/S0031-3203(96)00142-2. ISSN 0031-3203. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0031320396001422.

[4] Cao, K., Liu, Y., Meng, G. a Sun, Q. An Overview on Edge Computing Research. *IEEE Access*. 2020, zv. 8, s. 85714–85728. DOI: 10.1109/ACCESS.2020.2991734.

[5] Cao, Q., Shen, L., Xie, W., Parkhi, O. M. a Zisserman, A. VGGFace2: A dataset for recognising faces across pose and age. *CoRR*. 2017, abs/1710.08092. Dostupné z: http://arxiv.org/abs/1710.08092.

[6] Deng, J., Guo, J. a Zafeiriou, S. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. *CoRR*. 2018, abs/1801.07698. Dostupné z: http://arxiv.org/abs/1801.07698.

[7] Fei Fei Li, R. K. a Xu, D. *CS231n Convolutional Neural Networks for Visual Recognition* [online]. [cit. 2022-01-10]. Dostupné z: https://cs231n.github.io/neural-networks-1/.

[8] Glorot, X. a Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. Chia Laguna Resort, Sardinia, Italy: PMLR. 13–15 May 2010, zv. 9, s. 249–256. Proceedings of Machine Learning Research. Dostupné z: https://proceedings.mlr.press/v9/glorot10a.html.

[9] Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A. et al. Recent advances in convolutional neural networks. *Pattern Recognition*. 2018, zv. 77, s. 354–377. DOI: https://doi.org/10.1016/j.patcog.2017.10.013. ISSN 0031-3203. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0031320317304120.

[10] He, K., Zhang, X., Ren, S. a Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *CoRR*. 2015, abs/1502.01852. Dostupné z: http://arxiv.org/abs/1502.01852.

[11] Huang, G. B., Mattar, M., Lee, H. a Learned Miller, E. *Learning to Align from Scratch* [online]. 2012 [cit. 2019-5-01]. Dostupné z: http://vis-www.cs.umass.edu/lfw/.

[12] Ioffe, S. a Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *CoRR*. 2015, abs/1502.03167. Dostupné z: http://arxiv.org/abs/1502.03167.

[13] Jain, A., Mao, J. a Mohiuddin, K. Artificial neural networks: a tutorial. *Computer*. 1996, zv. 29, č. 3, s. 31–44. DOI: 10.1109/2.485891.

[14] Larochelle, H., Bengio, Y., Louradour, J. a Lamblin, P. Exploring Strategies for Training Deep Neural Networks. *Journal of Machine Learning Research*. 2009, zv. 10, č. 1, s. 1–40. Dostupné z: http://jmlr.org/papers/v10/larochelle09a.html.

[15] Liu, W., Wen, Y., Yu, Z., Li, M., Raj, B. et al. SphereFace: Deep Hypersphere Embedding for Face Recognition. *CoRR*. 2017, abs/1704.08063. Dostupné z: http://arxiv.org/abs/1704.08063.

[16] Liu, Z., Luo, P., Wang, X. a Tang, X. Deep Learning Face Attributes in the Wild. December 2015.

[17] Moghaddam, B., Wahid, W. a Pentland, A. Beyond eigenfaces: probabilistic matching for face recognition. 1998, s. 30–35. DOI: 10.1109/AFGR.1998.670921.

[18] O'Shea, K. a Nash, R. An Introduction to Convolutional Neural Networks. *CoRR*. 2015, abs/1511.08458. Dostupné z: http://arxiv.org/abs/1511.08458.

[19] Premsankar, G., Di Francesco, M. a Taleb, T. Edge Computing for the Internet of Things: A Case Study. *IEEE Internet of Things Journal*. 2018, zv. 5, č. 2, s. 1275–1284. DOI: 10.1109/JIOT.2018.2805263.

[20] Ruder, S. An overview of gradient descent optimization algorithms. *CoRR*. 2016, abs/1609.04747. Dostupné z: http://arxiv.org/abs/1609.04747.

[21] Rumelhart, D. E., Hinton, G. E. a Williams, R. J. Learning representations by back-propagating errors. *Nature*. Nature Publishing Group. 1986, zv. 323, č. 6088, s. 533–536.

[22] Saez-Trigueros, D., Meng, L. a Hartnett, M. Face Recognition: From Traditional to Deep Learning Methods. *CoRR*. 2018, abs/1811.00116. Dostupné z: http://arxiv.org/abs/1811.00116.

[23] Schroff, F., Kalenichenko, D. a Philbin, J. Facenet: A unified embedding for face recognition and clustering. 2015, s. 815–823.

[24] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. a Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*. 2014, zv. 15, č. 56, s. 1929–1958. Dostupné z: http://jmlr.org/papers/v15/srivastava14a.html.

[25] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E. et al. Going Deeper with Convolutions. *CoRR*. 2014, abs/1409.4842. Dostupné z: http://arxiv.org/abs/1409.4842.

[26] Traore, B. B., Kamsu Foguem, B. a Tangara, F. Deep convolution neural network for image recognition. *Ecological Informatics*. 2018, zv. 48, s. 257–268. DOI: https://doi.org/10.1016/j.ecoinf.2018.10.002. ISSN 1574-9541. Dostupné z: https://www.sciencedirect.com/science/article/pii/S1574954118302140.

[27] Turk, M. a Pentland, A. Eigenfaces for Recognition. *Journal of Cognitive Neuroscience*. Január 1991, zv. 3, č. 1, s. 71–86. DOI: 10.1162/jocn.1991.3.1.71. ISSN 0898-929X. Dostupné z: https://doi.org/10.1162/jocn.1991.3.1.71.

[28] Viola, P. a Jones, M. Rapid object detection using a boosted cascade of simple features. 2001, zv. 1, s. I–I. DOI: 10.1109/CVPR.2001.990517.

[29] Wang, M. a Deng, W. Deep face recognition: A survey. *Neurocomputing*. Elsevier. 2021, zv. 429, s. 215–244. DOI: https://doi.org/10.1016/j.neucom.2020.10.081. ISSN 0925-2312. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0925231220316945.

[30] Wolf, L., Hassner, T. a Maoz, I. Face recognition in unconstrained videos with matched background similarity. 2011, s. 529–534. DOI: 10.1109/CVPR.2011.5995566.

[31] Wong, W. G. *Intel's Myriad X Vision Chip Incorporates Neural Network* [online]. 2017 [cit. 2022-4-15]. Dostupné z: https://www.electronicdesign.com/industrial-automation/article/21805511/intels-myriad-x-vision-chip-incorporates-neural-network.

[32] Xuehai Hong, Y. W. Edge Computing Technology: Development and Countermeasures. *Strategic Study of Chinese Academy of Engineering*. Strategic Study of Chinese Academy of Engineering. 2018, zv. 20, č. 2, s. 20. DOI: 10.15302/J-SSCAE-2018.02.004. Dostupné z: https://journal.hep.com.cn/sscae/EN/abstract/article_23127.shtml.

[33] Yi, D., Lei, Z., Liao, S. a Li, S. Z. Learning Face Representation from Scratch. *CoRR*. 2014, abs/1411.7923. Dostupné z: http://arxiv.org/abs/1411.7923.

[34] Zeiler, M. D. a Fergus, R. Visualizing and Understanding Convolutional Networks. *CoRR*. 2013, abs/1311.2901. Dostupné z: http://arxiv.org/abs/1311.2901.

[35] Zhang, K., Zhang, Z., Li, Z. a Qiao, Y. Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks. *CoRR*. 2016, abs/1604.02878. Dostupné z: http://arxiv.org/abs/1604.02878.

Príloha A

Obsah priloženého pamäťového média

README.md – Obsahuje návod a popis aplikácie
/doc – Zdrojové súbory technickej správy
xhorni20.pdf – Práca v PDF
/src – Adresár so zdrojovými súbormi a dátami
    /demo_data – Demonstračné dáta k použitiu spolu s aplikáciou
    /face_database - Predvytvorená databáza k použitiu spolu s demonstračnými dátami
    /models - Predtrénované modely k aplikácii