VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

AKCELERACE ALGORITMU PRO ROZPOZNÁVÁNÍ OBLIČEJE POMOCÍ NEURAL COMPUTE STICK 2

ACCELERATION OF FACE RECOGNITION ALGORITHM WITH NEURAL COMPUTE STICK 2

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **EVA MIČÁNKOVÁ**
AUTHOR

VEDOUCÍ PRÁCE **Ing. TOMÁŠ GOLDMANN**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání bakalářské práce

141562

Ústav: Ústav inteligentních systémů (UITS)
Studentka: **Mičánková Eva**
Program: Informační technologie
Specializace: Informační technologie
Název: **Akcelerace algoritmu pro rozpoznávání obličeje pomocí Neural Compute Stick 2**
Kategorie: Vestavěné systémy
Akademický rok: 2022/23

Zadání:

1. Seznamte se s problematikou rozpoznávání osob podle obličeje. Sumarizujte informace o dostupných algoritmech pro rozpoznávání osob podle obličeje založených na neuronových sítích.
2. Seznamte se s knihovnou OpenVINO a akcelerační platformou pro neuronové sítě Intel Neural Compute Stick 2 a zjistěte, jaké neuronové sítě lze pomocí této platformy akcelerovat.
3. Navrhněte aplikaci, která bude provádět rozpoznávání osob v obraze pomocí alespoň 2 dostupných algoritmů. Předpokládejte, že databáze osob bude uložená na zařízení, kde poběží vaše aplikace.
4. Navržené řešení implementujte tak, aby výpočetní část algoritmu mohla běžet na CPU, GPU a Intel Neural Compute Stick 2.
5. Proveďte experimenty zaměřené na zhodnocení rychlosti rozpoznávání na jednotlivých platformách.

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.

Při obhajobě semestrální části projektu 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: 1.11.2022
Termín pro odevzdání: 10.5.2023
Datum schválení: 3.11.2022

Abstrakt

Tato práce je zaměřena na problematiku rozpoznávání obličejů v obraze pomocí neuronových sítí a jejich akceleraci. Obsahuje souhrn dříve používaných technik a zabývá se využitím dnes dominujících konvolučních neuronových sítí pro řešení této problematiky. Práce se také zaměřuje na mechanismy akcelerace, které lze v této oblasti použít. Na základě znalostí problematiky získaných studiem byl vytvořen systém na konceptu *edge computingu*, který může být použit jako domácí bezpečnostní systém připojený k IP kameře, který zasílá upozornění o přítomnosti neznámé osoby ve střežené oblasti.

Abstract

This thesis focuses on the issue of facial recognition in a face image using neural networks and its acceleration. It provides an overview of previously used techniques and addresses the use of currently dominant convolutional neural networks to solve this issue. The work also focuses on acceleration mechanisms that can be used in this area. Based on the knowledge of the issue, a system based on the concept of edge computing was created, which can be used as a home security system connected to an IP camera, which sends a notification about the presence of an unknown person in a guarded area.

Klíčová slova

Rozpoznávání obličeje, akcelerace, Neural Compute Stick 2, OpenVINO, neuron, neuronové sítě, konvoluční neuronové sítě, datasety, ArcFace, SphereFace, FaceNet, edge computing, RTSP stream, IP kamera.

Keywords

Face racognition, acceleration, Neural Compute Stick 2, OpenVINO, neuron, neural networks, convolutional neural networks, datasets, ArcFace, SphereFace, FaceNet, edge computing, RTSP stream, IP camera.

Citace

MIČÁNKOVÁ, Eva. *Akcelerace algoritmu pro rozpoznávání obličeje pomocí Neural Compute Stick 2.* Brno, 2023. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Tomáš Goldmann

Akcelerace algoritmu pro rozpoznávání obličeje pomocí Neural Compute Stick 2

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně pod vedením pana Ing. Tomáše Goldmanna. Uvedla jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpala.

.....
Eva Mičánková
8. května 2023

Poděkování

V této sekci bych chtěla velmi poděkovat svému vedoucímu Ing. Tomášovi Goldmannovi za skvělé vedení, rady, ochotu, čas a nasměrování vypracování.

Obsah

Kapitola 1

Úvod

Systémy pro rozpoznávání tváří jsou systémy schopné identifikovat, či verifikovat osobu podle obličeje z fotografií, nebo videa na základě vedené databáze obličejů. Jsou úzce spjaty se systémy detekce obličeje, které předcházejí samotnému rozpoznávání a jejich cílem je detekovat obličej. Takový systém je neinvazivní a probíhá přirozeně prostřednictvím kamerového systému, aniž by omezoval jednotlivé subjekty. Tyto systémy nacházejí využití primárně u bezpečnostních systémů, kde se využívají pro identifikaci, verifikaci osob, nebo sběr statistických dat a u mobilních technologií, kde slouží jako forma autentizace. Bezpečnost a identifikace/verifikace je dnes velké téma, ať už v malém měřítku – domácnosti, nebo velkém – firmy, letiště, státní instituce apod. Systémy jsou postaveny na konceptu *edge computingu*, kdy veškeré zpracování dat probíhá většinou na nízkospotřebových a méně výkonných zařízeních na okraji sítě. S rostoucími požadavky na *real-time* odezvu se systém stává výpočetně náročnější a komplexnější a pro zachování co nejvyšší rychlosti a přesnosti je jej potřeba optimalizovat a akcelerovat.

Práce se zajímá o vývoj metod rozpoznávání obličejů z obrazu až po dnes velmi používanou metodu hlubokých neuronových sítí. Popisuje metriky vyhodnocení výsledků rozpoznávání a metody binární klasifikace. Zaměřuje se na architekturu konvolučních neuronových sítí, které spadají pod hluboké neuronové sítě, jež je velmi používaná metoda v této oblasti počítačového vidění. Sumarizuje algoritmy používané pro detekci a rozpoznávání obličejů, včetně datasetů, které byly použity pro trénování, testování a validaci. Věnuje se technologiím, které se používají pro vestavěné systémy, které zahrnují *edge computing* a IP kamery. Také sumarizuje metody akcelerace včetně hardwarového akcelerátoru Intel Neural Compute Stick 2, jeho architekturu, schopnosti a sadu nástrojů OpenVINO.

Dále se práce zabývá návrhem a implementací vestavěného systému, který je postaven na konceptu *edge computingu*. Tento systém může být využit jako domácí bezpečnostní systém, který disponuje funkcí zasílání notifikací, která může být využita pro upozornění uživatele na přítomnost neznámé osoby ve střežené oblasti. Práce popisuje návrh a způsob implementace nejen výpočetní části, ale také uživatelského rozhraní. Poslední část práce je věnována experimentům, které porovnávají přesnost použitých modelů pro rozpoznávání obličejů a schopnosti akcelerátoru *Intel Neural Compute Stick 2* s platformami CPU a GPU. Cílem experimentů bylo zjistit, jakých výsledků dosahuje akcelerace systému pomocí CPU, GPU a NCS2, jaká konfigurace systému je nejvhodnější pro jednotlivé modely a která konfigurace dosahuje nejvyšší snímkové frekvence a nejnižší latence mezi porovnávanými modely.

Práce je strukturována do pěti kapitol. Kapitola 2 je věnována popisu metod reprezentace obličeje, které se dříve používaly pro rozpoznávání obličejů 2.1, používaným metrikám

pro vyhodnocení 2.1.5 a validaci binárních klasifikátorů 2.1.6. Dále se věnuje nejpoužívanější metodě – konvolučním neuronovým sítím 2.2.3, jejím základním prvkům a architektuře, která dnes dominuje v této oblasti počítačového vidění. Kapitola 3 se zaměřuje na konkrétní algoritmy detekce 3.2 a rozpoznávání 3.3, které se pro řešení této problematiky používají a datasety, které se v těchto řešeních používají pro trénování, testování a validaci 3.1.

Kapitola 4 je věnována používaným technologiím pro vestavěné systémy, konceptu *edge computing* 4.3 a mechanismům akcelerace 4.4, jejich způsobům a cílům na různých úrovních. Dále se zaobírá hardwarovým akcelerátorem *Intel Neural Compute Stick 2* a sadě nástrojů OpenVINO 4.4.3, které jsou velmi používané mezi vývojáři v oblastech chytré domácnosti, video monitorovacích systémech pro domácnosti nebo malé firmy a dalších, kde je aplikováno strojové vidění, řešené neuronovými sítěmi. Kapitola 5 se věnuje implementaci a návrhu bezpečnostního systému, který je postaven na výše zmiňovaném základu. Je zde popsán návrh a implementace backendu i uživatelského rozhraní. Poslední kapitola 6 je věnována experimentům, zaměřeným na validaci jednotlivých modelů rozpoznávání a na zhodnocení rychlosti rozpoznávání na jednotlivých platformách.

Kapitola 2

Teoretický úvod do problematiky rozpoznávání podle obličeje a neuronových sítí

Lidé jsou fenomenální v rozpoznávání obličejů. Mozek člověka disponuje čtyřmi oblastmi, které se tímto problémem zabývají. Člověk snímá obličej holisticky, zaměřuje se na proporce jednotlivých částí obličeje vůči sobě, narozdíl od ostatních objektů, které si zapamatovává na základě tvarů, které v mozku skládá dohromady [1]. Studie z roku 2018 [2], která se zabývala otázkou, kolik obličejů je si člověk schopen zapamatovat, dospěla k rozmezí 1 000-10 000 obličejů a průměrné hodnotě 5 000. První práce, zabývající se rozpoznáváním obličejů počítačem z obrazu, se objevily již v 70. letech minulého století [3]. Od té doby tato oblast počítačového vidění zaznamenala významný pokrok v přesnosti a rychlosti rozpoznávání, a to zejména díky hlubokým neuronovým sítím, které jsou inspirovány biologickými sítěmi, ale oproti nim jsou značně zjednodušeny a používají jiné mechanismy učení, jelikož jejich cílem je řešení problémů.

Systémy rozpoznávání se skládají z následujících částí [3]. Detektor obličeje slouží k lokalizaci obličeje v obraze a jeho výstupem jsou souřadnice obdélníku ohraničující nalezený obličej (*bounding box*). Poté, co je obličej detekován, je proveden jeho preprocesing. Cílem této části je extrakce obličeje z obrazu na základě výsledků detekce, změna měřítka, normalizace jasu, vylepšení kvality snímku a zarovnání obličeje pomocí referenčních bodů obličeje vhodnou afinní transformací vyhovující nalezeným bodům. Dále následuje část reprezentace obličeje, kde dochází k převedení hodnot pixelů obrazu na vektor vlastností, který se označuje jako *embedding*. V ideálním případě má obličej patřící stejnému člověku v různých obrazech podobný vektor vlastností. V poslední části, porovnávání obličejů, dochází ke srovnávání dvou vektorů vlastností. Výsledkem je hodnota podobnosti, kterou lze převést na pravděpodobnost, že obličeje popsané vektorem vlastností patří stejnému člověku.

Obrázek 2.1: Schéma architektury systému pro rozpoznávání obličejů.

2.1 Reprezentace obličeje

Reprezentace obličeje představuje proces kódování obrazové informace do matematického formátu, který lze použít pro další analýzu a zpracování. Jejím účelem je zachytit klíčové rysy obličeje (tvar očí, úst a nosu), které jsou relevantní pro identifikaci osoby. Jedná se bezpochyby o nejdůležitější část systému pro rozpoznávání tváří a od 70. let byla pro tento úkol vyvinuta řada metod.

Dle [3] první metody používaly pro rozpoznávání porovnávání jednoduchých vlastností, které vystihovaly geometrické vlastnosti obličeje. Na tyto metody navázaly metody holistické, které se řadí mezi statistické metody a pro rozpoznávání používaly celý obličej jako vstup. Současně docházelo k vývoji metod založených na vlastnostech, které staví rozpoznávání na extrakci důležitých vlastností obličeje a jejich vzájemné porovnávání mezi snímky obličejů. Na tyto metody poté navázaly metody hybridní, které kombinovaly metody holistické a metody založené na vlastnostech. Do nedávné doby dosahovaly nejvyšší úspěšnosti, ale byly předstiženy hlubokým učením, které je v dnešní době nejpoužívanější metodou pro počítačové vidění, včetně systému pro rozpoznávání obličejů. Následující text obecně shrnuje metody používané pro rozpoznávání obličejů, od tradičních metod (holistické metody, metody založené na vlastnostech a hybridní metody), po hluboké neuronové sítě.

2.1.1 Holistické metody

Holistické metody [3][4] využívají pro rozpoznávání celou oblast obličeje, kterou projektují do nízkodimenzionálního prostoru. Tato projekce extrahuje důležité vlastnosti obličeje a vypouští redundatní data a data nepotřebná pro rozpoznání obličeje. Tyto vlastnosti zřetelně zachycují rozdíly mezi jednotlivými tvářemi a umožňují jednoznačnou identifikaci obličeje. Nejpoužívanějšími holistickými metodami jsou metody PCA (*Analýza hlavních komponent*) a LDA (*Lineární Diskriminační Analýza*). Obě tyto metody poskytují sadu vlastností, které obsahují pouze podstatné informace pro klasifikaci obličejů.

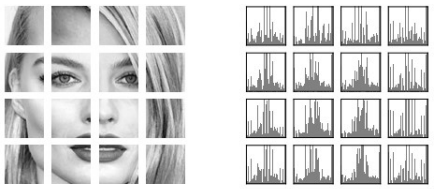
Metoda PCA je statistická metoda, která se používá ke snížení dimenze dat při co nejmenší ztrátě informací. Tato metoda se používá k vytvoření vlastních vektorů (*eigenfaces*, znázorněné na obrázku 2.2) z trénovací sady obličejů, které reprezentují jejich charakteristické rysy. Vlastní vektory vytvářejí podprostor, ve kterém každý *eigenface* představuje určitý směr a slouží jako referenční obrazec pro porovnání s rozpoznávaným obličejem.

Obrázek 2.2: Eigenfaces [3].

Metoda LDA představuje zobecnění Fisherova lineárního diskriminantu, metody používané ve statistice a dalších oborech, k nalezení lineární kombinace vlastností, které charakterizují nebo oddělují dvě nebo více tříd objektů nebo událostí. Cílem metody LDA je nalezení malého množství vlastností, které od sebe rozlišují jednotlivé obličeje, ale rozpoznají obličej patřící stejnému jedinci.

2.1.2 Metody založené na vlastnostech

Metody založené na vlastnostech (*feature-based*) [3] využívají extrahované lokální rysy obličeje (oči, uši, nos, brada a obrys hlavy) pro rozpoznání a zaměřují se na extrakci diskriminačních vlastností obličeje. Nejpoužívanějšími metodami založených na vlastnostech jsou metody LBP a SIFT. Metoda LBP rozdělí snímek na lokální regiony, ze kterých nezávisle extrahuje descriptory, které zřetězí do globálního vektoru vlastností (obrázek 2.3). Tato metoda má mnoho variant. Metoda SIFT transformuje snímek na velkou sbírku lokálních vektorů, kdy každý je invariantní vůči posunu, škálování a rotaci a částečně invariantní vůči změnám osvětlení a afinní nebo 3D projekci [5]. Má tři různé metodologie, přičemž ta nejpřesnější metodologie dle [3] počítá deskriptory přes běžnou mřížku a jako skóre podobnosti používá průměrnou vzdálenost mezi dvěma korespondujícími deskriptory.



Obrázek 2.3: (a) Snímek obličeje rozdělený na lokální regiony (b) Histogramy LBP deskriptorů pro jednotlivé lokální regiony [3].

Metody založené na vlastnostech jsou více robustní, než holistické metody [3]. Pokud porovnáváme dva obličeje patřící stejné osobě, jeden s otevřenýma a druhý se zavřenýma očima, při použití holistické metody by se koeficienty vektorů vlastností mohly odlišovat. Při použití metody založené na vlastnostech se koeficienty vektorů vlastností budou lišit pouze v oblasti očí.

2.1.3 Hybridní metody

Hybridní metody [3] kombinují techniky holistických metod s metodami založenými na vlastnostech. Nejpoužívanějším hybridním přístupem je extrakce vlastností obličeje pomocí metod LBP nebo SIFT a poté jejich projekce do nižší dimenze a diskriminačního podprostoru metodami PCA nebo LDA. Limitace hybridních metod dle [3] spočívá v problematické extrakci všech vlastností, které jsou potřebné pro rozpoznání obličeje. Proto se některé přístupy, jako například *GaussianFace*, snaží tuto limitaci překonat kombinováním různých typů vlastností, jiné zavedly fázi učení pro zlepšení diskriminační schopnosti jednotlivých vlastností.

GaussianFace [3] je založena na předpokladu, že obličejové deskriptory, jako například LBP deskriptory, jsou generovány z latentních proměnných, které jsou modelovány pomocí Gaussova procesu. Začlenila kernel LDA pro naučení reprezentace obličeje z LBP deskriptorů, která může využít data z více zdrojových domén. Použitím této metody byla dosažena přesnost 98,52 % na datasetu LFW. Toto je konkurenceschopné s přesností dosaženou mnoha metodami hlubokého učení.

2.1.4 Metody založené na neuronových sítích

Rozpoznávání obličejů v obraze není triviální úkol, jelikož se člověk mění, mění se jeho vzhled, stárne a je v obraze často zachycen z různých úhlů. Pro rozpoznávání obličeje v obraze jsou nejčastěji používány konvoluční neuronové sítě [3]. Tyto sítě se učí extrahovat relevantní rysy obličeje, jako jsou poloha očí, nosu, rtů apod., a zakódovat je do kompaktního vektoru, který obsahuje významné informace o obličeji, které jsou nezávislé na okolním osvětlení, pozici nebo výrazu v obličeji. Výstupem je reprezentace obličeje v podobě vektoru, který se označuje jako *face embedding*. Tento vektor, o velikosti 128 nebo 512 hodnot, obsahuje zakódované informace o rysech obličeje, pro následné porovnávání s jinými obličeji za účelem identifikace nebo verifikace osoby.

2.1.5 Metriky pro porovnávání vektorů vlastností

Při rozpoznávání obličejů je nezbytné měřit podobnost mezi dvěma obličeji, aby bylo možné rozhodnout, zda se jedná o stejnou osobu, či nikoliv. K tomu se používají metriky, které kvantifikují rozdíly mezi obličeji na základě jejich vektorů vlastností (*face embedding*). Porovnávání vektorů vlastností obličeje se provádí typicky pomocí jedné ze dvou metrik, kterými jsou euklidovská vzdálenost a kosinová podobnost [6]. Kosinová podobnost je poté často interpretovaná pomocí kosinové vzdálenosti. V této sekci se budeme podrobněji zabývat euklidovskou vzdáleností a kosinovou podobností, jako dvěma základními metrikami používanými při porovnánání obličejů.

Euklidovská vzdálenost

Euklidovská vzdálenost (*Euclidean distance*) [7] je nejběžnější používanou metriku při rozpoznávání obličejů [6]. Tato metrika se používá pro měření vzdáleností mezi dvěma body v euklidovském prostoru. Euklidovská vzdálenost mezi dvěma body v euklidovském prostoru se vypočítá jako délka nejkratší cesty (přímé spojnice) mezi těmito dvěma body.

Euklidovská vzdálenost mezi libovolnými dvěma body na reálné ose je rovna absolutní hodnotě jejich numerického rozdílu souřadnic, tedy jejich absolutnímu rozdílu. Pokud tedy u a v jsou dva body na reálné ose, pak vzdálenost mezi nimi je dána vzorcem:

$$d(u, v) = \sqrt{(u - v)^2} \tag{2.1}$$

Euklidovská vzdálenost v euklidovské rovině je definována: Nechť bod p má kartézské souřadnice (u_1, u_2) a bod v má souřadnice (v_1, v_2). Potom vzdálenost mezi body u a v je dána vzorcem:

$$d(u, v) = \sqrt{(v_1 - u_1)^2 + (v_2 - u_2)^2} \tag{2.2}$$

Tento vzorec vychází z aplikace Pythagorovy věty na pravoúhlý trojůhelník s horizontální a vertikální stranou, kde úsečka mezi body u a v je přepona. Dva vzorce umocněné uvnitř odmocniny udávají plochy čtverců na horizontální a vertikální straně, a vnější odmocnina převede plochu čtverce na přeponě na délku této přepony.

Kosinová podobnost

Kosinová podobnost (*Cosine similarity*) [7] je matematická metoda pro měření podobnosti mezi dvěma vektory v n-rozměrném prostoru. V problematice rozpoznávání obličejů se používá k měření úhlu mezi dvěma vektory vlastností obličeje [6]. Tato metrika zohledňuje

směr a orientaci vektorů, což umožňuje lépe zachytit podobnost mezi obličeji, i když se mohou lišit ve velikosti, či měřítku.

Tato podobnost se vypočítá jako cosinus úhlu mezi vektory. Výsledkem kosinové podobnosti je číslo v intervalu <-1, 1>, kde 1 znamená, že vektory jsou shodné, 0 znamená, že jsou vektory nezávislé a -1 znamená, že jsou vektory naprosto odlišné. Kosinová podobnost dvou vektorů v n-rozměrném prostoru je dána vztahem [8]:

$$cosinesimilarity = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\|\|\mathbf{y}\|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}, \tag{2.3}$$

kde x_i a y_i jsou i-té složky vektorů x a y.

Kosinová vzdálenost

Kosinová vzdálenost[1] (*Cosine distance*) úzce souvisí s kosinovou podobností, protože se jedná o vzájemně komplementární hodnoty. Jedná se o matematickou metodu pro měření vzdálenosti mezi dvěma vektory v n-rozměrném prostoru. Vztah kosinové vzdálenosti a podobnosti je dán vztahem [8]:

$$cosinedistance = 1 - cosinesimilarity, \tag{2.4}$$

Obecně platí, že kosinová podobnost roste se zmenšující se kosinovou vzdáleností, a naopak kosinová podobnost klesá s rostoucí kosinovou vzdáleností. Hodnoty kosinové vzdálenosti se tedy pohybují v intervalu <0, 2>, kde dva vektory svírající úhel 0° mají kosinovou vzdálenost 0 a jsou si maximálně podobné, a naopak dva vektory, které mezi sebou svírají úhel 180°, mají kosinovou vzdálenost 2 a jsou si minimálně podobné.

2.1.6 Metriky pro vyhodnocení kvality klasifikátorů

Vyhodnocování kvality klasifikátorů je nezbytným krokem, aby bylo možné posoudit úspěšnost modelu a porovnat jej s jinými modely. Metriky, které se používají k vyhodnocení kvality klasifikátorů, jsou různorodé a každá z nich poskytuje jiný pohled na úspěšnost klasifikačního modelu. Mezi nejčastěji používané metriky [9] pro hodnocení binárních klasifikátorů patří přesnost (*accuracy*), preciznost (*precision*), senzitivita (*recall*), specificita (*specificity*) a F1 skóre. V této části kapitoly si přiblížíme tyto metriky a ukážeme, jak je použít k vyhodnocení kvality klasifikačního modelu.

Důležité termíny používané pro binární klasifikaci jsou:

- TP (*True positive*) = klasifikátor označí příklad jako pozitivní a toto označení odpovídá skutečnosti
- TN (*True negative*) = klasifikátor označí příklad jako negativní a toto označení odpovídá skutečnosti
- FP (*False positive*) = klasifikátor označí příklad jako pozitivní a toto označení neodpovídá skutečnosti

[1] https://medium.datadriveninvestor.com/cosine-similarity-cosine-distance-6571387f9bf8

- FN (*False negative*) = klasifikátor označí příklad jako negativní a toto označení neodpovídá skutečnosti

Přesnost (*accuracy*) [9] je základní metrika, která udává jak často klasifikátor správně klasifikoval data. Přesnost je vhodná pro vyhodnocování modelů s vyváženými třídami (stejný počet pozitivních a negativních případů), u nevyvážených tříd přesnost nemusí zachytit menšinovou třídu.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.5}$$

Specificita (*specificity*) [9] a senzitivita (*recall*) [9] jsou doplňující se metriky, kdy specificita udává kolik skutečně negativních predikcí bylo správně identifikováno a senzitivita naopak udává, kolik skutečně pozitivních predikcí bylo identifikováno správně.

$$Specificity = \frac{TN}{TN + FP}, \tag{2.6}$$

$$Recall = Sensitivity = \frac{TP}{TP + FN}, \tag{2.7}$$

Preciznost (*precision*) [9] (vzorec 2.8) udává kolik pozitivních predikcí bylo skutečně pozitivních. V případě nevyvážených tříd může být těžké dosáhnout vysoké preciznosti a senzitivity zároveň. Z tohoto důvodu se často používá metrika F1 skóre [9] (vzorec 2.9), která představuje harmonický průměr mezi precizností a senzitivitou, jinými slovy slouží k nalezení kompromisu mezi precizností a senzitivitou. F1 skóre představuje hodnotu v intervalu <0,1>, kdy čím je hodnota vyšší, tím je klasifikátor kvalitnější.

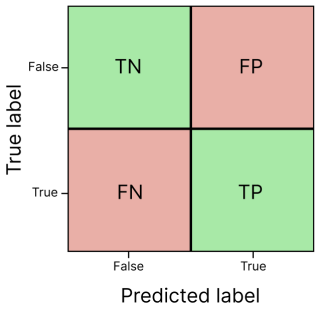
ROC (*Receiver Operating Characteristic*) [9] křivka vizualizuje vztah mezi hodnotami TPR (*True positive rate*) (vzorec 2.10) a FPR (*False positive rate*) (vzorec 2.11) různých prahových hodnot. TPR popisuje, jak je model úspěšný v predikování pozitivní třídy, pokud je opravdový výsledek pozitivní. FPR naopak udává, jak často model predikuje pozitivní třídu, když je skutečný výsledek negativní.

Jako kvalitní klasifikační model je označován model, který průměrně přiřazuje vyšší pravděpodobnost k náhodně zvolenému skutečnému pozitivnímu výskytu než k negativnímu. Obecně jsou kvalitní klasifikační modely reprezentovány křivkami, které se ohýbají k levému hornímu rohu grafu. Nekvalitní klasifikátor je takový, který nedokáže rozlišovat mezi třídami a vždy předpoví náhodnou třídu nebo konstantní třídu. Takový klasifikátor je reprezentován diagonální linií od dolního levého rohu grafu k hornímu pravému rohu.

AUC (*Area Under the Curve*) [9] je metrika, která měří celkovou schopnost klasifikátoru rozlišit mezi pozitivními a negativními příklady na základě pravděpodobností, přiřazených k jednotlivým příkladům. AUC je vypočítáno jako plocha pod křivkou ROC. Pohybuje se v rozmezí 0 až 1, kde vysoká hodnota AUC (blížící se k 1) indikuje, že klasifikátor má vysokou schopnost rozlišit mezi pozitivními a negativními příklady, zatímco nízká hodnota AUC (blížící se k 0,5) naznačuje, že klasifikátor nedosahuje lepších výsledků než náhodný klasifikátor.

Tabulka záměn (*confusion matrix*), zobrazena na obrázku 2.4, vizualizuje hodnoty TP, TN, FP a FN pro posouzení schopností a výkonu klasifikačního algoritmu. Na základě této tabulky lze určit výkonnostní metriky jako jsou přesnost, specificita, senzitivita, preciznost a F1 skóre. Osa x reprezentuje výstup klasifikačního algoritmu a osa y správný výsledek.

PR (*Precision-Recall*) [9] křivka vizualizuje vztah mezi hodnotami preciznosti a senzitivitou. Umožňuje porovnat vztah mezi precizností a senzitivitou při různých úrovních prahů pro klasifikaci. Průměrá přesnost AP (*Average precision*) [9] je metrika, která sumarizuje PR křivku do jediné hodnoty. Vypočítá se jako průměr precision hodnot na každé



Obrázek 2.4: Schéma tabulky záměn

$$Precision = \frac{TP}{TP + FP}, \tag{2.8}$$

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{2.9}$$

$$TPR = Sensitivity = \frac{TP}{TP + FN} \tag{2.10}$$

$$FPR = 1 - Specificity = 1 - \frac{TN}{TN + FP} \tag{2.11}$$

recall úrovni (když se mění práh), váženým podle změny recall od předchozího kroku. Tento ukazatel poskytuje ucelenější a spolehlivější hodnocení výkonu klasifikátoru než jednoduchá míra přesnosti nebo úplnosti. V praxi, čím blíže je PR křivka ke krajnímu pravému hornímu rohu, tím je výkon klasifikačního algoritmu lepší a hodnota AP vyšší.

2.2 Základy neuronových sítí

Tak jako ptáci inspirovali lidstvo létat, architektura mozku inspirovala lidstvo k vytvoření umělých neuronových sítí. Překvapivě se první zmínka o umělých neuronových sítích objevila v roce 1943, kdy neurofyzik W. McCulloch a matematik W. Pitts publikovali práci, ve které vytvořili matematický model, který měl simulovat práci neuronů v mozku zvířat. Druhá vlna zájmu přišla na začátku 80. let, ale až během posledních 10 let se jim dostalo zasloužené pozornosti [10].

Umělé neuronové sítě byly inspirovány biologickými neuronovými sítěmi, ačkoli se v některých ohledech značně liší. Biologické neuronové sítě jsou složeny z neuronů, které přijímají signály od jiných neuronů přes synapse. Když je signál dostatečně silný, neuron vyšle signál dál do dalších neuronů v síti [11]. Umělé neuronové sítě mají podobnou architekturu, kde jsou umělé neurony (uzly) propojeny pomocí vážených hran, které mají vlastnosti podobné synapsím. Každý uzel provádí výpočet výstupní hodnoty aplikováním určité funkce, která je tvořena vektorem vah a prahem (*biasem*), na vstupní hodnoty [10]. Nicméně, biologické neuronové sítě jsou výrazně složitější a dynamičtější než umělé neuronové sítě. V biologických neuronových sítích jsou neurony spojeny pomocí desítek tisíc synapsí a neustále se mění síla spojení mezi neurony. Zároveň jsou ovlivněny dalšími prvky, jako jsou neuromodulace a oscilace [11], které nejsou v umělých neuronových sítích implementovány.

Následující sekce je věnována popisu umělého neuronu a jeho inspiraci biologickým neuronem, dále se věnuje popisu jednovrstvého a vícevrstvého perceptronu a učícího algoritmu backpropagation. Nakonec se zaměří na architekturu konvoluční neuronové sítě, které se nejčastěji používají pro rozpoznávání obličeje v obraze.

2.2.1 Biologický neuron

Neuron [11] je základní funkční jednotkou nervové soustavy. Je tvořen tělem nervové buňky a jejími výběžky. Nervová buňka je schopna přijímat určité formy signálů, odpovědět speciálními signály, vést a vytvářet specifické funkční kontakty (synapse) s ostatními neurony. Neurony jsou organizovány do rozsáhlé sítě, kde je každý neuron spojen s N tisíci dalšími neurony a tato síť tvoří nervovou tkáň v lidském těle.

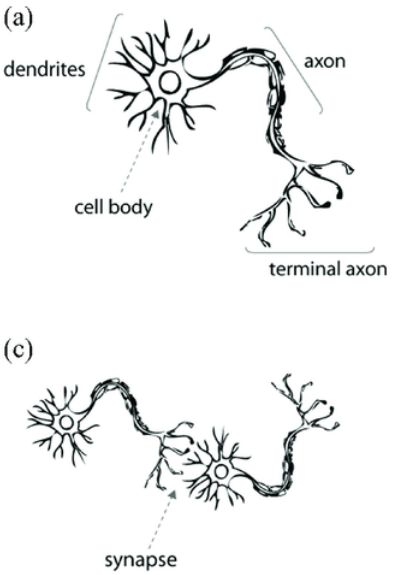
Tělo nervové buňky je tvořeno neuroplazmou, jádrem a jedérkem a je ohraničeno plazmatickou membránou, která podmiňuje vznik a šíření vzruchu. Výběžky buňky jsou dvojího typu – dendrity a neurity (axony). Dendrity přijímají vstupní informace od synapsí a je jich zpravidla větší počet, naopak axony vedou vzruch směrem od těla neuronu, přičemž délka axonu může dosahovat až desetitisícátého násobku velikosti samotného těla neuronu. Schéma neuronu je popsáno na obrázku 2.5.a. Jako synaptické terminály (synapse), znázorněny na obrázku 2.5.c, jsou označovány všechny funkční kontakty mezi membránami dvou buněk, pomocí kterých neurony přijímají krátké elektrické, nebo chemické signály od jiných neuronů. K přenosu vzruchu dochází typicky z axonu jednoho neuronu na dendrit druhého neuronu, případně přímo na tělo neuronu.

Každý neuron má svůj klidový membránový potenciál, který je dán koncentrací iontů vně a uvnitř membrány. Akční potenciál vzniká vychýlením membránového potenciálu o 15 mV, k vyhodnocování akčního potenciálu všech dendritů dochází v místě iniciálního segmentu. Vzniklý akční potenciál se nadále šíří axonem neuronu až do jeho knoflíkového rozšíření (presynaptické membrány), odkud přecházejí na postsynaptickou mebránu druhého neuronu.

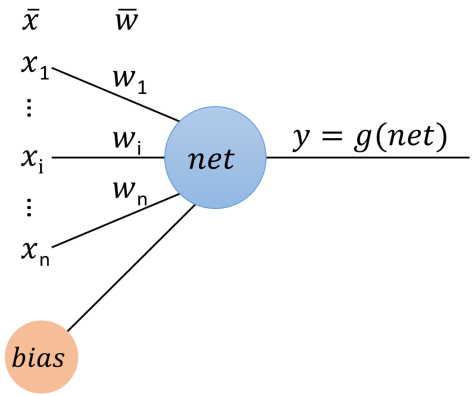
2.2.2 Umělý neuron a učení

Umělý neuron (uzel) [12] je základní procesorovou jednotkou umělých neuronových sítí a je inspirován biologickým neuronem. Je to uzel, který zpracovává všechny výstupy z jiných uzlů a generuje výstup podle přenosové funkce nazývané aktivační funkce.
Model neuronu je znázorněn na obrázku 2.6. Lineární bázová funkce (LBF) určuje hodnotu vnitřního potenciálu neuronu a je dána vztahem:

$$net = \sum_{i=1}^{n} w_i x_i - \theta = w^T x - \theta, \tag{2.12}$$



Obrázek 2.5: (a) lidský neuron, (c) synapse [13]

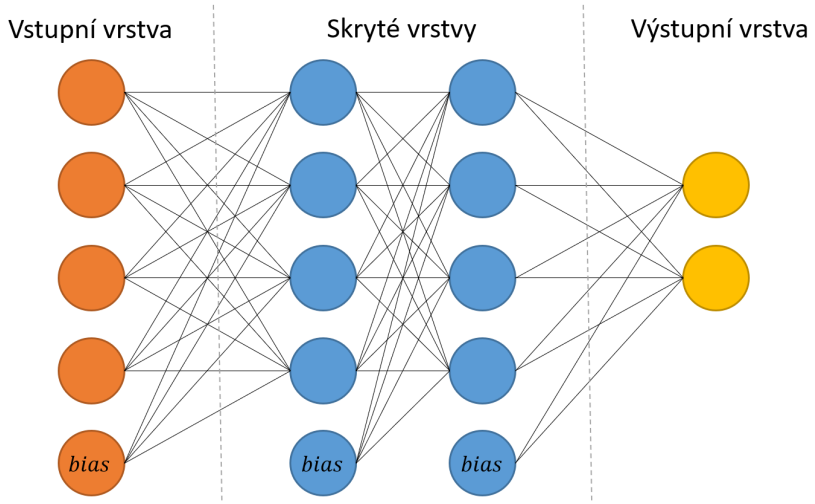


Obrázek 2.6: Model umělého neuronu. Synapse jsou modelovány vektorem vah $\vec{w}$, které přivádějí vstupy $\vec{x}$ do těla neuronu. Tělo neuronu je tvořeno vnitřním potenciálem neuronu net, aktivační funkcí g a výstupem y, který je dán $y = g(net)$.

kde θ značí práh (*threshold*) $\theta = -b$, b je bias, x_i je i-tý vstup, w_i je váha vstupu, n je počet vstupů. [12] Aktivační funkce $g(net)$ slouží k výpočtu výstupu neuronu, provádí nelineární transformaci vnitřního potenciálu na jednu, obecně reálnou hodnotu v intervalu (-1, 1) nebo (0, 1). Může být skoková, nebo spojitá. Nejjednodušším neuronem se skokovou aktivační funkcí

$$y = \begin{cases} 1 & \text{pro } net \geq 0 \\ -1 & \text{pro } net < 0 \end{cases} \tag{2.13}$$

je perceptron. Perceptron [10] je schopen klasifikovat lineárně oddělitelné číselné vektory do dvou tříd. Spojením několika perceptronů vzniká jednovrstevný perceptron, který je schopen klasifikovat vstupní vektor $\vec{x}$ do několika tříd. Některá omezení perceptronů lze eliminovat jejich vrstvením, tím vznikají vícevrstvé perceptrony, které umožňují vyřešit například problém XOR. Vícevrstvé perceptrony MLP (*Multi Layer Percepton*) [12] obrázek 2.7 jsou schopny klasifikovat lineárně neseparovatelné vzory a aproximovat funkce. Jsou to dopředné sítě, které se skládájí z jedné vstupní vrstvy perceptronů, jedné nebo více skrytých vrstev a poslední výstupní vrstvy. Každá vrstva kromě výstupní vrstvy obsahuje *bias* neuron a je plně propojena s další vrstvou. Pokud má taková síť více než dvě skryté vrstvy, označuje se jako hluboká neuronová síť.



Obrázek 2.7: Architektura MLP.

Strojové učení je podoblast umělé inteligence, která umožňuje počítačovým systémům se učit. Zabývá se algoritmy a technikami, které se zabývají změnou vnitřního stavu systému tak, aby vykovával stejnou činnost účinněji a efektivněji. Strojové učení lze rozdělit na 3 typy: učení s učitelem, učení bez učitele a posilované učení. Pro učení neuronových sítí se využívá učení s učitelem, které se provádí na trénovací množině $T = \{(\vec{x_1}, y_1), ..., (\vec{x_N}, y_N)\}$, kde $\vec{x_d}$ je vstupní vektor hodnot, y_d je správný výstup a $d \in \{1, ..., N\}$ [14]. Pro každý příklad je systému poskytnut vstupní vektor hodnot $\vec{x_d}$ a správný výstup y_d a cílem učení je minimalizovat odchylku výstupu sítě $\hat{y_d}$ od správného výstupu y_d.

Nejpopulárnější metodou učení s učitelem dopředných vícevrstvých neuronových sítí je algoritmus Backpropagation (BP), který používá gradient sestupu. Tento algoritmus byl

poprvé představen v 70. letech minulého století a od 80. let je hojně využíván v oblasti počítačového vidění a rozpoznávání řeči, především díky své efektivitě.

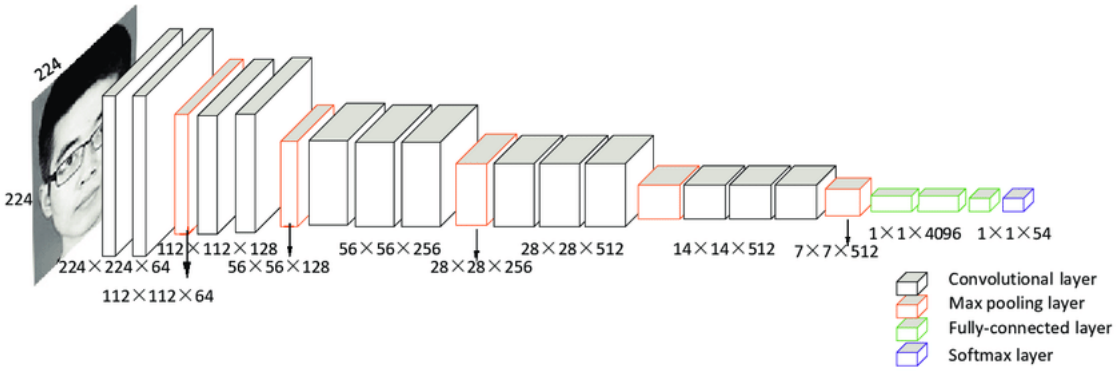
Dle [14] lze průběh iterace BP rozdělit do čtyř fází: dopředná fáze, zpětná fáze, výpočet gradientů a fáze úpravy vah. Jednotlivé fáze jsou prováděny za předpokladu vhodného parametru rychlosti učení α a náhodné inicializace vah w_{ij}^k. V dopředné fázi jsou každé dvojice vstupně-výstupních hodnot $(\vec{x_d}, y_d)$ nejprve zpracovávány skrytými vrstvami sítě a poté jsou postoupeny výstupní vrstvě jejímž výsledkem je výstup $\hat{y_d}$. Hodnoty $\hat{y_d}$, a_j^k, o_j^k každého neuronu j vrstvy k jsou ukládány. Ve zpětné fázi dochází k porovnání výstupu dopředné fáze $\hat{y_d}$ s referenční hodnotou y_d a výpočtu chyby δ_1^m. Poté dochází ke zpětné propagaci chyby do skrytých vrstvev δ_j^k, kde $k = m-1$ a zároveň je vyhodnocována parciální derivace chyby E_d s ohledem na váhu w_{ij}^k. Ve fázi výpočtu gradientů je proveden výpočet celkového gradientu pro páry $X = \{(\vec{x_1}, y_1), ..., (\vec{x_N}, y_N)\}$, jako průměr individuálních gradientů. Pro výpočet gradientu, musí být aktivační funkce neuronu spojité, tj. derivovatelné, proto často neurony dopředných neuronových sítí používají sigmoidní aktivační funkce $y = \frac{1}{1+e^{-u}}$. Ve finální fázi – fázi úpravy vah dochází k úpravě vah jednotlivých neuronů, podle rychlosti učení α a výsledného gradientu, posunutím proti směru gradientu.

2.2.3 Konvoluční neuronové sítě

Konvoluční neuronové sítě (CNN) se řadí mezi hluboké neuronové sítě, protože jsou tvořeny mnoha vrstvami. Jejich vznik byl inspirován vizuálním vnímáním živých tvorů. První předchůdce konvolučních neuronových sítí se objevil v roce 1980 a v roce 1989 publikovali LeCun a kolegové významnou práci [15], která stanovila moderní rámec CNN a byla později rozšířena. V rámci práce vyvinuli vícevrstvou umělou neuronovou síť nazvanou LeNet-5, která mohla klasifikovat ručně psané číslice, byla tvořena několika vrstvami a mohla být trénována algoritmem backpropagation 2.2.2. Nicméně kvůli nedostatku trénovacích dat a výpočetního výkonu ve své době, nedokázala dosáhnout dobrých výsledků při řešení složitějších problémů, jako klasifikace velkoformátových obrázků a videa. Přesto byla LeNet-5 významným přínosem pro rozvoj konvolučních neuronových sítí a inspirovala vznik dalších úspěšných architektur, jako je například síť AlexNet [16]. Jednou z prvních úspěšných konvolučních neuronových sítí, která byla použita pro rozpoznávání obličejů v obraze byla *DeepFace* společnosti Facebook, která dosáhla 99,74% přesnosti. Tato CNN byla natrénovaná na datasetu obsahující 4,4 miliónů snímků obličeje 4 030 různých osob [3].

Konvoluční neuronové sítě jsou trénovány na velkém množství variantních dat a jsou schopny se učit, přičemž učení (trénování) spočívá v upravování biasů a vah a vede ke vzniku filtrů. Při aplikaci CNN pak tyto filtry procházejí přes obraz a extrahují rysy. Proces trénování konvolučních neuronových sítí obvykle vyžaduje velké množství výpočetních zdrojů a trénovacích dat, ale pokud je sítě správně natrénována, může být velmi účinná v řešení mnoha různých úkolů zpracování obrazu a analýzy dat.

Architektura konvoluční neuronové sítě je tvořena mnoha vrstvami, jedná se tedy o hlubokou neuronovou síť. Konvoluční neuronová síť se skládá z velkého množství konvolučních vrstev (*Convolution layer*), které jsou následovány sdružujícími vrstvami (*Pooling layer*). Dále jsou tvořeny plně propojenými vrstvami (*Fully-connected layer*) a ztrátovou funkcí (*Loss function*) [17]. Jednotlivé vrstvy a jejich sled jsou znázorněny na schématickém obrázku 2.8 modelu VGG16, který je používán pro rozpoznávání obličejů z obrazu. Následující podkapitola je věnována popisu zmiňovaných vrstev.



Obrázek 2.8: Architektura modelu VGG16 [18].

Konvoluční vrstva

Konvoluční vrstva provádí matematickou operaci, konvoluci, za účelem zmenšení a zobecnění výstupu. Každá vrstva konvoluční neuronové sítě používá vlastní jádro (filtr) pro konvoluci s vstupními daty. Filtr je typicky náhodně inicializován při trénování a postupně se jeho hodnoty optimalizují pomocí algoritmu backpropagation. To umožňuje konvoluční neuronové síti adaptovat se na specifické funkce a zlepšuje výkonnost sítě v dané úloze. Vzorec konvoluce obrazu je [19]:

$$I(u,v) = \sum_{i=-a}^{b} \sum_{j=-c}^{d} h(i,j) I(u-i, v-j), \tag{2.14}$$

kde I je vstupní obrázek, $h(i,j)$ je filtr o rozměru $(a+b+1) \times (c+d+1)$.

V případě dvou-dimenzionální diskrétní konvoluce lze jádro chápat jako tabulku koeficientů (filtr), která je položena na příslušné místo obrazu. Cílem filtrů je extrakce důležitých vlastností z obrazu. Každý pixel překrytý tabulkou je vynásoben koeficientem v příslušné buňce a je proveden součet všech těchto hodnot. Výsledkem je jeden nový pixel. [20] Příklad konvoluce obrázku I a filtrem K:

$$I * K = \begin{pmatrix} i_{00} & i_{01} & i_{02} & i_{03} & i_{04} \\ i_{10} & i_{11} & i_{12} & i_{13} & i_{14} \\ i_{20} & i_{21} & i_{22} & i_{23} & i_{24} \\ i_{30} & i_{31} & i_{32} & i_{33} & i_{34} \\ i_{40} & i_{41} & i_{42} & i_{43} & i_{44} \end{pmatrix} * \begin{pmatrix} k_{00} & k_{01} & k_{02} \\ k_{10} & k_{11} & k_{12} \\ k_{20} & k_{21} & k_{22} \end{pmatrix} = \begin{pmatrix} z_{11} & z_{12} & z_{13} \\ z_{21} & z_{22} & z_{23} \\ z_{31} & z_{32} & z_{33} \end{pmatrix}$$

Každá vrstva obsahuje jiné filtry a tím dosáhne extrakce různých vlastností ze zpracovávaného snímku.

Jeden ze způsobů, jak dále zmenšit počet parametrů je zvětšit velikost kroku (*stride*) a zmenšit tak překrytí jednotlivých kroků konvoluce. Nevýhodou konvoluce je ztráta informací na okraji obrazu. Tento problém je řešen ohraničením nulou (*zero-padding*), a tím umožnění jádra konvoluce provést výpočet i pro pixely na okraji obrazu. Podobným přístupem je reflexe, kdy ohraničení obrazu je zopakováno až v několika vrstvách a tím jsou zachovány informace na okraji obrazu.

S konvoluční vrstvou je spjata nelineární aktivační funkce [17], která určuje výstup neuronu ve vícevrstvých neuronových sítích, protože její výstup není lineárně oddělitelný. Funkce je aplikovaná za účelem saturace nebo limitace výstupu konvoluční vrstvy. Nejpoužívanější aktivační funkcí je funkce ReLu (*Rectified Linear Unit*) [17], která oproti ostatním

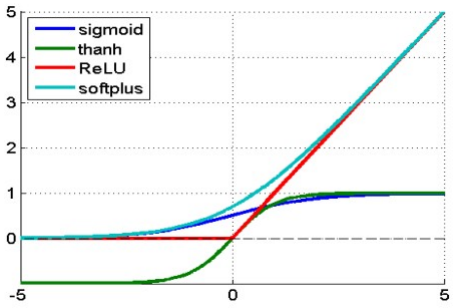
nelineárním aktivačním funkcím (than a sigmoid, porovnání viz obrázek 2.9) má jednodušší definici funkce a gradientu. Má konstantní gradient pro pozitivní vstupy a nulový gradient pro záporné vstupy, což je ve prospěch tréninku.

Výstupem konvoluční vrstvy (a nelineární aktivační funkce) je mapa vlastností (*feature map*) a naučená jádra jsou detektory vlastností. S přibývajícími konvolučními vrstvami roste složitost a schopnost sítě vyjádřit více propracovanější vlastnosti obrazu.

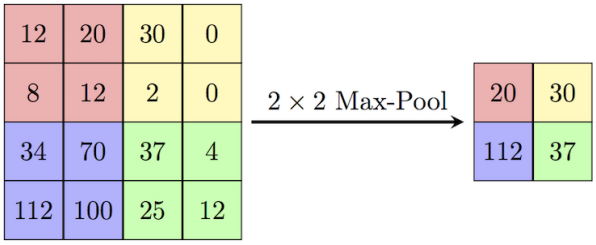
Pooling vrstva

Pooling vrstva [17] je vrstva, jejíž hlavní myšlenkou je *down-sampling* – snížení složitosti pro další vrstvy, tato operace lze přirovnat ke snížení rozlišení obrazu. Sdružování nemá vliv na počet filtrů.

Dle [20] je sdružování obvykle implementováno poté, co je vstup filtrován pomocí konvoluce a prošel nelineární aktivační funkcí. Existuje více než jeden typ sdružování, přičemž obecný princip sdružování je nahrazení aktuálního výstupu na určitém místě souhrnnou statistikou blízkých výstupů. Příkladem sdružování je nahrazení čtyř pixelů jedním pixelem obsahujícím maximální hodnotu z původních čtyř (Max-pooling, viz obrázek 2.10), nebo jejich průměrnou hodnotou, či podobně.



Obrázek 2.9: Porovnání nelineárních aktivačních funkcí [17].



Obrázek 2.10: Princip Max-poolingu [21].

Sdružování zmenšuje rozměr a shrnuje celé okolí výstupů na úkor obětování jemných detailů. Zlepšuje výpočetní efektivitu a paměťové požadavky sítě, protože snižuje počet vstupů do další vrstvy a pomáhá zpracovávat vstupy různých rozměrů, protože umožňuje regulovat velikost výstupu po sdružení.

Plně spojená vrstva

Plně propojená vrstva (*Fully connected layer* nebo také *Dense layer*) [17] je podobná uspořádání neuronů v tradiční neuronové síti. Každý uzel v plně propojené vrstvě je přímo připojen ke každému uzlu v předchozí i následující vrstvě, jak je znázorněno na obrázku 2.7. Hlavní nevýhodou plně propojené vrstvy je, že obsahuje mnoho parametrů, které vyžadují komplexní výpočet, proto se snažíme počet eliminovat uzly a spojení pomocí techniky dropout.

Dropout vrstva

Dropout vrstva [22] zabraňuje *overfittingu* (tj. velký rozdíl v chybovosti při tréninku a testování CNN) a poskytuje způsob, jak efektivně kombinovat mnoho různých architektur

neuronových sítí. V této vrstvě dochází během tréninku k vyřazení jednotek v neuronové síti, čímž se rozumí její dočasné odstranění ze sítě spolu se všemi jejími příchozími a odchozími připojeními. Volba, které jednotky vypustit, je náhodná. V nejjednodušším případě je každá jednotka zachována se stejnou pravděpodobností p nezávisle na jiných jednotkách. Pravděpodobnost p se typicky pohybuje v rozsahu 0.5-1.

Kapitola 3

Moderní metody pro detekci a rozpoznávání obličeje

Detekce a rozpoznávání obličeje jsou v současné době velmi důležitými úkoly v oblasti počítačového vidění, protože se využívají v širokém spektru aplikací, jako jsou například bezpečnostní systémy. S nástupem pokročilých technik hlubokého učení se objevují stále sofistikovanější metody, které umožňují dosáhnout vysoké přesnosti a robustnosti v detekci a rozpoznávání obličejů.

V této kapitole budou představeny nejnovější metody pro detekci a rozpoznávání obličeje v obraze, které se v současné době používají, včetně používaných datasetů pro jejich trénování a testování. Tyto metody zahrnují moderní techniky založené na hlubokých neuronových sítích, jako jsou konvoluční neuronové sítě. Tyto metody se vyznačují vysokou úspěšností při detekci a rozpoznávání obličeje a jsou schopny pracovat i s velkým množstvím dat a s různými podmínkami nasvícení a pozicí hlavy.

3.1 Datasety

Datasety jsou soubory dat, na kterých jsou neuronové sítě trénovány, validovány a testovány, přičemž obsah datasetů má velký podíl na úspěšnosti detekce a rozpoznávání.

Systémy detekce a rozpoznávání obličejů musí v reálných situacích čelit řadě výzev. I když systémy zaznamenaly významný pokrok v dobře kontrolovaných prostředích, stejné techniky v reálných situacích mohou selhat. Výzvu pro úspěšné rozpoznání představuje primárně osvícení, natočení obličeje, stárnutí, výraz v obličeji, plastické operace, make-up, nízké rozlišení snímků a zakrytí obličeje. Proto je důležité, aby byl systém trénován dostatečně rozmanitými snímky, aby v reálných situacích neselhal.

3.1.1 MS-Celeb-1M

MS-Celeb-1M [23] dataset byl vytvořen společností Microsoft Research a skládá se z přibližně 10 milónů snímků obličejů 100 000 různých celebrit. Tento rozsáhlý dataset byl původně vytvořen pro účely výzkumu rozpoznávání obličejů a snažil se být co nejvíce reprezentativní pro různé kategorie celebrit a různé výzvy v rozpoznávání obličejů. Nicméně, dataset se stal kontroverzním kvůli otázkám ochrany osobních údajů a zneužití pro účely sledování, což vedlo k tomu, že některé výzkumné instituce a firmy odmítly používat tento dataset.

3.1.2 VGGFace2

VGGFace2 [24] dataset je zatím největším volně dostupným datasetem pro výzkumné účely. Skládá se z přibližně 3,31 miliónu snímků obličeje 9 131 různých osob různých etnik. Je přibližně gendrově vyvážený (poměr počtu snímků mužů je 59,3%), přičemž průměr počtu snímků na osobu je 362,6. Dataset je rozdělen na dvě části – trénovací s 8 631 třídami a hodnotící s 500 třídami.

3.1.3 CASIA-WebFace

CASIA-WebFace [25] je veřejně dostupný dataset snímků schromážděných z filmové databáze IMDb. Obsahuje 494 414 snímků obličeje 10 575 různých osob a je přibližně gendrově vyvážený (zastoupení snímků mužů činí 58,9%). Rozlišení snímků je 250×250 pixelů, některé snímky jsou barevné a obsahují výzvy v osvětlení, výrazech a pózách. Dataset byl semi-automaticky vyčištěn od chybových označení snímků, ale ne zcela.

3.1.4 Labeled faces in the Wild

Labeled faces in the Wild (LFW) [26] obsahuje 13 233 snímků obličeje 5 749 různých osob, přičemž 1 680 osob má v datasetu dva nebo více snímků. Snímky byly získány z webu a jsou označeny jménem osoby, která se na snímku nachází. Jako detektor obličeje byl použit Viola-Jones algoritmus. Dataset obsahuje snímky které představují různé výzvy jako osvícení, natočení tváře, výrazy, zakrytí obličeje a nízké rozlišení snímků. Většina snímků je barevných, ve formátu JPEG a má rozlišení 250×250 pixelů.

3.1.5 YouTube Faces Database

YouTube Faces Database (YTF) [27] obsahuje 3 425 videí 1 595 různých osob, které byly stažené z platformy YouTube. Každá osoba se průměrně objevuje ve 2,5 videích. Nejkratší klip má 48 snímků a nejdelší klip má 6 070 snímků. Tento dataset spolu s LFW je používán spíše pro tesování, jelikož jejich velikost nemusí být dostačující pro trénování.

Dataset	Počet snímků	Počet osob	Rozlišení	Barva
MS-Celeb-1M	10 miliónů	100 000	Různé	Různé
VGGFace2	3,31 miliónů	9 131	137×180 avg.	Barevné
CASIA	494 414	10 575	250×250	Různé
LFW	13 233	5 749	250×250	Různé
YTF	3 425 videí	1 595	200×200	Grayscale

Tabulka 3.1: Porovnání datasetů pro rozpoznávání obličejů.

3.1.6 WIDER FACE

WIDER FACE dataset [28], dataset pro detekci obličeje, je podmnožinou veřejně dostupného datasetu WIDER. Obsahuje 32 203 snímků zachycující 393 703 obličejů různých velikostí, póz, zakrytí obličeje a rušného pozadí. Tyto faktory činí dataset náročným, ale jsou důležité pro splnění požadavků reálných systémů detekce. Pro každý snímek dataset obsahuje soupis souřadnic obdélníků ohraničující obličeje osob, přičemž obdélník obsahuje

čelo, tvář a bradu, a anotaci pózy a úrovně zakrytí obličeje. Každá anotace byla provedena jedním anotátorem a zkontrolována dvěma dalšími lidmi. WIDER FACE je rozdělen do tří kategorií podle úrovně obtížnosti detekce: *Easy*, *Medium* a *Hard* pro lepší zhodnocení přesnosti detektoru. Dataset je také náhodně rozdělen do tří podmnožin pro trénování (40 %), validaci (10 %) a testování (50 %).

3.2 Algoritmy pro detekci tváří

Detekce obličeje je nedílnou součástí systému pro rozpoznávání obličeje, jelikož obličej musí být nejprve v obraze detekován, aby mohl být rozpoznán. Typicky jsou výstupem detektoru obličeje souřadnice obdélníku *bounding boxu* kolem obličeje, popřípadě i souřadnice dalších důležitých bodů obličeje – očí, nosu a úst. Tato sekce se zabývá populárními metodami detekce objektů (obličejů), používající se v dnešní době.

3.2.1 You Only Look Once detektor

YOLO (*You Only Look Once*) [29] je rychlý a efektivní objektový detekční algoritmus, který byl vyvinut v roce 2015. Jedná se o první jednostupňový detektor objektů, postavený na konvolučních neuronových sítích, kterou používá pro predikci ohraničujících obdélníků a pravděpodobnosti tříd přímo ze vstupních obrazů během jedné evaluace. Přístup dělí vstupní obraz na mřížku buněk a přímo predikuje souřadnice a klasifikaci pro každou buňku. I když dosahoval mnohem vyšší rychlosti detekce, oproti vícestupňovým detektorům, jeho přesnost byla s jejich porovnáním horší. Vývojem vzniky jeho další verze, a to YOLOv2, YOLOv3, které zachovaly jeho rychlost a zvýšily výkon na úroveň dvoustupňových detektorů. Hlavním problémem YOLO je, že jeho výkon klesá při detekci objektů malého měřítka, proto autoři YOLO-face [29] představili detektor postavený na detektoru YOLOv3 ale upravený pro detekci obličejů různého měřítka při zachování rychlosti YOLO detektoru. Porovnání detektorů YOLO je zobrazen na obrázku 3.1.

Architektura YOLO-face je postavena na YOLOv3, přičemž autoři YOLO-face navrhli vylepšení *backbone*, ztrátové funkce a kotevních rámečků. Autoři vylepšili strukturu sítě původního vlastního *backbone* Darknetu-53 zvýšením počtu vrstev sítě prvních dvou reziduálních bloků, pro získání adekvátnějších rysů obličeje malého měřítka. Dále představili novou ztrátovou funkci. Ztrátová funkce, kterou YOLO používá je vícesložková a skládá se ze složek ztráty regrese, ztráty důvěry, ztráty klasifikace a ztráty odpovědné za nedetekování objektu. Tyto složky jsou v poměru 1:1:1:1. Pro přizpůsobení ztrátové funkce pro detekci obličeje autoři upravili poměr jednotlivých složek na 2:1:0.5:0.5 a představili novou funkci ztráty regrese, která kombinuje ztráty MSE a GIoU. Poté použili k-means shlukování pro výpočet kotevních rámečků. Pro natrénování modelu autoři použili dataset WIDER FACE (viz kapitola 3.1.6) a pro hodnocení datasety FDDB a WIDER FACE.

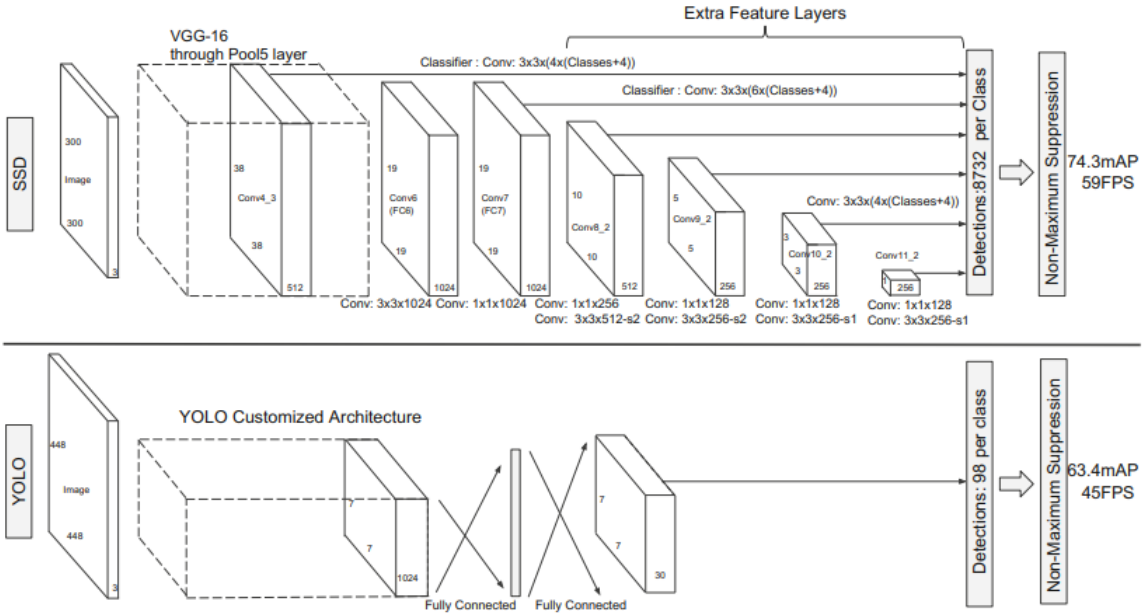
Dnes jsou dostupné YOLO modely verze 5 a 7, které také nabízejí své modely zaměřené pouze na detekci obličejů.

3.2.2 Single Shot MultiBox detektor

Single Shot MultiBox Detector (SSD) [30] je další populární metoda pro detekci objektů v obraze. Tato metoda byla vyvinuta v roce 2016 a od té doby se stala jednou z nejpopulárnějších metod pro detekci objektů v obraze. Je používána v různých typech aplikací, včetně aplikací určených pro detekci obličejů, vozidel, osob, objektů apod.

Obrázek 3.1: Porovnání úspěšnosti (zleva) YOLOv2, YOLOv3 a YOLO-faces [29].

SSD se skládá ze dvou komponentů: základního modelu (*backbone*) a hlavy (*head*) SSD. Základním modelem je předtrénovaná síť (ResNet, ImageNet, VGG-16 apod.) pro klasifikaci obrazů, která je využita jako extraktor rysů. Hlava SSD je jedna nebo více konvolučních vrstev přidaných na tento základní model, jejichž výstupy jsou interpretovány jako *bounding boxy* a třídy objektů na prostorové pozici, aktivací finálních vrstev. Tato metoda kombinuje detekci objektů a predikci jejich ohraničujících rámečků v jediném kroku (*single shot*).



Obrázek 3.2: Porovnání detekčních modelů SSD a YOLO. SSD model přidává na konec základní sítě několik vrstev funkcí, které předpovídají posuny k výchozím rámečkům různých měřítek a poměrů stran a jistotu přiřazení [30].

Tato reprezentace umožňuje efektivně modelovat prostor možných tvarů rámečků. Tímto způsobem je SSD schopen detekovat objekty různých velikostí a tvarů a je vhodný pro aplikace s vysokým požadavkem na rychlost.

Non-Maximum Suppression

Non-Maximum Suppression (NMS) [30], vizualizovaná na obrázku 3.2, je technika, která se používá k potlačení překrývajících se detekcí jednoho objektu. Eliminuje redundatní detekce výběrem nejvíce spolehlivé detekce a potlačením ostatních. Proces spočívá v třídění

detekovaných ohraničujících obdélníků podle jejich skóre spolehlivosti (pravděpodobnosti obsahu objektu). Algoritmus prochází seřazený seznam a vybírá ohraničující rámeček s nejvyšším skóre jako referenční a potlačuje ostatní ohraničující rámečky, které mají vysoký překryv s referenčním rámečkem. Překryv se měří pomocí metriky *Intersection over Union* (IoU) [30], která vypočítává poměr plochy průniku mezi dvěma ohraničujícími rámečky a jejich sjednocenou plochou. Pokud je hodnota IoU mezi dvěma ohraničujícími rámečky nad určitým prahem, ohraničující rámeček s nižším skóre spolehlivosti je potlačen.

3.3 Algoritmy pro rozpoznávání tváří

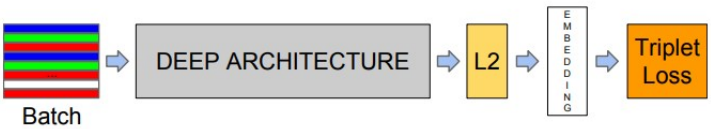
Rozpoznávání tváří je důležitou oblastí počítačového vidění, která má širokou škálu aplikací, jako je identifikace, autentizace, sledování osob, apod. V posledních letech se objevily různé pokročilé algoritmy pro rozpoznávání tváří, které dosahují vysoké přesnosti a robustnosti. V této kapitole se zaměříme na populární algoritmy FaceNet, SphereFace a ArcFace, které se liší v architektuře a trénovacím postupu, ale všechny se snaží naučit se vektorovou reprezentaci obličeje, která je robustní vůči změnám v podmínkách. Konkrétně použití těchto metod umožňuje dosáhnout vysoké přesnosti rozpoznávání obličejů. Budeme se věnovat jejich základním principům, architektuře a klíčovým vlastnostem.

3.3.1 FaceNet

Model FaceNet byl postaven výzkumnou skupinou společnosti Google v roce 2015 [31]. Model mapuje obličeje osob do shluků geometrických bodů (Eukleidovských prostorů) označovaných jako *embedding*, který je získán z míry podobnosti a rozdílnosti obličejů.

Výzkumná skupina představila ve své práci [31] dvě různé architektury hluboké konvoluční neuronové sítě. První architektura je postavena na modelu Zeiler&Fergus, který se skládá z více prokládaných vrstev konvolucí, nelineárních aktivací, lokálních normalizací a Max-pooling vrstev. K tomuto modelu bylo dodatečně přidáno několik vrstev konvoluce $1 \times 1 \times d$. Druhá architektura je založena na Inception modelu, který používá smíšené vrstvy, které paralelně provádí konvoluci a sdružování vrstev a poté sdružují svoje výstupy.

Struktura modelu, zobrazena na obrázku 3.3, je navržena tak, že vstupní snímek obličeje x modelu FaceNet je předán hluboké konvoluční neuronové síti, na jejíž výstup je aplikována normalizace ℓ_2. Výstupem je *face embedding* $f(x) \in \mathbb{R}^d$, který představuje 128-mi prvkovou vektorovou reprezentaci obličeje. Dále následuje ztrátová funkce *Triplet loss*, která porovnává trojice obličejů, které se skládají ze dvou obličejů stejné osoby a jednoho cizího.



Obrázek 3.3: Architektura modelu FaceNet [31].

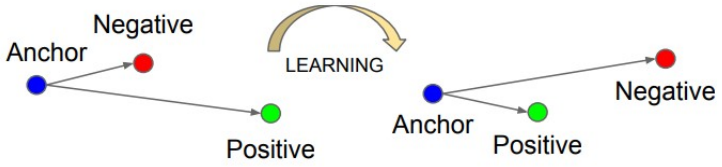
Embedding $f(x)$ mapuje snímek x do d-dimenzionálního Eukleidovského prostoru, přičemž je omezen na d-dimenzionální hypersféru tj. $||f(x)||_2 = 1$. Cílem ztrátové funkce *Triplet loss* je minimalizovat vzdálenost mezi právě porovnávaným snímkem obličeje osoby

x_i^a (*anchor*) a všemi jeho ostatními snímky x_i^p (*positive*) a naopak maximalizovat vzdálenost mezi snímkem x_i^a a snímky cizích obličejů x_i^n (*negative*) jak znázorňuje obrázek 3.4. Pro vzdálenosti x_i^a, x_i^p a x_i^n, pak platí [31]:

$$||x_i^a - x_i^p||_2^2 + \alpha < ||x_i^a - x_i^n||_2^2, \forall (x_i^a, x_i^p, x_i^n) \in \tau, \tag{3.1}$$

kde α je vynucená hranice mezi pozitivními a negativními páry a τ je soubor všech možných trojic v trénovacím setu s kardinalitou N.

Správný výběr trojic je důležitý pro rychlou konvergenci. Dosažení vyšší přesnosti spočívá ve volbě složitých trojic výpočtem, který spočívá ve výběru snímku s pozitivní identitou $argmax_{x_i^p} ||f(x_i^a) - f(x_i^p)||_2^2$ (hard positive) a negativní identitou $argmin_{x_i^n} ||f(x_i^a) - f(x_i^n)||_2^2$ (hard negative). Volbu složitých trojic nelze provádět skrze celý dataset, proto existují dva způsoby řešení – offline generování a online generování. Při offline generování je každých n kroků počítán *argmax* a *argmin* na části datasetu. Pro online generování jsou složité trojice vybírány z mini-batch, který představuje malé množství snímků reprezentující trénovací dataset.



Obrázek 3.4: *Triplet Loss* [31].

Ztrátová funkce *Triplet Loss*, znázorněna na obrázku 3.4, je definována jako:

$$L_{TripletLoss} = \sum_i^N \left[||f(x_i^a) - f(x_i^p)||_2^2 - ||f(x_i^a) - f(x_i^n)||_2^2 + \alpha \right]_+ \tag{3.2}$$

Pro trénink FaceNetu byl použit způsob online generování na mini-batch o velikosti 1800 snímků. Model byl validován na setu přibližně milionu snímků obličeje, nejvyšší úspěšnosti dosáhl model s architekturou Inception a velikostí snímku 224×224, a to 89,4 %±1, 6. Model s architekturou Zeiler&Fergus a velikostí snímku 220 × 220 dosáhl úspěšnosti 87,9 %±1, 9. Při testování přesnosti klasifikace pro jednotlivé testovací datasety, autoři dosáhli přesnosti 99,63 % pro dataset *Labeled faces in the Wild* (viz kapitola 3.1.4) a pro *YouTube Faces DB* (viz kapitola 3.1.5) přesnosti 95,12 %.

3.3.2 SphereFace

Autoři SphereFace [25] představili v rámci své práce, publikovanou v roce 2017, ztrátovou funkci *A-Softmax* odvozenou od ztrátové funkce *softmax*, která je popsána rovnicí 3.4. Ztrátová funkce *A-Softmax* (*Angular Softmax*) byla navržena s cílem naučit se rozlišovací vlastnosti obličeje s jasnou geometrickou interpretací, jež do té doby žádný dostupný algoritmus rozpoznávání obličejů nenabízel.

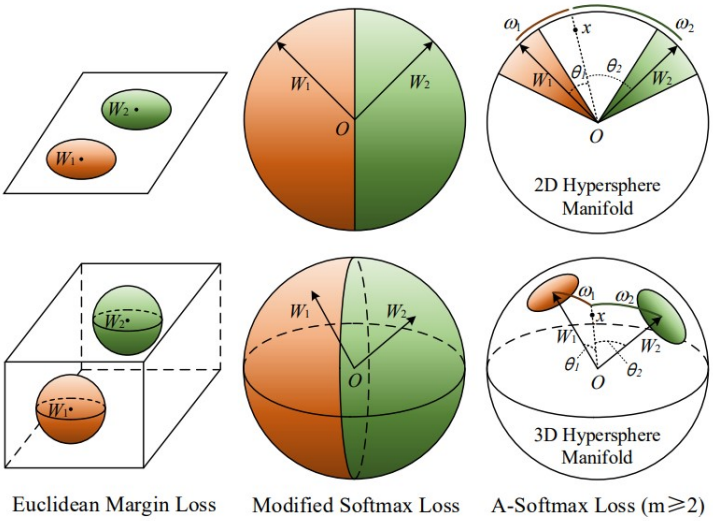
Naučené vlastnosti obličeje vytváří metriku diskriminační úhlové vzdálenosti, která je ekvivalentní k geodetické vzdálenosti na hypersféře. Naučené vlastnosti tak mají jasnou geometrickou interpretaci. *A-Softmax* ztrátová funkce umožňuje upravit *angular margin* přes parametr m, přičemž čím větší m, tím větší *angular margin*. Požadované nastavení

parametru m je takové, aby maximální vzdálenost v rámci jedné třídy byla menší než minimální vzdálenost mezi třídami.

Autoři SphereFace provedli normalizaci vah $||W_i|| = 1$ a vynulovali bias $b_i = 0$ ztrátové funkce *softmax*. Výsledkem je ztrátová funkce *A-Softmax*:

$$L_{A-Softmax} = \frac{1}{N}\sum_i -\log\left(\frac{e^{||x_i||\psi(\theta_{y_i,i})}}{e^{||x_i||\psi(\theta_{y_i,i})} + \sum_{j\neq y_i} e^{||x_i||\cos(\theta_{j,i})}}\right), \tag{3.3}$$

kde $\psi(\theta_{y_i,i}) = (-1)^k \cos(m\theta_{y_i,i}) - 2k$, $\theta_{y_i,i} \in \left[\frac{k\pi}{m}, \frac{(k+1)\pi}{m}\right]$ a $k \in [0, m-1]$. Parametr m ovlivňuje velikost *angular margin*, pokud $m = 1$ jedná se o ztrátovou funkci *modified softmax*, která je popsána rovnicí 3.5.



Obrázek 3.5: Srovnání ztrátových funkcí Modified Softmax a SphereFace [25].

Při experimentech s jedením modelem, trénovaný na datasetu CASIA-WebFace (viz kapitola 3.1.3), SphereFace dosáhl na datasetech LFW (viz kapitola 3.1.4) a YTF (viz kapitola 3.1.5) úspěšnosti 99,42 % a 95,0 %.

3.3.3 ArcFace

ArcFace (*Additive Angular Margin loss*) je ztrátová funkce poprvé představena v roce 2018 [32]. Vychází z předcházející práce SphereFace 3.3.2, která zavedla koncept *angular margin*, který pomáhá zlepšit separabilitu tříd a tím i výkonnost rozpoznávání obličejů. Nicméně, jejich ztrátová funkce vyžadovala řadu aproximací, což vedlo k nestabilnímu trénování sítě. Kromě toho standardní *softmax* ztrátová funkce dominovala trénování, což znamenalo, že koncept *angular margin* nebyl plně využit. ArcFace přichází s novou ztrátovou funkcí, která se snaží tyto nedostatky řešit. Zavedl ztrátovou funkci *Additive Angular Margin*, která umožňuje lepší separabilitu tříd a stabilnější trénování, aniž by byly nutné aproximace používané ve SphereFace.

Autoři práce [32] dospěli ke ztrátové funkci *Additive Angular Margin* úpravou ztrátové funkce *Softmax*, která je definována [32]:

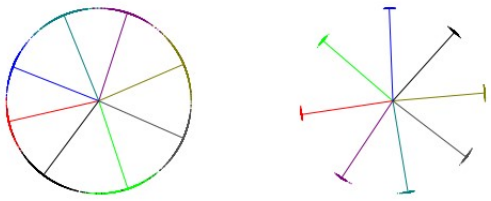
$$L_{softmax} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{W_{y_i}^T x_i + b_{y_i}}}{\sum_{j=1}^{N} e^{W_j^T x_i + b_{y_j}}}, \tag{3.4}$$

kde $x_i \in \mathbb{R}^d$ označuje vlastnosti i-tého vzorku patřícího do y_i-té třídy, d je hodnota 512, $W_j \in \mathbb{R}^d$ označuje j-tý sloupec váhy $W \in \mathbb{R}^{d \times N}$, $b_j \in \mathbb{R}^N$ je bias a N je číslo třídy.

Nejprve zafixovali bias $b_j = 0$. Následně transformovali $W_j^T x_i = ||W_j||\,||x_i|| \cos\theta_j$, kde θ_j je úhel mezi váhou W_j a vlastností x_i. Zafixovali $||W_j|| = 1$ normalizací ℓ_2. Dále zafixovali *embedding feature* $||x_i||$ normalizací ℓ_2 a přeškálováním do s. Normalizace vah a vlastností způsobí, že predikce závisí pouze na úhlu mezi vlastností a váhou. Protože *embedding features* jsou rozmístěny okolo centra vlastností v hypersféře, penalizace *additive angular margin* m mezi x_i a W_{y_i} současně podpoří kompaktnost uvnitř třídy a rozpor mezi třídami. *Additive angular margin* je rovna geodetické vzdálenosti v normalizované hypersféře. Výsledná ztrátová funkce má tvar:

$$L_{ArcFace} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s\cos(\theta_{y_i}+m)}}{e^{s\cos(\theta_{y_i}+m)} + \sum_{j=1, j\neq y_i}^{N} e^{s\cos\theta_j}} \tag{3.5}$$

Autoři provedli srovnání ztrátových funkcí softmax a ArcFace. Vytvořili dataset obsahující 8 tříd, každá obsahovala kolem 1500 snímků, pro natrénování 2D *feature embedding* sítě. Jak je vidět na obrázku 3.6 softmax poskytuje zhruba oddělitelné *feature embeding* ale nejednoznačné hranice mezi nejbližšími třídami, zatímco ArcFace poskytuje zřetelnější hranici.



(a) Softmax (b) ArcFace

Obrázek 3.6: Srovnání ztrátových funkcí Softmax a ArcFace [32].

Model s architekturou ResNet100 a trénovaný pomocí datasetu MS1MS0 dosáhl úspěšnosti 99,83 % při testování na datasetu LFW (viz kapitola 3.1.4) a úspěšnosti 98,02 % při testování na datasetu YTF (viz kapitola 3.1.5). Podobné úspěšnosti dosáhl model natrénovaný na datasetu IBUG500K, a to 99,83 % a 98,01 %.

Kapitola 4

Technologie pro vestavěné kamerové systémy

Tato kapitola poskytuje ucelený přehled o kamerových systémech, technologiích pro videozáznam a přenos dat, s důrazem na *edge computing* a akcelerační metody neuronových sítí na různých úrovních. Bude se zabývat jejich výhodami, využitím a možnostmi implementace v moderních systémech pro zpracování videa a dat.

4.1 Základní rozdělení kamerových systémů

Kamerový systém [33][34] je soubor kamer a souvisejících komponent, které slouží ke snímání a zpracování obrazu a umožňují uživatelům sledovat a zaznamenávat dění v určitém prostoru. Kamerové systémy jsou nejčastěji používány pro zabezpečení objektů a majetku, ale také se používají v průmyslu, vědě a v dalších oblastech. Kamerové systémy se podle používaných přesnosových technologií dělí do tří kategorií: analogové, AHD a digitální.

Základem analogových kamerových systémů [33] je kompozitní analogový videosignál snímaný bezpečnostní kamerou, který je přenášen běžným koaxiálním kabelem. Analogový signál je veden přenosovým vedením až do místa využití obrazové informace. Dnes však bývají analogové kamery připojeny k digitálnímu záznamovému zařízení, které zpracovává analogový signál do digitální podoby.

AHD systémy [33] představují nejnovější technologii pro analogový přenos obrazu ve vysokém rozlišení (HD) po koaxiálním kabelu nebo UTP. Převádí digitální signál do jednoho modulovaného analogového signálu a umožňují tak výrazně prodloužit délku přenosových tras a snížit kapacitu záznamových úložišť. Počet kamer je omezen na maximálně 32 v jednom nahrávacím zařízení, proto bude pravděpodobně tato technologie nahrazena IP technologií z důvodu omezeného rozlišení [34].

Digitální kamerové systémy [33] fungují na principu číslicově zpracovávaného signálu. Tyto systémy jsou plně digitální, protože neobsahují žádné analogové prvky a jejich základ tvoří sítové neboli IP kamery. Zpracovávání obrazu do digitální podoby probíhá přímo v kameře, proto jejich výhodou je možnost vysokého rozlišení digitálního obrazu, který není omezen normou analogového signálu. Obraz je přenášen jako celek v podobě paketu běžnou síťovou infrastrukturou. Digitální kamerové systémy jsou dnes nejpoužívanější, protože poskytují řadu výhod oproti starším analogovým systémům, jako je větší rozlišení, lepší kvalita obrazu, možnost digitálního zoomu, snadnější instalaci a konfiguraci a jsou schopny

přenášet data po síti, což umožňuje snadné sledování kamerového systému z jakéhokoli místa, kde je k dispozici připojení k internetu [33].

4.2 Technologie pro videozáznam a přenos dat

V posledních letech se stále více prosazují tzv. IP kamery, které jsou digitálními kamerami, které se připojují k počítačové síti přes protokol IP. Tyto kamery nabízejí ještě větší flexibilitu a možnosti využití, což dále podporuje trend směrem k digitálním kamerovým systémům. S rostoucím počtem zařízení a technologií, které produkují, zpracovávají a využívají video a data, se i požadavky na jejich efektivní a spolehlivý záznam a přenos stále zvyšují. Pro mnoho bezpečnostních systémů se stává *real-time* přenos videa zásadním požadavkem. RTSP je jedním z nejpoužívanějších protokolů pro streamování videa z IP kamer, existuje však řada dalších protokolů, jako například RTMP nebo HLS [35], které se také používají v kamerových systémech.

4.2.1 IP kamery

IP kamery jsou kamery, která přenáší obraz a zvuk přes internetový protokol IP (*Internet Protocol*). Takové kamery komprimují obraz a zvuk na digitální signál a pak ho přenášejí po síti, a to je odlišuje od tradičních analogových kamer, které pro přenos signálu využívají koaxiální kabel. IP kamery často poskytují výhody jako vyšší rozlišení obrazu, vzdálený přístup prostřednictvím internetu, možnost snadného propojení s jinými zařízeními v síti a možnost ukládání nahrávek na síťové úložiště nebo do cloudu.

Existuje mnoho různých typů IP kamer, mezi které patří stacionární kamery pro monitorování interiérů nebo exteriérů budov, PTZ (*pan-tilt-zoom*) kamery, které umožňují ovládání pohybu kamery do všech stran a zoom dálkově, a specializovaných kamer pro různé aplikace, jako je sledování dopravy nebo monitorování prostředí. IP kamery jsou široce používány ve sledovacích systémech, bezpečnostních systémech, dohledových systémech a dalších aplikacích, kde je potřeba monitorovat a zaznamenávat obraz a zvuk v reálném čase.

IP kamery často používají RTSP protokol pro streamování obrazu a zvuku v reálném čase. RTSP protokol (viz kapitola 4.2.4) je používaný pro streamování živého videa a audia přes síť. Kamery, které podporují RTSP, poskytují video a audio streamy ve formátu, který je kompatibilní s RTSP. Tyto streamy jsou přístupné prostřednictvím URL adresy, kterou lze zadat do aplikace nebo zařízení, které je schopné přijímat a přehrát RTSP streamy.

4.2.2 RTMP

Real Time Messaging Protocol (RTMP) [35] je tradiční protokol pro přenos zpráv, postavený na TCP. Je určen jak pro ovládací příkazy, tak pro přenos datových zpráv přes stejný TCP kanál a pevný port. Byl velmi populární pro streamování videí přes internet a byl integrován do platformy Adobe Flash, aby byl snadno použitelný. Výhody RTMP zahrnují ne-lineární přehrávání (uživatel může přeskakovat nebo přetáčet stream), není potřeba vyrovnávací paměti pro přehrávání a rychlé doručení je zajištěno pomocí dedikovaných streamingových serverů. Hlavní nevýhodou je potřeba přehrávače Flash Media Player, který není nyní široce podporován, a RTMP chybí podpora HTML5. RTMP protokol proto stále méně používán, a byl nahrazen modernějšími technologiemi, jako je například HLS nebo MPEG-DASH.

4.2.3 HLS

HLS (*HTTP Live Streaming*) [35] protokol byl původně vyvinutý společností Apple pro zasílání videa a audia na iPhone/iPad zařízení. Skládá se ze tří hlavních komponentů, kterými jsou server, distribuce a klientský software. Serverová komponenta přijímá vstupní média, provádí jejich digitálnímu kódování a připravuje zahrnutá média k distribuci. Distribuční komponenta se skládá ze standardních serverů, které jsou zodpovědné za přijímání požadavků klientů a následného doručení připraveného média a s ním spojených zdrojů. Posledním komponentem je klientovský software, který je zodpovědný za nalezení vhodného média k požadavku, sběr zdrojů a použití zdrojů k představení média uživateli v kontinuálním proudu.

IP kamery, které podporují HLS, mohou streamovat video pomocí segmentovaného přenosu dat, kdy je video stream rozdělen na menší segmenty. Tyto segmenty jsou následně posílány pomocí HTTP protokolu klientovi. Klient si vybírá, které segmenty si přehraje, a v případě ztráty konektivity nebo zhoršení síťových podmínek se může přepnout na nižší kvalitu videa.

4.2.4 RTSP

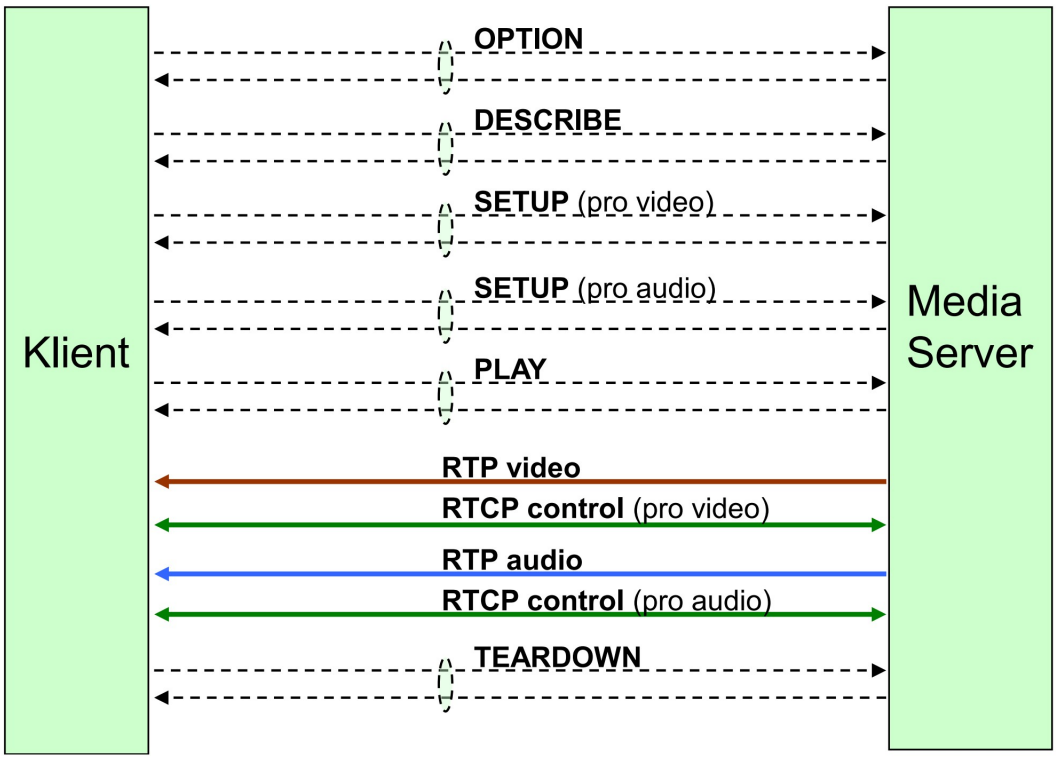
RTSP (Real-Time Streaming Protocol) [36] je síťový protokol na aplikační úrovni, určený pro kontrolu doručování dat v reálném čase. Řadí se mezi signalizační protokoly a jedná se o textový protokol se syntaxí podobnou HTTP. Protokol poskytuje rozšiřitelný rámec pro umožnění kontrolovaného doručení audio a video dat, přičemž zdroj dat může být živý, nebo ze záznamu. Protokol vytváří a řídí jeden, nebo několik časově synchronizovaných proudů audio a video dat a poskytuje prostředky pro výběr doručovacích kanálů, jako je UDP, multicast UDP a TCP a zároveň poskytuje prostředky pro výběr doručovacích mechanismů postavených na RTP.

Protokol [37][36] slouží k navázání, řízení a ukončení spojení. Navázání a ukončení spojení spočívá v domluvě mezi serverem a klientem na parametrech přenosu, řízení umožňuje ovládání jednoho, či několika časově synchronizovaných média streamů. Řadí se mezi out-of-band protokoly, protože samotná audio a video data jsou doručována jiným protokolem (např. RTP, MPEG-TS v UDP apod.), přičemž video a audio data jsou přenášeny separátními kanály. Server udržuje informace o stavu relace označené identifikátorem relace, nebo sekvenčním číslem.

Pro komunikaci, zobrazené na obrázku4.1, protokol využívá metody specifikované v [36], kde jsou podle volitelnosti zvýrazněny – volitelné, povinné, doporučené:

- DESCRIBE – metoda sloužící k inicializaci RTSP, požaduje zaslání popisu médií příslušících k URL
- ANNOUNCE – metoda zaslána ve směru klient-server požaduje popis médií příslušících k URL na server, ve směru server-klient metoda aktualizuje popis relace v reálném čase
- GET_PARAMETER – požadavek pro zisk hodnoty parametru streamu specifikovaným prostřednictvím URI
- OPTIONS – metoda pro žádost klienta o nestandartní požadavek, může být podána kdykoliv, nemá vliv na stav serveru
- PAUSE – metoda pro dočasné přerušení doručování streamu

- PLAY – metoda sdělí serveru, že má začít zasílat data prostřednictvím mechanismu specifikovaného metodou SETUP, metoda PLAY nemůže být zahájena, dokud nebyla úspěšně dokončena metoda SETUP
- RECORD – metoda zahájí nahrávání
- REDIRECT – požadavek informuje klienta o přesměrování na jiný server
- SETUP – požadavek specifikuje transportní mechanismus, který má být použit pro streamovaná média
- SET_PARAMETER – požadavek na nastavení hodnoty parametru streamu specifikovaným prostřednictvím URI
- TEARDOWN – absolutní ukončení doručování streamu a uvolnění rezervovaných prostředků



Obrázek 4.1: Schéma komunikace RTSP [37].

4.3 Edge Computing

Koncept *edge computingu* [38] se objevil ve spojitosti s IoT (*Internet of Things*), který nabyl v posledních letech velké popularity. *Internet of Things* je termín používaný pro popis sítě zařízení, která jsou propojena a komunikují mezi sebou pomocí internetu. Tato zařízení mohou být různorodá, od klasických elektronických zařízení (jako jsou chytré telefony, chytré hodinky, chytré domácnosti) až po průmyslová zařízení (jako jsou senzory, monitory, stroje) nebo zařízení v dopravě (jako jsou autonomní vozidla, chytré silnice). IoT umožňuje těmto zařízením sbírat data, komunikovat, analyzovat a provádět akce na základě těchto

dat. Nicméně tradiční modely *cloud computingu* se mohou potýkat s výzvami zpracování obrovského množství dat generovaných IoT a splněním praktických potřeb. V reakci na tyto události získal nový výpočetní model zvaný *edge computing* (EC) značnou pozornost od průmyslu i akademie.

Edge computing je koncept v oblasti informačních technologií, který se týká zpracování dat a výpočtů na okraji sítě, tedy blízko zdroje dat, namísto jejich přenosu do cloudového datového centra (*cloud computing*). Jako *device on edge* je označováno zařízení, které je umístěno přímo na okraji sítě nebo blízko u koncového uživatele. Tato zařízení jsou často nízkospotřebová a jsou schopna zpracovávat, analyzovat a vyhodnotit získaná data na místě, kde vznikají.
Výhody, které přináší *Edge Computing* jsou [38]:

1. **Snížení latence**: Zpracování dat na okraji sítě umožňuje rychlejší odezvu a akce na základě těchto dat.
2. **Bezpečnost**: Zpracování dat na místě jejich vzniku zvyšuje ochranu soukromí a bezpečnosti dat.
3. **Snížení zátěže sítě**: Při zpracování dat na okraji sítě se snižuje zátěž sítě.

Edge Computing může mít také nevýhody, kterými často jsou:

1. **Omezená výpočetní kapacita**: Edge zařízení obvykle mají omezené výpočetní kapacity ve srovnání s cloudem nebo centrálními datovými centry, a to může omezit možnosti provádění složitějších a výkonnostně náročnějších výpočtů.
2. **Omezená škálovatelnost**: Edge zařízení jsou obvykle navržena pro menší měřítko a nemusí být jednoduše škálovatelná na velké množství zařízení nebo dat. To může být omezením pro další rozšiřování systému.
3. **Omezené možnosti aktualizace a správy**: Správa a aktualizace edge zařízení může být náročnější, protože vyžaduje určitou míru znalosti systému a odbornosti uživatele.

Tato technologie se zaměřuje na poskytování služeb a provádění výpočtů na okraji sítě. Jejím cílem je přesunout cloudové zdroje (úložné a výpočetní kapacity a zdroje) blízko zdrojů dat a poskytovat služby, které potřebují provádět výpočty s nízkou latencí, bezpečně a s menší síťovou zátěží.

4.4 Akcelerační mechanismy

Pro dosažení dobrých výsledků rozpoznávání se architektura konvolučních neuronových sítí prohlubuje, stává se složitější, výpočetně náročnou a pomalou, kvůli této skutečnosti je náročná na implementaci. Právě díky velkému rozvoji počítačového vidění a jeho využití v různých odvětvích je potřeba CNN optimalizovat a akcelerovat. Následující text popisuje jednotlivé způsoby akcelerace na různých úrovních sítě od akcelerace na úrovni struktury sítě, akceleraci trénování na algoritmické úrovni, po implementační akceleraci s pomocí hardwarových akcelerátorů.

4.4.1 Strukturální akcelerace

Trénovací a inferenční procesy mohou být akcelerovány dvěmi hlavními způsoby: odstraněním redundance vah a redundance v reprezentaci nacházející se ve struktuře sítě. Re-

dundance vah může být odstaněna jejich predikcí, nebo nepotřebné váhy mohou být redukovány. Metody redukce vah jsou: metoda redukce vrstev, prořezání (*pruning*), bloková cirkulující projekce a destilace znalostí [39]. Mnoho vah v neuronových sítích jsou velmi malé hodnoty, blížící se nulové hodnotě. Většina aritmetických operací používá pro reprezentaci 32-bitovou hodnotu s plovoucí řádovou čárkou, za účelem zachování vysoké přesnosti, ale na úkor výpočetní náročnosti a paměťových nároků. Snížení bitové přesnosti do nižších řádů, vede k mnohonásobnému zrychlení konvolučních operací při poměrně minimálním snížení přesnosti. Odstranění redundance v reprezentaci znamená aproximovat reprezentaci dat za účelem dosažení vyšší rychlosti, snížení spotřeby a paměťových nároků hardwarové implementace.

Výsledkem akcelerace konvoluční neuronové sítě na strukturální úrovni je komprimovaná síť bez významné ztráty přesnosti.

4.4.2 Algoritmická akcelerace

Ve vícevrstvých dopředných neuronových sítích je typicky při průchodu vpřed počítán výstup konvoluční neuronové sítě a naopak při zpětném průchodu jsou upravovány váhy a *bias*. Snížením počtu iterací vedoucích ke konvergenci lze zkrátit dobu tréninku konvoluční neuronové sítě. I když konvoluce dramaticky snižují množství vah, opakované sčítání a násobení zvyšují výpočetní náročnost a stávají se náročné na implementaci. Metody akcelerace na algoritmické úrovni jsou: metoda optimalizace pomocí *Gradient decent* algoritmu a metoda zvýšení efektivity konvoluce [39].

Cílem akcelerace na algoritmické úrovni je zkrácení doby tréninku a zvýšení výkonu.

4.4.3 Implementační akcelerace

Neuronové sítě v poslední době znovu nabývají na síle díky vysoce výkonnému hardwaru. Dříve byly algoritmy strojového učení implementovány pro CPU, protože implementovaly techniky potřebné pro násobení matic, ale v dnešní době se používají GPU, FPGA, ASIC a VPU pro urychlení tréninku a inferenci. Kromě toho bylo navrženo mnoho nových zařízení pro potřeby velmi velkých modelů a velkých trénovacích datasetů. V následujícím textu jsou shrnuty některé hardwarové akcelerátory, které jsou slibné pro zrychlení hlubokých konvolučních neuronových sítí.

GPU

Grafická procesorová jednotka (GPU) je typ výpočetní technologie, navržená pro paralelní zpracování výpočtů. I když byla tato jednotka navržena primárně pro grafické a video renderování 3D grafiky a gaming, začíná být velice populární v oblasti umělé inteligence a její akcelerace.

GPU může být použito pro akceleraci rozpoznávání obrazu, protože disponuje mimořádným množstvím výpočetních prostředků pro velké množství tréninkových dat, výpočty matic a konvolucí a může zrychlit pracovní zátěž, která využívá paralelismus. Její instrukce, které jsou nativně podporované hardwarem, jsou fixované kvůli fixované architektuře GPU. GPU klastry mohou akcelerovat velké neuronové sítě využitím paralelismu, přičemž hlavní oblastí výzkumu kolem GPU jednotky je komunikace mezi jednotlivými klastry.

Paralelismus GPU je výhodný pro trénink konvoluční neuronové sítě, ale pro inferenci, kdy je zpracováván vždy jeden vstup, nemá žádný přínos. Nevýhody GPU jsou vyšší cena,

rozměry a vyšší spotřeba energie, které zabraňují využití GPU pro mobilní a malá zařízení [40][39].

FPGA

Programovatelná hradlová pole (*Field Programmable Gate Array* (FPGA)) [39][40] je typ logického integrovaného obvodu, které je navrženo tak, aby mohl být naprogramován až samotným zákazníkem a dává zákazníkům volnost úpravy zařízení speciálně pro potřeby algoritmů, které používají. FPGA paralelizuje zřetězené zpracování strojových instrukcí a práci s daty, takže má nižší latenci při zpracování úkolů. Je energeticky úspornější než GPU, při dosaženém stejném výkonu, zlepšuje účinnost a je používán pro inferenci. V současné době mnoho výzkumníků úspěšně implementuje CNN na FPGA, které využívá funkce FPGA ke zrychlení výpočtů a snížení spotřeby energie celé CNN.

ASIC

Zákaznický integrovaný obvod (*Application-Specific Integrated Circuit* (ASIC)) [39] je oproti FPGA čip, přizpůsobený pro konkrétní použití. Tensorová procesorová jednotka (*Tensor Processing Unit (TPU)*) je ASIC, který byl vyvinut pro datacentra společností Google, pro akceleraci strojového učení. ASIC, který byl navržen pro akceleraci malých systémů, vychází z Google TPU a je nazýván Edge TPU. Na Edge TPU jsou postaveny inferenční akcelerátory Google Coral Dev a Coral USB [41].

VPU

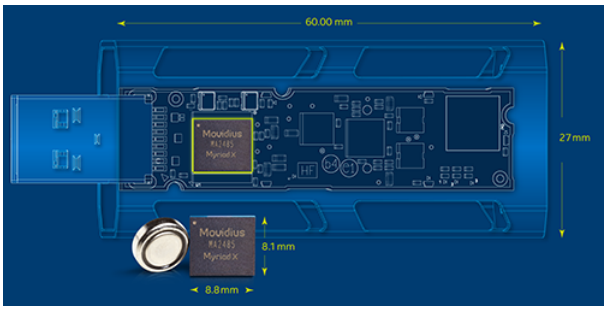
Akcelerátory vizuálních procesorových jednotek (*Vision Processing Unit* (VPU)) jsou čipy, vytvořené pro akceleraci zpracování obrazu pomocí počítačového vidění a algoritmů hlubokého učení.

Intel Neural Compute Stick 2 [42][43] je výkonné, levné a kompaktní řešení, s nízkou spotřebou, pro akceleraci neuronových sítí. Je navrženo pro běh hlubokých neuronových sítí ve vysokých rychlostech s nízkou spotřebou energie bez ztráty přesnosti, umožňující zpracovávat počítačové vidění v reálném čase. Je velmi žádané mezi vývojáři v oblastech chytré domácnosti, video monitorovacích systémech pro domácnosti nebo malé firmy a dalších, kde je aplikováno strojové vidění řešené neuronovými sítěmi.

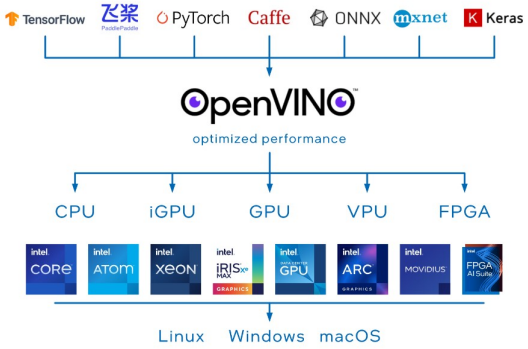
Jedná se o zařízení, které vypadá jako standartní USB flash disk a má rozhraní USB 3.0 typu A. Je postaveno na procesoru Intel Movidius Myriad X VPU MA2485 (obrázek 4.2), který obsahuje neurální výpočetní jednotku Neural Compute Engine (NCE), určenou pro hardwarovou akceleraci pro inferenci hlubokých neuronových sítí.

NCS2 je podporováno sadou nástrojů společnosti Intel OpenVINO [44]. Tato open-source sada nástrojů nabízí vývojové nástroje pro optimalizaci a nasazení modelů hlubokého učení. Poskytuje lepší výkon pro modely vidění, zvuku a jazyka populárních frameworků TensorFlow, Caffe, PyTorch a dalších. OpenVINO optimalizuje kanály hlubokého učení pomocí opětovného použití paměti, fúze grafů, vyvažování zátěže a odvození paralelismu napříč CPU, GPU, VPU a dalšími, viz obrázek 4.3. Na akcelerátory mohou být přeneseny, nebo integrovány další operace pro předběžné a následné zpracování, aby byla snížena latence mezi koncovými body a zlepšena propustnost.

Společnost Intel uvedla, že produkt NCS2 ukončuje [42]. Technickou podporu bude poskytovat do 30. června 2023 a záruční podporu do 30. června 2024. Jako nástupní technologii NCS2 Intel nabízí program Vision Accelerator Design with Intel Movidius VPU [45].



Obrázek 4.2: Architektura Intel NCS2 a porovnání fyzické velikosti procesoru MA2485 [42].



Obrázek 4.3: Sada nástrojů OpenVINO [44].

Kolaborací s dodavateli AAEON, ADLINK, Advantech, IEI, JWIPC a NEXCOM nabízí procesor Intel Movidius Myriad X VPU MA2485 v konfiguraci jednoho, dvou, nebo osmi MA2485 s podporou platformy OpenVINO. Oproti NCS2 jsou akcelerátory v provedení M.2, M-PCIe a PCIe, což umožňuje vyšší datovou propustnost oproti USB 3.0. Intel slibuje podporu až 16-ti video streamů na jedno zařízení, jednoduchou škálovatelnost a nízkou spotřebu.

Dalšími příklady VPU jsou Movidius Myriad 2, Pixel Visual Core, Microsoft HoloLens, Eyeriss a Nvidia PVA (Programmable Vision Accelerator).

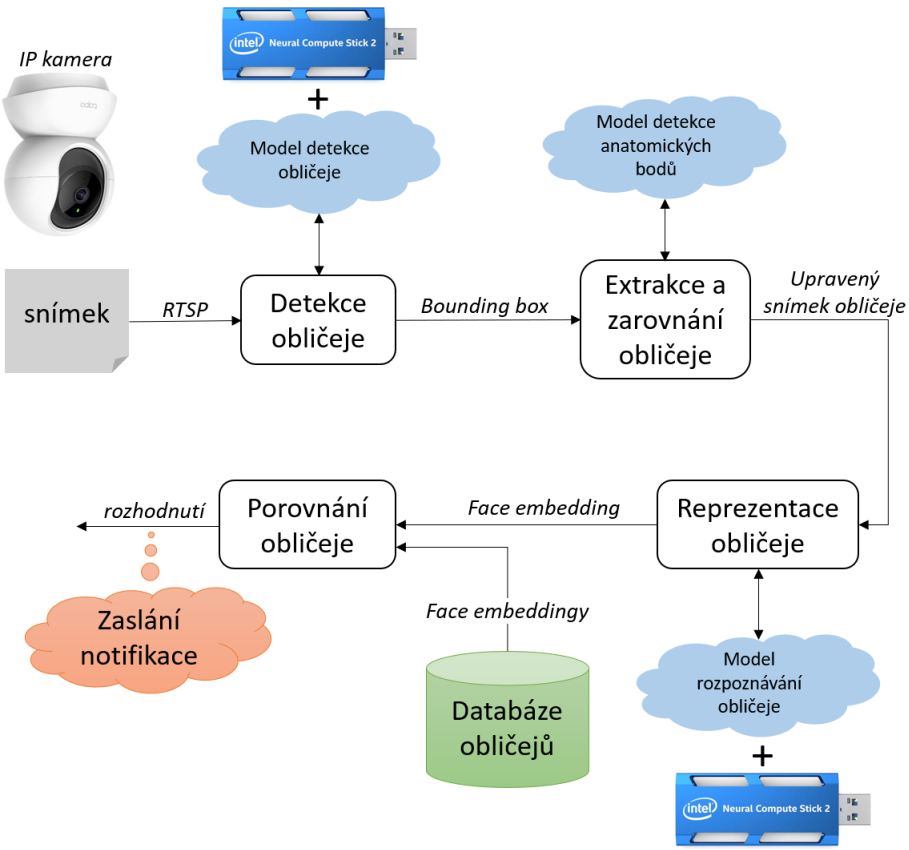
Dalšími populárními akcelerátory hlubokého učení jsou akcelerátory Jetson a Drive společnosti NVIDIA. Akcelerátory řady Jetson jsou kompletními systémy, které disponují GPU, CPU, managementem paměti a napájení, vysokorychlostními rozhraními atp. Akcelerátory řady Drive jsou škálovatelné počítače, uzpůsobené pro akceleraci autonomních automobilů.

Kapitola 5

Návrh a implementace

V této kapitole se budeme podrobněji zabývat návrhem a realizací backendu a frontendu aplikace. Kapitola detailně popisuje architekturu backendu, včetně použitých softwarových nástrojů a modelů neuronových sítí. Dále je věnována samotné realizaci uživatelského rozhraní spolu s použitým frameworkem.

5.1 Návrh backendu



Obrázek 5.1: Blokové schéma backendu.

Vstupem aplikace bude video, nebo adresa RTSP streamu. Výstupem aplikace bude notifikace zaslaná na mobilní zařízení/počítač, uložený snímek zachycené události a přehrávač video výstupu. Při použití RTSP streamu (viz sekce 4.2.4) jako vstupu, může být systém využit jako *Home Security* – střežící systém, kdy IP kamera může být umístěna ve venkovních i vnitřních prostorech, např. pro střežení okolí branky, před hlavními dveřmi, místností apod. Notifikace slouží pro informování uživatele při příchodu osoby, která není známá databázi osob. Cílem řešení je, aby aplikace běžela autonomně na nízkospotřebovém zařízení a umožňovala tak konstatní dohled nad střeženým okolím, popřípadě sloužila pro experimentování.

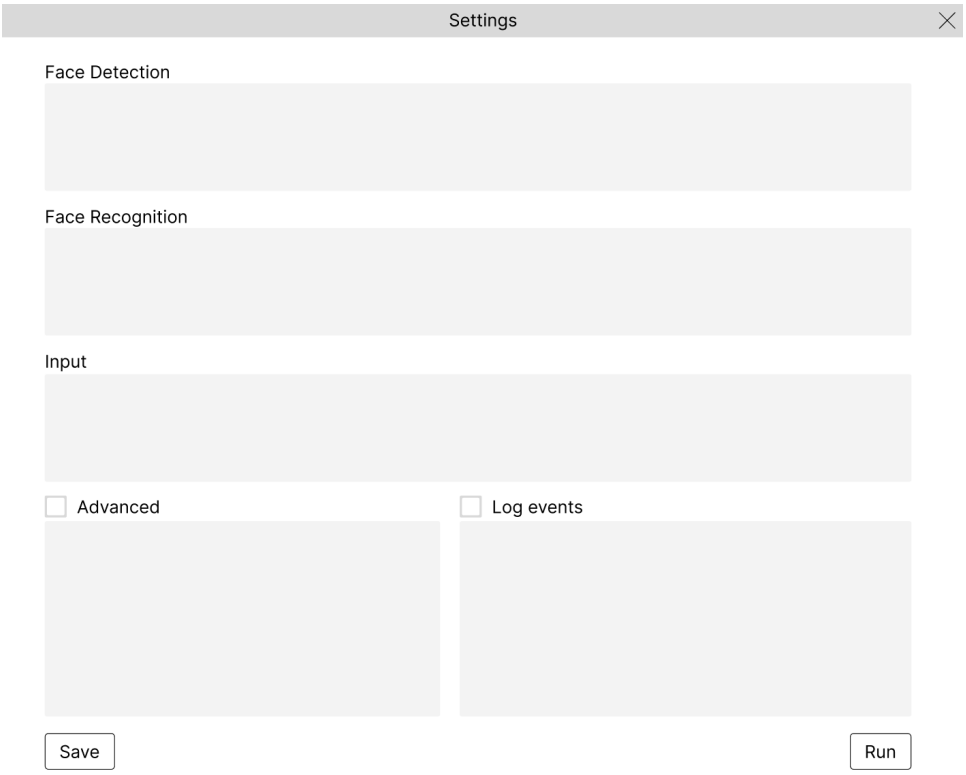
Systém (viz blokové schéma 5.1) nejprve detekuje obličej s použitím modelu pro detekci obličejů, jehož výstupem je procentní úspěšnost detekce a souřadnice *bounding boxu* ohraničující nalezený obličej. Pomocí výstupních souřadnic je ze vstupního snímku vyříznut obličej, který je zarovnán a postoupen modelu pro rozpoznávání obličeje. Výstupní *face embedding* je poté porovnán s *face embeddingy* snímků uložených v databázi obličejů a na základě *thresholdu* (hodnota, která určuje hranici mezi neznámým a známým obličejem) rozhodne, zda je obličej známý a patří některé z osob nacházející se v databázi obličejů, nebo se jedná o neznámý obličej.

Samotný proces rozpoznávání osob je velmi výpočetně náročný. Pro střežení oblasti je důležité, aby systém reagoval v reálném čase a uživatel dostal informaci o příchodu neznámé osoby včas. Za účelem zajištění této funkcionality je systém postaven na konceptu *Edge Computingu* (kapitola 4.3) a využívá dostupný akcelerátor *Intel Neural Compute Stick 2* (NCS2), přičemž bude možné akcelerovat jak detekci, tak rozpoznávání obličeje použitím dvou akcelerátorů NCS2.

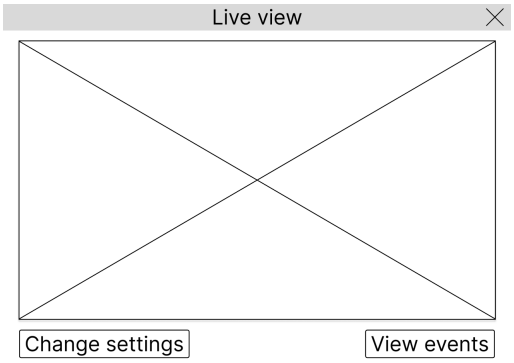
Aplikace je navržena tak, aby mohla být spuštěna s různými modely pro detekci a rozpoznání v konvertovaném modelu pomocí model optimizeru (mo), který je součástí OpenVINO Toolkitu (tj. modely *.xml). Zároveň uživateli umožňuje výběr akceleračního zařízení (vyvíjeno s podporou akcelerace na CPU, GPU a NCS2) podle možností uživatele. Uživatel má také možnost upravit minimální procentní úspěšnost detekce a práh rozpoznávání, pokud bude potřebovat zvýšit/snížit přesnost detekce a rozpoznávání. Pro možnost logování událostí, má uživatel možnost výběru, kam se budou snímky zaznamenaných událostí ukládat a maximální velikost, které může úložiště dosahovat.

5.2 Návrh frontendu

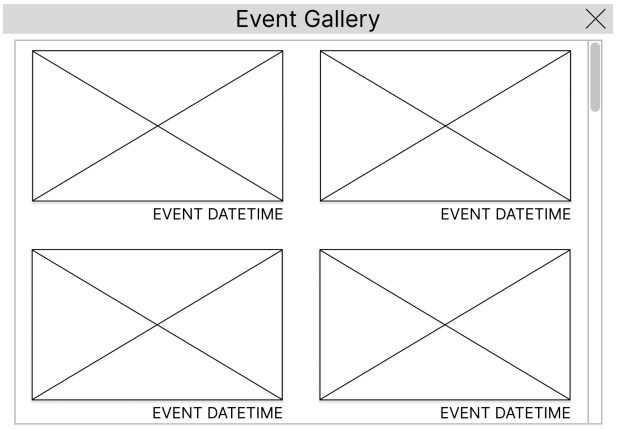
Pro uživatelskou přívětivost bude součástí řešení jednoduché grafické rozhraní, které zvládne i v problematice neznalého uživatele provést nastavením aplikace, zobrazením výstupu aplikace a možností zobrazení galerie již odehraných událostí. Aplikace se bude skládat z okna *Settings* – nastavení vstupů aplikace (obrázek 5.2), *Live view* – zobrazení video výstupu (obrázek 5.3) a *Event Gallery* – galerie archivovaných událostí (obrázek 5.4).



Obrázek 5.2: *Settings*. Okno nastavení je vstupní dialogové okno, které se bude zobrazovat po spuštění aplikace. Dialog se skládá z povinných částí pro nastavení detekce, rozpoznávání a video vstupu. Nabídka bude obsahovat také volitelné pokročilé (*Advanced*) nastavení *thresholdu*, pro detekci a rozpoznání, a volitelného nastavení pro logování událostí (*Log events*), které slouží k nastavení potřebných parametrů pro zálohování snímků událostí a zasílání notifikací. Dále bude možné uložit navolené nastavení pomocí tlačítka *Save* do souboru JSON. Tlačítko *Run* bude sloužit ke spuštění aplikace.



Obrázek 5.3: *Live view* okno bude obsahovat video výstup, kde bude zobrazován výstup rozpoznávání. Dále bude disponovat tlačítkem *Change settings* pro úpravu nastavení, po jehož stisku bude znovu zobrazeno okno *Settings* (obrázek 5.2). Tlačítko *View events* slouží pro zobrazení galerie událostí *Event gallery* (obrázek 5.4).



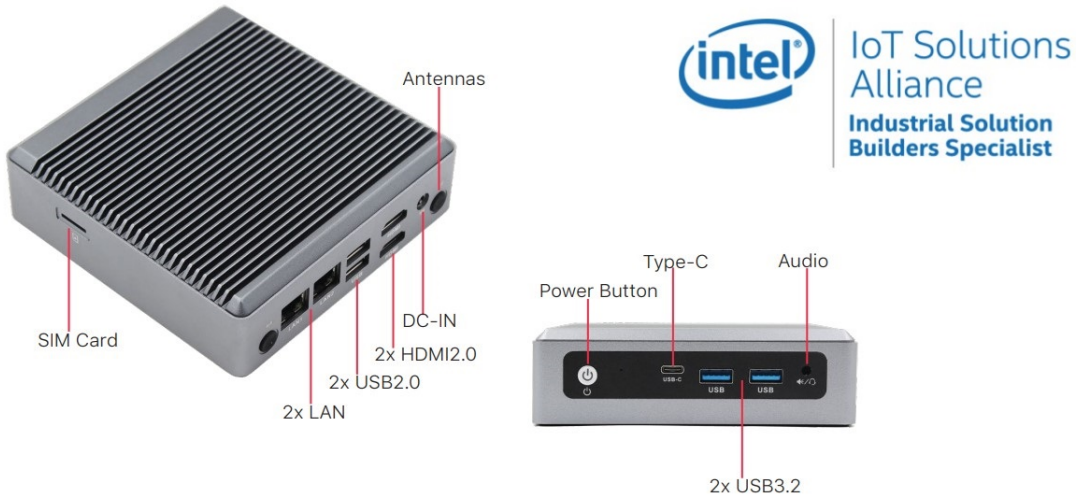
Obrázek 5.4: Okno *Event Gallery*. Galerie událostí bude zobrazovat události, seřazené od nejnovější po nejstarší. Každá událost bude označena datem a časem, kdy se udála.

5.3 Implementace backendu

Realizace backendu je klíčovou částí vývoje aplikace, která zahrnuje implementaci logiky a funkcí systému. Správná implementace backendu je nezbytná pro zajištění spolehlivého a bezpečného provozu celé aplikace. V této části kapitoly se budeme zabývat implementací aplikace, která je postavena na konceptu *edge computingu*. Věnovat se budeme konkrétnímu hardwarovému zařízení, využitým jako *edge device*, popisem realizace backendu včetně softwarových nástrojů, využitých pro implementaci logiky backendu.

5.3.1 Device on edge

Jako výpočetní zařízení na okraji sítě (*device on edge*) je využito zařízení Maxtang NX6412. Maxtang NX6412 [46], který je zobrazen na obrázku 5.5, je mini PC bez ventilátorů, který vyniká tichým provozem, kompaktním designem a malými rozměry $127 \times 127 \times 37$ mm. Disponuje procesorem Intel Elkart Lake procesorem J6412, Intel® UHD Graphics for 10th Gen Intel® GPU procesorem, podporuje dvoukanálový SO-DIMM DDR až do 32GB, připojení dvou nezávislých displejů přes 2xHDMI2.0. I/O rozhraní zahrnuje 2×LAN, 2×USB3.2 a 2×USB2.0. NX6412 má 1xM.2 2242/2280 SSD úložiště, SATA volitelné, vestavěné Wi-Fi a Bluetooth připojení a SIM slot pro rozšíření.



Obrázek 5.5: Maxtang NX6412 [46].

5.3.2 Softwarové prostředky

Python

Programovací jazyk Python[1] vznikl v roce 1991 jako open-source projekt, jehož autorem je Guido van Rossum. Jedná se o vysokoúrovňový programovací jazyk, který nabyl největší popularity v roce 2018 a zařadil se mezi nejoblibenější programovací jazyky. Tento jazyk nabízí instalační balíky pro platformy Unix, Windows, macOS nebo Android. Python je dynamický programovací jazyk, který podporuje různá programovací paradigmata – objektově orientované, imperativní nebo funkcionální.

[1] https://docs.python.org/3/license.html

OpenCV-Python

OpenCV[2] (Open Source Computer Vision Library) je open-source knihovna, která zahrnuje několik set algoritmů určených pro počítačové vidění. Tato knihovna začala vznikat s podporou společnosti Intel v roce 1999, první verze byla uveřejněna v roce 2000. Podporuje řadu programovacích jazyků, mezi které patří C++, Python a Java a je k dispozici na řadě platforem zahrnujících Windows, Linux, Android, iOS a OS X.

OpenCV-Python je Python API, kombinující OpenCV C++ API s jazykem Python. Tato kombinace umožňuje zachovat rychlost C/C++ kódu a zároveň jednoduchost, pro kterou je tolik oblíben jazyk Python. OpenCV-Python tedy nabízí tzv. wrapper kolem originální OpenCV C++ implementace. OpenCV-Python také využívá pythonovského modulu Numpy, což je vysoce optimalizovaná knihovna pro číslicové operace se syntaxí podobnou platformě MATLAB. OpenCV-Python proto konvertuje všechna pole z a do polí modulu Numpy a tím zajišťuje jednoduchost integrace s dalšími knihovnami, využívající Numpy jako např. SciPy nebo Matplotlib.

OpenVINO Runtime

OpenVINO Runtime [44] je softwareový framework, který poskytuje běhové prostředí pro aplikace využívající neuronové sítě a je určen pro nasazení těchto aplikací na hardware. Byl vyvinut společností Intel a umožňuje vývojářům optimalizovat a nasazovat aplikace s použitím neuronových sítí na široké spektrum hardwarových platforem, což vede ke zlepšení výkonu a rychlosti zpracování neuronových sítí. Díky tomu mohou aplikace používající OpenVINO runtime zpracovávat velké objemy dat v reálném čase, což je klíčové pro mnoho aplikací v oblasti umělé inteligence, jako jsou například rozpoznávání obrazu a zpracování řeči.

Oproti tomu OpenVINO-dev je softwarový vývojový kit, který umožňuje vytváření a ladění neuronových sítí. Je určen především pro vývojáře a vědecké pracovníky, kteří se zabývají vývojem nových algoritmů pro neuronové sítě a jejich optimalizací.

5.3.3 Použité modely

Detekce obličeje

Obličeje jsou detekovány modelem **face-detection-retail-0004** [47]. Tento předtrénovaný model od společnosti Intel je založen na konvoluční neuronové síti a využívá algoritmus Single Shot MultiBox Detector (SSD) pro detekci obličejů v obraze. Backbone tohoto modelu je založen na architektuře SqueezeNet Light, která obsahuje moduly *fire*, které snižují počet výpočtů. Hlava SSD z funkční mapy s měřítkem 1/16 obsahuje devět shlukovaných předchozích boxů.
Specifikace modelu jsou:

- GFlops: 1,067
- MParams: 0,588
- Framework: Caffe

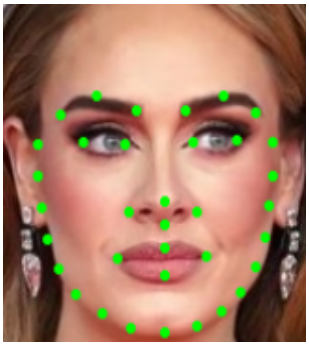
Výstupem modelu je pole sedmi hodnot ve formátu [image_id, label, confidence, x_min, y_min, x_max, y_max], kde *image_id* je identifikátor obličeje v sadě, *label* je predikovaný

[2] https://docs.opencv.org/4.x/d0/de3/tutorial_py_intro.html

identifikátor třídy (1 = obličej), *conf* je důvěryhodnost detekce a [x_min, y_min, x_max, y_max] jsou souřednice obdélníku, ohraničující nalezený obličej.

Model byl trénován na různých datasetech, aby byl schopen rozpoznávat obličeje v různých podmínkách osvětlení, pozic a velikostí, je tedy vhodný pro detekci ve venkovních i vnitřních prostorech.

Detekce anatomických bodů obličeje



Obrázek 5.6: Výstup modelu **facial-landmarks-35-adas-0002**.

Anatomické body obličeje jsou detekovány předtrénovaným modelem společnosti Intel **facial-landmarks-35-adas-0002** [48], který je postaven na customizované konvoluční neuronové síti, která detekuje polohu 35 významných bodů na obličeji. Výstupem modelu je pole 70 hodnot, které specifikují 35 predikovaných bodů obličeje. Specifikace modelu jsou:

- GFlops: 0,042
- MParams: 4,595
- Framework: Caffe

Model byl validován na náhodném podvýběru o velikosti 1000 vzorků z velkého datasetu společnosti Intel, který obsahuje obrázky 300 lidí s různými výrazy obličeje.

Rozpoznávání obličeje

Model architektury **ArcFace** (3.3.3) **face-recognition-resnet100-arcface-onnx** [49] je model postavený na ResNet100 a využívá ztrátovou funkci ArcFace. Tento model je předtrénovaný v MXNet frameworku a je konvertován do ONNX formátu. Model byl trénován na MS-Celeb-1M datasetu (3.1.1).
Specifikace modelu jsou:

- GFlops: 24,2115
- MParams: 65,1320
- Framework: MXNet

Dalším modelem použitým pro rozpoznávání je model **Sphereface** (3.3.2) [50]. Tento model je postaven na 20 vrstvé konvoluční neuronové síti a byl trénován na datasetu CASIA-WebFace (3.1.3).
Specifikace modelu jsou:

- GFlops: 3,504
- MParams: 22,671
- Framework: Caffe

Posledním použitým modelem pro rozpoznávání obličejů je model architektury **FaceNet** (3.3.1) **20180402-114759** [51]. Backbone tohoto modelu je Inception Resnet V1, který

kombinuje bloky Inception, které umožňují efektivní zpracování obrazových dat různých velikostí, a ResNet. Model byl trénován na datasetu VGGFace2 (3.1.2).
Specifikace modelu jsou:

- GFlops: 2,846
- MParams: 23,469
- Framework: Tensorflow

Všechny modely byly konvertovány pomocí model optimizeru, který je součástí OpenVINO Toolkitu, do formátu *.xml. Díky této konverzi byly modely upraveny tak, aby jejich vstupní barevný formát odpovídal formátu, který podporuje OpenCV a byly transformovány do datového typu FP16, který je podporován NCS2.

5.3.4 Implementace

Aplikace je implementovaná v jazyce Python s použitím knihovny pro počítačové vidění OpenCV a OpenVINO runtime (viz kapitola 5.3.2).

Program při každém spuštění načítá galerii obličejů, kdy pro jednotlivé obličeje v databázi vytváří pomocí modelu pro rozpoznávání obličejů *face embeddings*, které si ukládá. Před zahájením procesu rozpoznávání si alokuje akcelerační zařízení a zkompiluje uživatelem specifikované modely. Takové modely jsou optimalizované a mohou být rychleji a efektivněji spuštěny na alokovaných zařízeních, což vede ke zrychlení výpočtů a zlepšení výkonu aplikace. Poté je spuštěn proces rozpoznávání.

Každý snímek obrazového vstupu je postoupen modelu na detekci obličejů. Pro nahrání snímku na akcelerační zařízení, inferenci a získání výsledků detekce je použita třída OpenVINO Python API `InferRequest`. Výstupem detekce jsou *bounding boxy* ohraničující nalezené obličeje (viz obrázek 5.7.a).

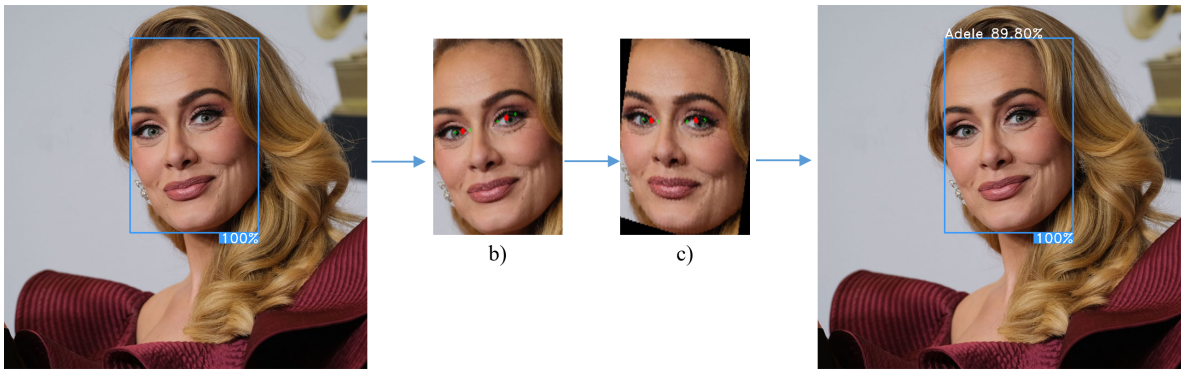
Nalezené obličeje jsou ze snímku vyříznuty a je na ně aplikována detekce anatomických bodů obličeje. Z detekovaných bodů jsou vybrány body $L_0[x_0, y_0]$, $L_1[x_1, y_1]$, které ohraničují levé oko a $P_0[x_2, y_2]$, $P_1[x_3, y_3]$, ohraničující oko pravé. Na základě těchto bodů je spočítán střed oka, jako středu úsečky L_0L_1 a P_0P_1, $L[\frac{(x_0+x_1)}{2}, \frac{(y_0+y_1)}{2}]$ a $P[\frac{(x_2+x_3)}{2}, \frac{(y_2+y_3)}{2}]$, které představují kotevní body obličeje (viz obrázek 5.7.b). Poté je obličej zarovnán metodou `alignment_procedure`, která je převzata z [52]. Tato metoda provádí následující kroky:

1. Na základě kotevních bodů zjistí směr, ve kterém má být snímek obličeje otočen a nalezne souřadnice třetího bodu v úrovni očí za vzniku pravoúhlého trojúhelníku mezi očima.
2. S použitím euklidovské vzdálenosti (viz kapitola 2.1.5) vypočítá strany vzniklého pravoúhlého trojúhelníku.
3. Pomocí kosinova pravidla vypočítá úhel, o který má být snímek obličeje otočen.
4. Otočí snímek obličeje.

Výsledek této metody je zobrazen na obrázku 5.7.c.

Jednotlivé zarovnané snímky obličejů jsou vloženy do asynchronní inferenční fronty `AsyncInferQueue`, která je součástí OpenVINO Python API. Tato třída slouží k asynchronnímu zpracování inferencí neuronových sítí, poskytuje metody pro přidání požadavku

na inferenci a získání výsledků inferencí v asynchronním režimu, což znamená, že požadavky na inferenci se zpracovávají v pozadí a výsledky jsou k dispozici, jakmile jsou dostupné. `AsyncInferQueue` umožňuje výrazně zlepšit výkon aplikace, zejména v případech, kdy se používají modely s vysokou výpočetní náročností. Třída `AsyncInferQueue` také poskytuje možnost konfigurace požadovaného počtu požadavků na inferenci, a díky tomu umožňuje optimalizovat výkon aplikace v závislosti na hardwaru a dostupné výpočetní kapacitě. Výstupem rozpoznávání každého obličeje je *face embedding*, který je porovnán s každým *face embeddingem* databáze osob pomocí kosinové vzdálenosti *cosine distance* (viz kapitola 2.1.5).



Obrázek 5.7: Proces rozpoznávání. a) detekce obličeje, b) vyříznutí obličeje ze snímku a nalezení kotevních bodů detekovaného obličeje, c) zarovnání obličeje pomocí kotevních bodů, d) rozpoznání podle obličeje a zobrazení výsledků

Výsledky kosinové vzdálenosti jsou porovnány metodou minimálního skóre, kdy za výslednou identitu je považován ten výsledek, který má hodnotu kosinové vzdálenosti nejmenší. Pokud je tato hodnota menší nebo rovna hodnotě *threshold* rozpoznávání obličeje, je osoba identifikována a je systémem považovaná za známou. Pokud hodnota přesahuje *threshold*, je klasifikována jako neznámá (UNKNOWN). Výstup aplikace je zobrazen na obrázku 5.7.d. Událost (*event*) nastává ve chvíli, kdy se na x snímcích za sebou vyskytne neznámá osoba (Tato hodnota může být upravena v nastavení v sekci *Advanced* 5.4.2). Poté je událost logována a je uživateli zaslána notifikace (viz kapitola 5.3.5) o výskytu neznámé osoby ve střeženém prostoru.

5.3.5 Zasílání notifikací

Zasílání notifikací je implementováno pomocí služby Pushover[3], která zasílá upozornění na mobilní zařízení pomocí stejnojmenné aplikace Pushover. Tato aplikace umožňuje vývojářům integrovat notifikační funkce do svých aplikací a webových služeb, aby mohli odesílat oznámení uživatelům, a je multiplatformní.

Pushover používá jednoduché, verzované REST API[4] pro přijímání zpráv a jejich rozesílání na zařízení, na kterých běží klientská aplikace. Pro zasílání notifikací pomocí Pushover se uživatel musí zaregistrovat na službu Pushover a získat API klíč, který je použit v aplikaci nebo webové službě pro odesílání notifikací. Následně si uživatel musí nainstalovat Pushover aplikaci na své mobilní zařízení a přidat toto zařízení do svého účtu.

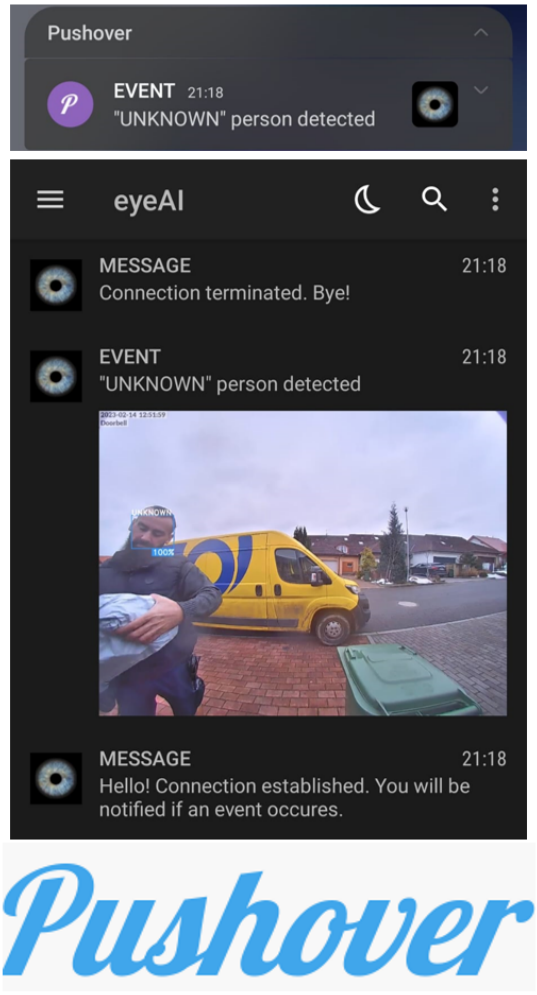
[3] https://pushover.net/
[4] https://pushover.net/api

Notifikace jsou odesílány HTTP metodou POST na API endpoint Pushoveru. Parametry pro odeslání notifikace přes Pushover API jsou:

- **token** – API klíč, který uživatel získal po registraci služby Pushover.
- **user** – uživatel, kterému má být notifikace odeslána
- **message** – text zprávy
- a další volitelné parametry...

Po odeslání požadavku na API endpoint Pushoveru se notifikace odešle na všechna zařízení přidružená k uživateli. Uživatelé dostanou notifikaci prostřednictvím Pushover aplikace nebo pomocí push notifikací na svých mobilních zařízeních.

Uživatel je také prostřednictvím těchto notifikací informován o tom, že aplikace byla spuštěna a byla připojena k notifikačnímu systému a naopak, kdy je ukončena a odpojena od notifikačního systému. Tato funkcionalita informuje uživatele o stavu aplikace, aniž by se musel na zařízení připojovat a kontrolovat stav.



Obrázek 5.8: Notifikace zaslaná prostřednictvím Pushover API registrovanému uživateli.

5.4 Implementace frontendu

V této části kapitoly se budeme zabývat použitým frameworkem na implementaci uživatelského rozhraní a detailním popisem výsledného rozhraní.

5.4.1 Softwarové prostředky

PyQt5

PyQt5[5] je vysokoúrovňové Python API pro knihovnu Qt5, která je multiplatformním nástrojem pro tvorbu desktopových aplikací s grafickým uživatelským rozhraním. PyQt5 je open-source projekt s licencí GPL a komerční licencí. PyQt5 je jedním z nejpopulárnějších nástrojů pro tvorbu desktopových aplikací v Pythonu. Díky své flexibilitě a možnostem, které poskytuje, je ideální pro tvorbu aplikací s náročným uživatelským rozhraním.

PyQt5 poskytuje rozhraní pro všechny klíčové funkce Qt5, včetně zpracování a zobrazení videa. Poskytuje snadný způsob, jak zobrazit video v GUI aplikacích. Díky možnostem, které poskytuje, je ideální pro aplikace, které vyžadují rychlé zpracování videa, jako jsou například video-editory nebo aplikace pro analýzu obrazu. PyQt5 je také multiplatformní a lze ho použít na různých operačních systémech, jako jsou Windows, Mac nebo Linux.

5.4.2 Implementace

Aplikace po svém spuštění zobrazí dialogové okno nastavení *Settings* pro zadání vstupních parametrů programu. Grafické rozhraní je rozděleno na několik částí, *Face Detection* – pro detekci, *Face Recognition* – pro rozpoznávání, *Input* – specifikaci vstupu a databázi obličejů, volitelný oddíl *Advanced* – úprava pokročilých parametrů programu, a *Log event* – povolení archivace událostí, zasílání oznámení a parametrů potřebných pro zasílání notifikací.

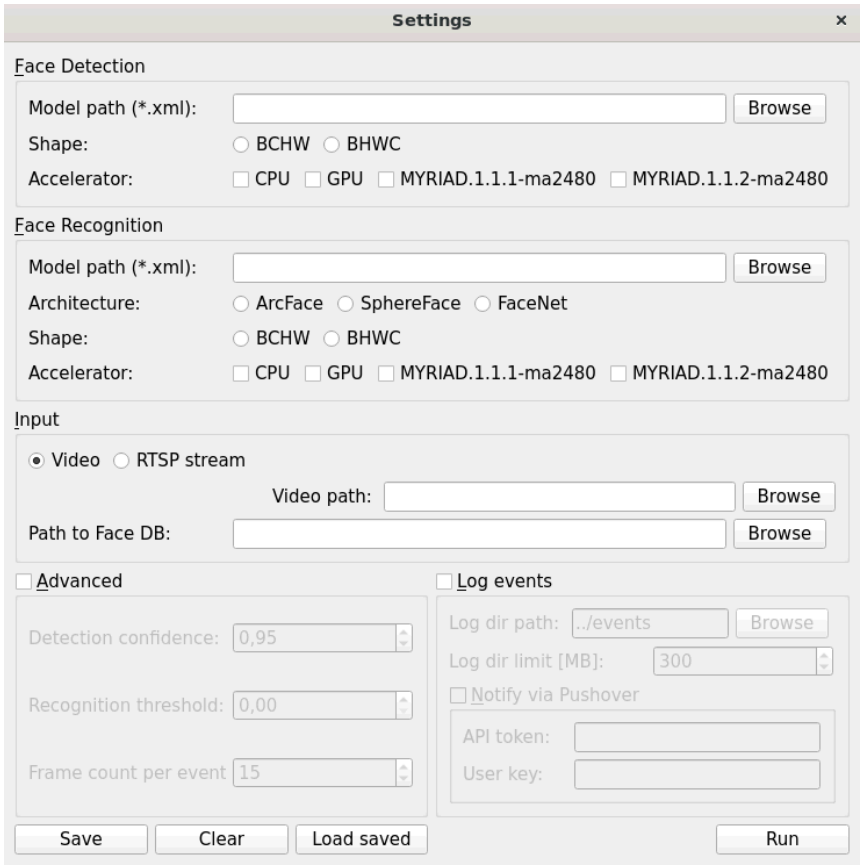
Face Detection & Face Recognition

Aplikace je implementována tak, aby byla použitelná s různými modely detekce a rozpoznávání v konvertovaném OpenVINO formátu (*.xml). Uživatel tedy specifikuje cestu ke konvertovanému modelu. V případě volby modelu pro rozpoznávání uživatel volí architekturu modelu, který zvolil pro rozpoznání. Tento výběr slouží k nastavení výchozího optimálního prahu pro rozpoznání, který lze dále upravit v sekci *Advanced*. Dalším vstupním parametrem je způsob uspořádání dat vstupního modelu (většinou platí pro model frameworku tensorflow tvar BHWC a PyTorch tvar BCHW), aby mohl být použit libovolný model. Dále uživatel specifikuje akcelerační zařízení pro danou úlohu, kdy dostupná akcelerační zařízení jsou detekována knihovnou OpenVINO. Platí, že akcelerační zařízení typu MYRIAD (Intel Neural Compute Stick 2) může být využito buď pro detekci, nebo rozpoznávání, a tedy pro akceleraci obou úloh tímto zařízením musí být dostupná zařízení MYRIAD dvě.

Input

Obrazovým vstupem pro aplikaci může být video, uložené na zařízení, nebo URL adresa RTSP streamu (viz kapitola 4.2.4). V případě využití možnosti RTSP streamu může aplikace sloužit na principu *Home Security* systému. Uživatel dále specifikuje cestu ke galerii osob, která systému specifikuje známé osoby. Galerie osob je adresářový systém, kdy každá osoba má svůj adresář označený jménem a uvnitř svoje fotografie.

[5] https://www.riverbankcomputing.com/static/Docs/PyQt5/introduction.html#license



Obrázek 5.9: Okno *Settings* výsledné aplikace.

Advanced

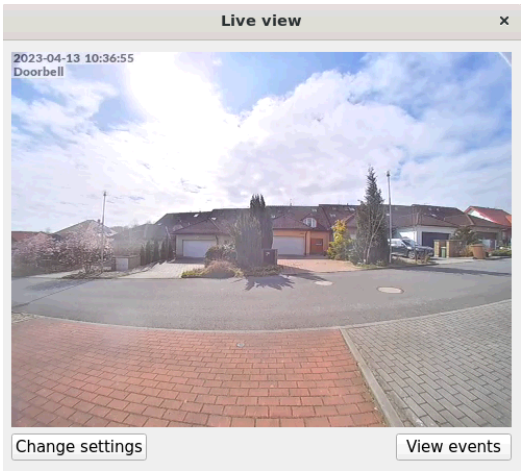
Tento oddíl specifikuje pokročilé nastavení aplikace, konkrétně umožňuje uživateli modifikovat jistotu detekce a práh rozpoznávání pro snížení/zvýšení přesnosti. Práh rozpoznávání je předvolen na základě výběru architektury modelu v sekci *Face Recognition*, které byly naměřeny v rámci validací jednotlivých modelů (viz kapitola 6.1), a může být dále upraven podle potřeb uživatele. Tento oddíl je potřebný, protože každý detekční/rozpoznávací model dosahuje různých přesností a aplikace by při statických hodnotách *thresholdů* mohla selhávat. Uživatel tak nemusí modifikovat kód, ale nastaví tyto hodnoty v grafickém rozhraní. Tento oddíl je volitelný a uživatel jej nastavuje pouze v případě, pokud není spokojen s výchozími hodnotami.

Log event

Tato sekce umožňuje povolit uživateli archivaci událostí do specifikovaného adresáře a limitovat velikost archivu. Uživatel může povolit zasílání notifikací přes aplikaci Pushover, musí však specifikovat identifikátory použité pro zasílání notifikací přes aplikaci Pushover registrovanému uživateli (viz kapitola 5.3.5).

Dialogové okno *Settings* umožňuje uložit na zařízení hotové nastavení ve formátu JSON a naopak také načíst uložené nastavení ve formátu JSON do dialogového okna. Aplikace také

poskytuje uživateli tlačítko na vyčistění dialogového okna. Po specifikaci všech vstupních polí je stisknutím tlačítka Run spuštěn program.



Obrázek 5.10: Okno *Live View* výsledné aplikace.

Obrázek 5.11: Okno *Event Gallery* výsledné aplikace.

Výsledek rozpoznávání je uživateli zobrazen v okně *Live view* (obrázek 5.10), kde uživatel může v reálném čase sledovat výstup aplikace. Toto okno obsahuje tlačítko *View events*, díky kterému může uživatel zobrazit galerii logovaných událostí *Event Gallery* (obrázek 5.11), které jsou seřazeny sestupně a jsou označeny datem a časem vzniku události. Okno *Live view* ještě disponuje tlačítkem *Change settings* pro úpravu nastavení, po jehož stisku bude znovu zobrazeno okno *Settings*.

Kapitola 6

Experimenty

V bezpečnostních systémech je kladen požadavek na *real-time* odezvu, proto je důležité najít rovnováhu mezi rychlostí rozpoznávání a přesností modelů. Tato kapitola se zabývá výsledky experimentů, které byly provedeny pro zhodnocení přesnosti použitých modelů pro rozpoznávání obličejů a rychlosti zpracování obrazových dat na CPU, GPU a NCS2 pro různý počet osob v obraze a s různými modely pro rozpoznávání tváří. Cílem experimentů bylo zjistit, jaké přesnosti jednotlivé modely nabývají, jakých výsledků dosahuje akcelerace systému pomocí CPU, GPU a NCS2, jaká konfigurace je nejvhodnější pro jednotlivé modely a která konfigurace dosahuje nejvyšší snímkové frekvence a nejnižší latence mezi porovnávanými modely. Pro splnění *real-time* požadavku na konfiguraci systému je stanovena hranice snímkové frekvence na 15 FPS, které dosahuje RTSP stream[1].

6.1 Validace modelů a stanovení optimálních prahů

Použité modely mají různou architekturu a byly trénovány na různých datasetech, proto dosahují různých přesností. Pro správnou klasifikaci obličeje je důležitá optimální prahová hodnota (*threshold*), která stanovuje hranici, kdy je obličej pro systém známý a kdy nikoliv. Optimální prahová hodnota představuje práh, při kterém model klasifikuje správně co nejvíce pozitivních i negativních případů. Práh je pro každý model specifický a závisí také na přesnosti, jaké model dosahuje. Cílem validace je změřit schopnosti jednotlivých modelů za stejných podmínek a určit co nejoptimálnější prahovou hodnotu pro každý model. Tyto zjištěné optimální prahové hodnoty budou nastaveny jako výchozí pro jednotlivé architektury v aplikaci viz sekce 5.4.2.

V bezpečnostním systému, který klasifikuje osoby na základě přístupné databáze osob, představují binární klasifikační pojmy následující situace:

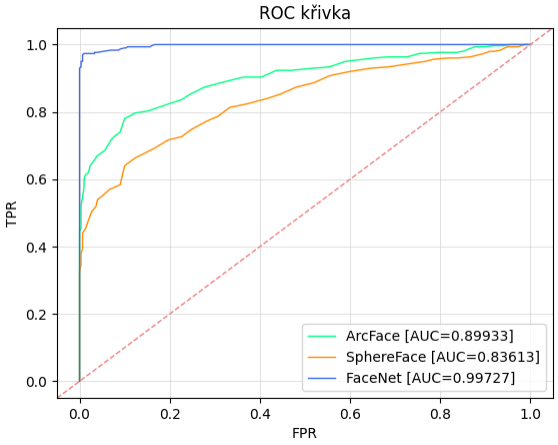
- TP – správně identifikována osoba, která patří do databáze obličejů
- TN – správně odhalena osoba, která nepatří do databáze obličejů
- FP – neznámá osoba je identifikována jako známá
- FN – známá osoba je označena jako neznámá

Jako validační dataset byl použit dataset LFW (kapitola 3.1.4), konkrétně jeho validační forma *10_folds*, která se skládá z náhodně vygenerovaných 10-ti setů dvojic snímků obličeje

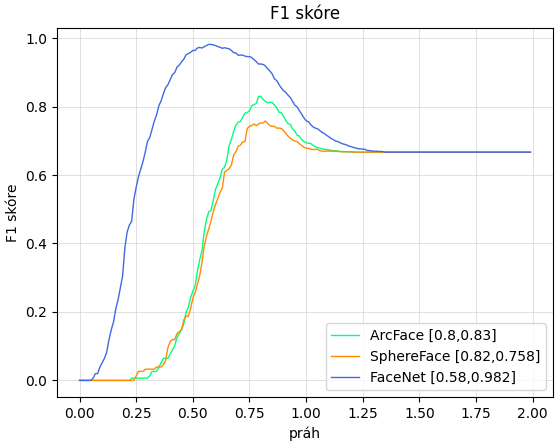
[1] https://www.tapo.com/en/product/smart-camera/tapo-c200/#tapo-product-spec

LFW. Každý set obsahuje 300 pozitivních dvojic (dvojice patří stejné osobě) a 300 negativních dvojic snímků (dvojice nepatří stejné osobě). Celkem testovací sada obsahuje 6000 dvojic snímků obličeje a jedná se o třídně vyvážený dataset. Metriky a pojmy, které byly použity pro validaci a stanovení prahů jsou popsány v kapitole 2.1.6, konkrétně se jedná o přesnost (*accuracy*), preciznost (*precision*), specificitu (*specificity*), senzitivitu (*recall*), F1 skóre a hodnoty AUC a AP. Pro validaci byl použit vlastní testovací skript, který každý obličej z dvojice postoupí validovanému modelu a pro výstupní dvojici *face embeddingů* vypočítá kosinovu vzdálenost. Pokud je vzdálenost menší, nebo rovna aktuálnímu prahu, predikuje obličeje jako patřící stejné osobě, jinak nikoliv. Predikovaný výsledek porovná s očekávaným výstupem a výsledek porovnání zařadí do příslušné kategorie TP, TN, FP, nebo FN. Celý proces opakuje pro každou hodnotu prahu z intervalu <0,2> s krokem 0,01.

Výsledky validace ukazují grafy, které zobrazují ROC křivku a AUC 6.1, F1 skóre 6.2, preciznost A.1, senzitivitu A.2, PR křivku a AP A.3 a přesnost A.4 validovaných modelů pro různé hodnoty prahu.

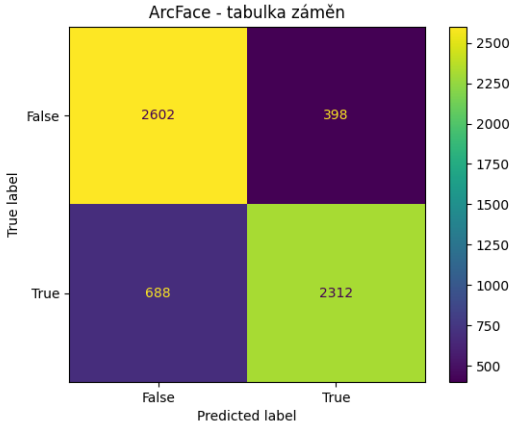


Obrázek 6.1: Graf zobrazující ROC křivky jednotlivých modelů společně s hodnotami AUC.

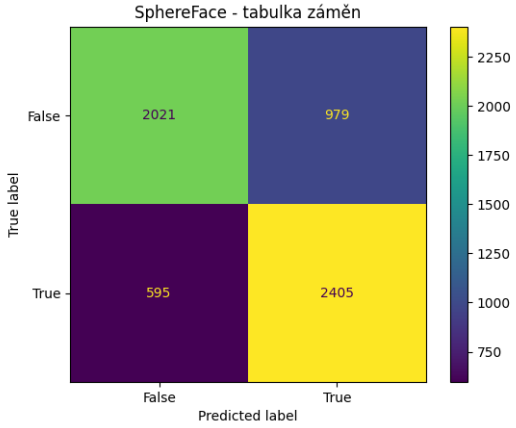


Obrázek 6.2: Graf zobrazující vývoj F1 skóre jednotlivých modelů v závislosti na rostoucím prahu. Nejvyšší F1 skóre modelu **ArcFace** nabývá hodnoty 0,830, při hodnotě prahu 0,8. Model **SphereFace** dosáhl nejvyšší hodnoty F1 skóre pro práh 0,82, a to 0,758. Třetí model **FaceNet** dosáhl nejvyššího F1 skóre 0,982 ze všech porovnávaných modelů při hodnotě prahu 0,58.

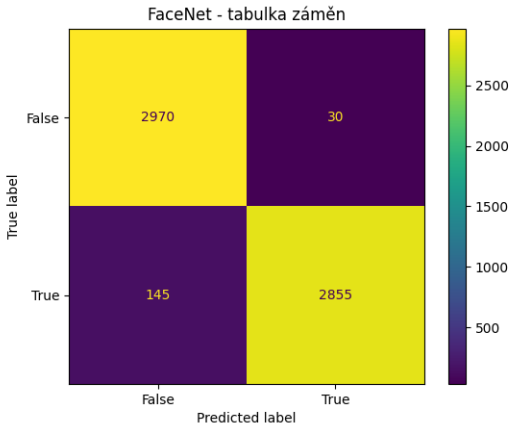
Na základě výsledků F1 skóre byl pro každý model určen optimální práh, který byl měřen s přesností na dvě desetinná místa. Grafy tabulek záměn 6.3, 6.4, 6.5 zobrazují hodnoty TP, TN, FP a FN pro optimální hodnotu prahu. Na základě hodnot v tabulce záměn byla vypočítána finální dosažená přesnost, specificita, senzitivita a preciznost modelu, které jsou uvedeny pod příslušnými grafy.



Obrázek 6.3: Tabulka záměn modelu ArcFace pro optimální práh 0,80. Na základě těchto hodnot dosahuje model přesnosti 81,9 %, specificity 86,73 %, senzitivity 77,07 % a preciznosti 85,31 %.



Obrázek 6.4: Tabulka záměn modelu SphereFace pro optimální práh 0,82. Na základě těchto hodnot dosahuje model přesnosti 73,77 %, specificity 67,36 %, senzitivity 80,17 % a preciznosti 71,07 %.



Obrázek 6.5: Tabulka záměn modelu FaceNet pro optimální práh 0,58. Na základě těchto hodnot dosahuje model přesnosti 97,08 %, specificity 99 %, senzitivity 95,17 % a preciznosti 98,96 %.

Z experimentů vyplynulo, že model FaceNet výrazně překonal ostatní modely ve všech porovnávaných metrikách. Graf ROC křivky modelu se nejvíce přiblížil levému hornímu rohu grafu s hodnotou AUC 0,99727. Dosáhl také nejvyšší přesnosti a to 97,08 %, ale také specificity, senzitivity a preciznosti. Na základě dosažených výsledků validace jej můžeme považovat za nejpřesnější z porovnávaných modelů. Výkonnostní rozdíl mezi jednotlivými algoritmy je způsoben předzpracováním, kdy použité předzpracování vyhovuje modelu FaceNet ale nevyhovuje ostatním algoritmům.

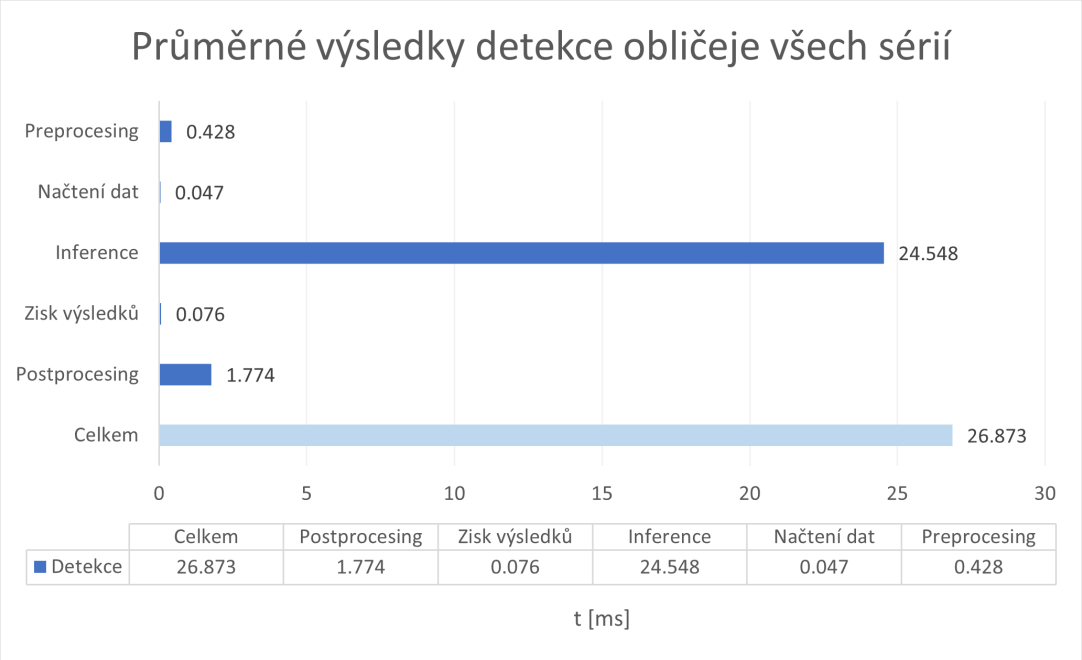
6.2 Výsledky akcelerace pro různé konfigurace systému

V rámci experimentování bylo provedeno celkem šest sérií experimentů s třemi modely pro rozpoznávání obličejů a dvěmi různými videi, které obsahují jednu, nebo dvě osoby. Každá série experimentů se skládá z detekce, akcelerované pomocí Intel NCS2, a rozpoznávání, které bylo akcelerováno postupně pomocí CPU, GPU a NCS2, jednoho typu videa. Rozlišení videí, se kterými byly experimenty provedeny, je 640 × 360. Experimenty byly provedeny a vyhodnoceny na zařízení Maxtang NX6412 (viz kapitola 5.3.1) s operačním systémem Ubuntu 20.04 LTS a OpenVINO verze 2022.1.0.

6.2.1 Výsledky detekce obličeje

Detekce obličeje byla prováděna s použitím modelu **face-detection-retail-0004**, přičemž byla akcelerována pomocí NCS2. Dosažené výsledky (graf 6.6) reprezentují průměrnou dobu zpracování jednoho snímku obrazového vstupu ze všech provedených šesti sérií experimentů. V rámci experimentů byly měřeny hodnoty:

- *Preprocesing* se skládá z přípravy snímku do formátu, požadovaného detekčním modelem (rozšíření dimenze snímku a převod na tensor).
- *Načtení dat* měří dobu načtení snímku na akcelerační zařízení.
- *Inference* měří dobu samotné inference.
- *Zisk výsledků* měří dobu získání výsledků z akceleračního zařízení.
- *Postprocesing* měří dobu evaluace výsledků detekce obličeje.
- *Celkem* sumarizuje předchozích pět časových úseků a představuje dobu od zisku vstupního snímku po stanovení výsledků detekce.



Obrázek 6.6: Graf výsledků detekce obličeje s použitím akcelerátoru *Intel Neural Compute Stick 2*. Výsledky jsou průměrem ze všech provedených šesti sérií experimentů a představují dobu zpracování jednoho snímku obrazového vstupu. Výsledky jsou uvedeny v milisekundách.

Z provedených experimentů vyplynulo, že detekování obličejů v obraze použitým detekčním modelem trvá průměrně 26,873 ms. Preprocesing vstupu představuje 1,59 % trvání detekce, načtení dat 0,17 % trvání detekce, samotná inference tvoří 91,35 % trvání detekce, zisk výsledků 0,28 % a postprocesing výsledků 6,6 % trvání detekce.

6.2.2 Výsledky rozpoznávání obličeje

Pro experimentování byly využity modely pro rozpoznávání obličejů architektury ArcFace, SphereFace a FaceNet, které jsou popsány v části kapitoly 5.3.3. V rámci experimentů bylo měřeno sedm časových úseků:

- *Preprocesing* se skládá ze zarovnání obličeje na snímku a přípravy snímku do formátu, požadovaného rozpoznávacím modelem (změna velikosti, rozšíření dimenze snímku a převod na tensor).
- *Načtení dat* měří dobu načtení snímku na akcelerační zařízení.
- *Inference* měří dobu samotné inference.
- *Zisk výsledků* měří dobu získání výsledků z akceleračního zařízení.
- *Postprocesing* měří dobu evaluace výsledků rozpoznávání.
- Hodnota *celkem* sumarizuje předchozích pět časových úseků a představuje dobu od detekce obličeje v obraze po stanovení výsledků klasifikace obličeje.
- *Latence* představuje dobu zpracování jednoho snímku, která zahrnuje načtení snímku, detekci, rozpoznávání obličeje, až po zobrazení snímku na obrazovce.

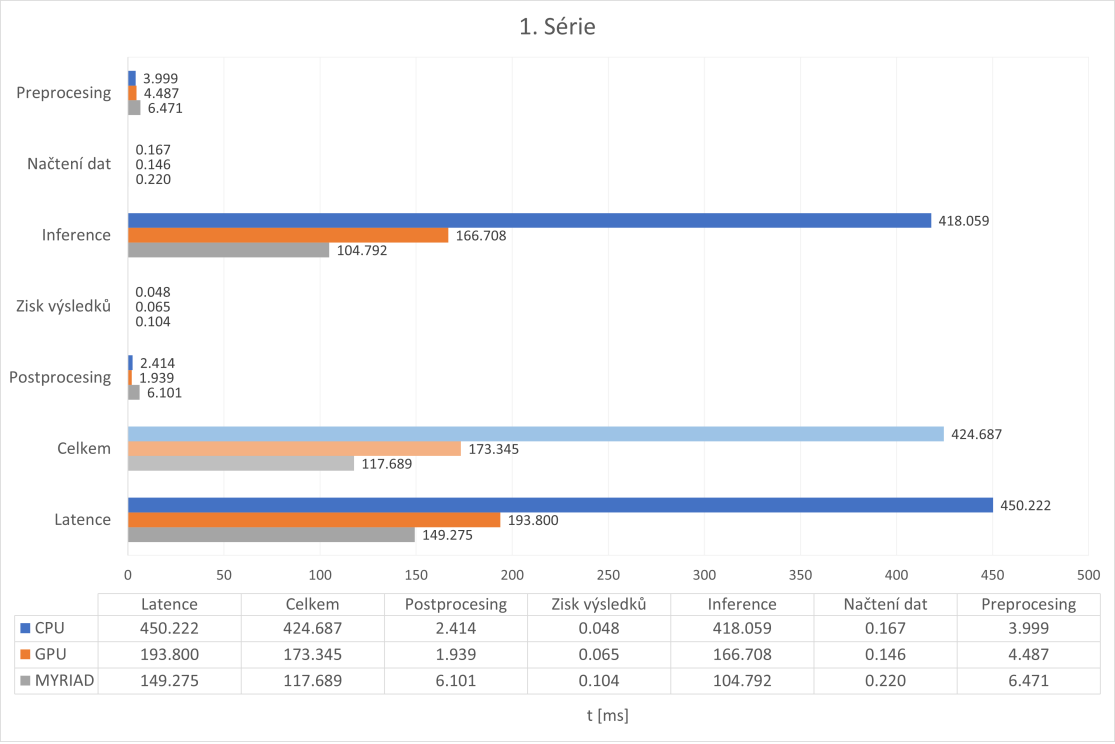
Výsledky experimentů zobrazují grafy 6.7, 6.8, 6.9, 6.10, 6.11, 6.12. Jednotlivé výsledky jsou zprůměrovány a představují čas zpracování jednoho snímku videa.

Výsledky experimentů první série 6.7 ukazují, že doba rozpoznání jednoho snímku videa s jednou osobou dosahuje na CPU hodnoty 424,687 ms. Akcelerací pomocí GPU došlo k 2,45 násobnému zrychlení oproti CPU na hodnotu 173,345 ms a akcelerací pomocí NCS2 k 3,61 násobnému zrychlení oproti CPU na hodnotu 117,689 ms. Výsledky druhé série 6.8 ukazují, že doba rozpoznání jednoho snímku videa se dvěmi osobami dosahují na CPU hodnoty 931,574 ms – 2,19 násobné hodnoty oproti první sérii, na GPU 405,165 ms – 2,34 násobné hodnoty oproti první sérii, 2,3 násobnému zrychlení oproti CPU a na NCS2 157,945 ms – 1,34 násobné hodnoty oproti první sérii, 5,9 násobnému zrychlení oproti CPU. Nejlepšího výsledku tedy dosáhla konfigurace systému s akcelerací rozpoznávání pomocí NCS2, latence 149,275 ms pro první sérii a 182,941 ms pro druhou sérii. Ačkoliv NCS2 velice výrazně zvýšil výkon systému, výsledné hodnoty jsou příliš nízké pro potřeby *real-time* zpracování videa. Vysoké hodnoty latence jsou způsobeny komplexností modelu ArcFace, který jako backbone používá ResNet100. Při rozpoznávání vstupu pouze s jednou osobou dosáhla tato konfigurace snímkové frekvence 6,70 FPS 6.13 a tento model se ukázal jako naprosto nevyhovující pro účely *real-time* zpracování videa.

Experimenty třetí série 6.9 ukazují, že doba rozpoznání jednoho snímku videa s jednou osobou dosahuje na CPU hodnoty 66,872 ms. Akcelerací pomocí GPU došlo k 1,43 násobnému zrychlení oproti CPU na hodnotu 46,750 ms a akcelerací pomocí NCS2 k 1,96 násobnému zrychlení oproti CPU na hodnotu 34,090 ms. Výsledky experimentů čtvrté série 6.10 ukazují, že doba rozpoznání jednoho snímku videa se dvěmi osobami dosahuje na CPU hodnoty 146,636 ms – 2,19 násobné hodnoty oproti třetí sérii, na GPU hodnoty 103,439 ms – 2,21 násobné hodnoty oproti třetí sérii, 1,42 násobnému zrychlení oproti CPU a na

NCS2 48,593 ms – 1,43 násobné hodnoty oproti třetí sérii, 3,02 násobnému zrchlení oproti CPU. Nejrychlejších výsledků opět dosáhla konfigurace systému, která byla akcelerována pomocí NCS2, a to latence 65,164 ms pro třetí sérii a 79,759 ms pro čtvrtou sérii. Naměřené výsledky jsou významně lepší než výsledky první a druhé série, a to je způsobeno mnohem jednodušší architekturou modelu FaceNet, který je postaven Inception ResNetV1. Při rozpoznávání vstupu pouze s jednou osobou dosáhla tato konfigurace 15,35 FPS 6.13 a tímto vyhověla požadavku pro *real-time* zpracování videa.

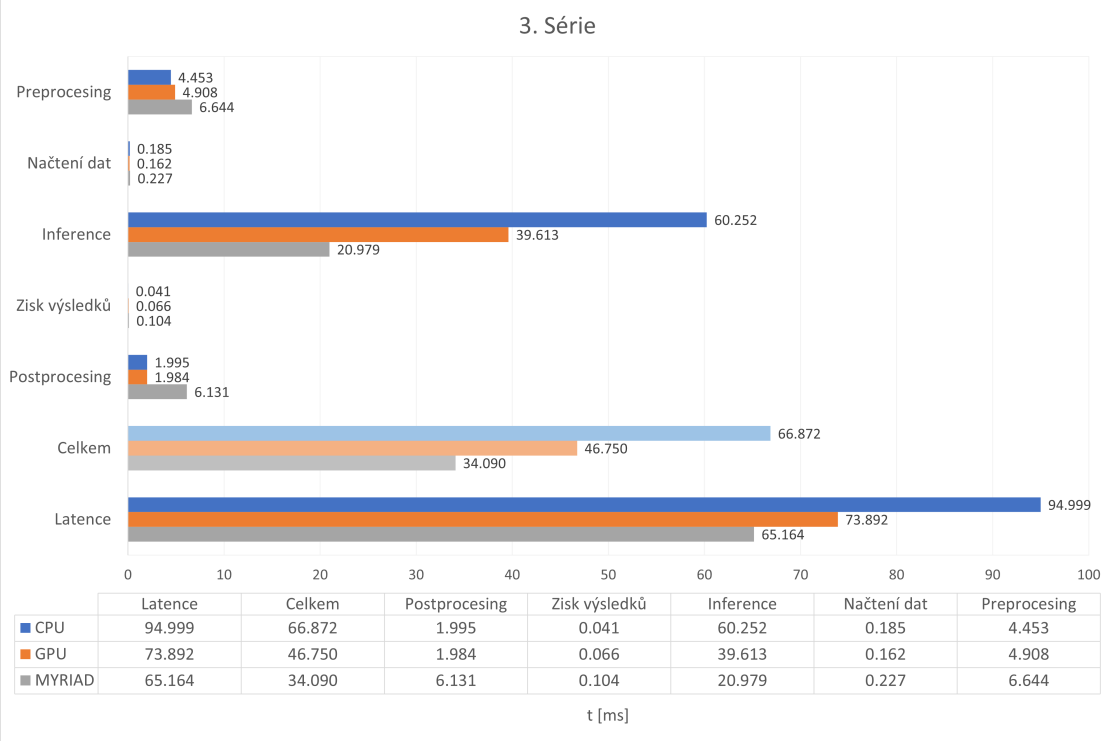
Výsledky experimentů páté série 6.11 ukazují, že doba rozpoznání jednoho snímku videa s jednou osobou dosahuje na CPU hodnoty 76,818 ms. Akcelerací pomocí GPU došlo k 2,08 násobnému zrychlení oproti CPU na hodnotu 36,980 ms a akcelerací pomocí NCS2 k 2,58 násobnému zrychlení oproti CPU na hodnotu 29,734 ms. Výsledky experimentů poslední šesté série 6.12 ukazují, že doba rozpoznání jednoho snímku videa se dvěmi osobami dosahuje na CPU hodnoty 170,007 ms – 2,21 násobné hodnoty oproti páté sérii, na GPU hodnoty 81,774 ms – 1,29 násobné hodnoty oproti páté sérii, 2,08 násobnému zrychlení oproti CPU a na NCS2 47,389 ms – 1,59 násobné hodnoty oproti páté sérii, 3,59 násobnému zrychlení oproti CPU. Nejlepších výsledků znovu dosáhla konfigurace systému, která byla akcelerována pomocí NCS2, a to latence 58,293 ms pro pátou sérii a 78,296 ms pro šestou sérii. Naměřené výsledky výrazně překonaly první a druhou sérii a jsou lepší než výsledky třetí a čtvrté série. Je to způsobeno mnohem jednodušší architekturou modelu SphereFace, který je postaven na dvacetivrstvé konvoluční neuronové síti. Při rozpoznávání vstupu pouze s jednou osobou dosáhla tato konfigurace 17,15 FPS 6.13 a tímto taktéž vyhověla požadavku pro *real-time* zpracování videa.



Obrázek 6.7: Graf 1. Série. Graf znázorňuje dosažené výsledky rozpoznávání s použitím modelu **ArcFace**. Experiment byl proveden s videem, zobrazující jednu osobu.

	Latence	Celkem	Postprocesing	Zisk výsledků	Inference	Načtení dat	Preprocesing
CPU	1062.584	931.574	5.897	0.112	914.241	0.439	10.885
GPU	432.038	405.165	4.989	0.153	389.445	0.316	10.263
MYRIAD	182.941	157.945	14.066	0.179	128.552	0.466	14.682

Obrázek 6.8: Graf 2. Série. Graf znázorňuje dosažené výsledky rozpoznávání s použitím modelu **ArcFace**. Experiment byl proveden s videem, zobrazující dvě osoby.



	Latence	Celkem	Postprocesing	Zisk výsledků	Inference	Načtení dat	Preprocesing
CPU	94.999	66.872	1.995	0.041	60.252	0.185	4.453
GPU	73.892	46.750	1.984	0.066	39.613	0.162	4.908
MYRIAD	65.164	34.090	6.131	0.104	20.979	0.227	6.644

Obrázek 6.9: Graf 3. Série. Graf znázorňuje dosažené výsledky rozpoznávání s použitím modelu **FaceNet**. Experiment byl proveden s videem, zobrazující jednu osobu.

	Latence	Celkem	Postprocesing	Zisk výsledků	Inference	Načtení dat	Preprocesing
CPU	174.748	146.636	4.992	0.097	131.009	0.403	10.134
GPU	131.786	103.439	4.892	0.146	86.985	0.358	11.058
MYRIAD	79.759	48.593	13.759	0.177	18.863	0.508	15.286

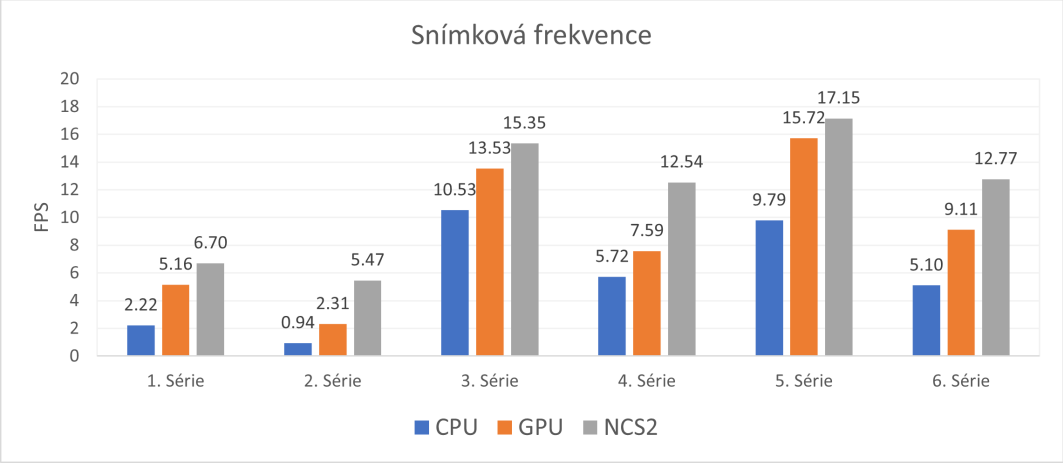
Obrázek 6.10: Graf 4. Série. Graf znázorňuje dosažené výsledky rozpoznávání s použitím modelu **FaceNet**. Experiment byl proveden s videem, zobrazující dvě osoby.

	Latence	Celkem	Postprocesing	Zisk výsledků	Inference	Načtení dat	Preprocesing
CPU	102.118	76.818	2.124	0.047	70.077	0.178	4.432
GPU	63.626	36.980	2.058	0.060	30.688	0.133	4.028
MYRIAD	58.293	29.734	4.472	0.084	19.809	0.151	5.186

Obrázek 6.11: Graf 5. Série. Graf znázorňuje dosažené výsledky rozpoznávání s použitím modelu **SphereFace**. Experiment byl proveden s videem, zobrazující jednu osobu.

	Latence	Celkem	Postprocesing	Zisk výsledků	Inference	Načtení dat	Preprocesing
CPU	196.087	170.007	5.191	0.110	154.668	0.377	9.661
GPU	109.719	81.774	5.061	0.136	66.197	0.330	10.050
MYRIAD	78.296	47.389	11.806	0.162	18.298	0.484	16.640

Obrázek 6.12: Graf 6. Série. Graf znázorňuje dosažené výsledky rozpoznávání s použitím modelu **SphereFace**. Experiment byl proveden s videem, zobrazující dvě osoby.



Obrázek 6.13: Graf srovnání dosažené snímkové frekvence pro jednotlivé série. 1. Série – model ArcFace, jedna osoba. 2. Série – model ArcFace, dvě osoby. 3. Série – model FaceNet, jedna osoba. 4. Série – model FaceNet, dvě osoby. 5. Série – model SphereFace, jedna osoba. 6. Série – model SphereFace, dvě osoby.

Experimenty dokazují že CPU poskytuje nejslabší akceleraci, GPU je výpočetně výkonnější, ale zdaleka nejlepší akceleraci nabízí NCS2. Experimenty dále ukazují význam paralelizace výpočetního výkonu, který je patrný u akcelerace pomocí GPU a NCS2. Zatímco zpracování snímku videa se dvěmi obličeji na CPU trvá přibližně dvojnásob času oproti snímku videa s jedním obličejem, zpracování snímku na GPU a NCS2 se blíží hod-

notě jedna. NCS2 se ukázal jako nejvýkonnější akcelerátor a dosáhl nelepších výsledků v každé sérii provedených experimentů. Z experimentů zaměřených na akceleraci pro různé konfigurace systému vyplynulo, že konfigurace dosahující nejvyšší snímkové frekvence 17,15 FPS a nejnižší latence 58,293 ms je konfigurace systému s využitím modelu Sphereface, která byla akcelerována pomocí *Intel Neural Compte Stick 2*. Tato konfigurace při předpokladu, že je použit video vstup z RTSP streamu, je schopna dosáhnout *real-time* zpracování. Na základě výsledků validace, kde SphereFace dosáhl přesnosti 73,77 % 6.4 je však vhodné uvažovat o použití přesnějšího modelu FaceNet 6.5 a učinit tak kompromis mezi přesností a rychlostí. FaceNet dosáhl přesnosti 97,08 %, maximální snímkové frekvence 15,35 FPS a latence 65,164 ms s akcelerací pomocí NCS2 a stále je schopen dosáhnout *real-time* zpracování. Konfigurace systému s modelem FaceNet a akcelerátorem NCS2 vyhovuje všem požadavkům, které jsou na takový systém kladeny, a proto je tou nejvhodnější konfigurací pro využití v bezpečnostním systému.

Kapitola 7

Závěr

Cílem této práce bylo nastudovat problematiku rozpoznávání obličejů a dostupné algoritmy založené na neuronových sítích, dále se seznámit s dostupnými metodami akcelerace a hardwarovým akcelerátorem *Intel Neural Compute Stick 2*, společně se sadou nástrojů OpenVINO a následně tyto dosažené znalosti sumarizovat a na základě těchto znalostí navrhnout a implementovat aplikaci, která bude provádět rozpoznávání osob v obraze pomocí dostupných algoritmů.

Tato práce shrnuje tradiční metody, dříve používané pro rozpoznávání obličejů a rozvádí metodu hlubokých neuronových sítí, konkrétně konvolučních neuronových sítí. Zabývá se konkrétními algoritmy pro detekci a rozpoznávání obličejů. Z algoritmů používaných pro rozpoznávání sumarizuje nejpopulárnější metody FaceNet, poprvé publikovanou v roce 2015, SphereFace z roku 2017 a ArcFace z roku 2018, včetně dosažené přesnosti na testovacích datasetech *LFW* a *YTF*. Sumarizuje způsoby jejich akcelerace na jednotlivých úrovních a technologie používané pro vestavěné systémy.

Na základě získaných znalostí byl vytvořen systém na konceptu *edge computingu* a navržena aplikace s uživatelským rozhraním, využívající zmiňované algoritmy používané pro rozpoznávání obličejů z obrazu. Tento systém byl navržen tak, aby mohl být použit jako domácí bezpečnostní systém. Lze jej napojit na RTSP stream z IP kamery a může být využit pro střežení vybrané oblasti. Zároveň jej lze připojit k mobilní aplikaci Pushover a zasílat upozornění o příchodu neznámé osoby na registrované mobilní zařízení. Aplikace je navržena tak, aby mohla být spuštěna s různým aloritmem pro detekci i rozpoznávání obličeje na CPU, nebo akcelerována pomocí GPU nebo NCS2.

S implementovaným systémem pak byla provedena řada experimentů. Nejprve byly experimenty zaměřeny na výkonnost a přesnost jednotlivých modelů, použitých pro rozpoznávání obličejů. Nejlepších výsledků dosáhl model FaceNet, který dosáhl přesnosti 97,08 %. Druhá část experimentů se zaměřovala na zhodnocení rychlosti rozpoznávání na jednotlivých platformách. Experimenty přinesly odpovědi na otázky jakých přesností dosahují jednotlivé modely, jakých výsledků dosahuje akcelerace systému pomocí CPU, GPU a NCS2, jaká konfigurace je nejvhodnější pro jednotlivé modely a která konfigurace dosahuje nejvyšší snímkové frekvence a nejnižší latence mezi porovnávanými modely. Nejlepší snímkové frekvence a latence dosáhl model SphereFace, akcelerovaný pomocí NCS2 a to 17,15 FPS a latence 58,293 ms. Model Facenet dosáhl snímkové frekvence 15,35 FPS a latence 65,164 ms. Pro dosažení rovnováhy mezi přesností a rychlostí je nejlepší konfigurace systému s použitím modelu FaceNet, akcelerovaným pomocí NCS2.

Tato práce přináší univerzální aplikaci, která může být využita různou konfigurací systému, může být napojena na vlastní RTSP stream a lze ji využít jako domácí jednokamerový

bezpečnostní systém. Zároveň práce přináší srovnání výkonnosti NCS2 s platformami CPU a GPU a přesnosti jednotlivých nejpopulárnějších algoritmů, používaných pro rozpoznávání obličejů z obrazu.

Literatura

[1] Lopatina, O. L., Komleva, Y. K., Gorina, Y. V., Higashida, H. a Salmina, A. B. Neurobiological Aspects of Face Recognition: The Role of Oxytocin. *Frontiers in Behavioral Neuroscience*. 2018, sv. 12. ISSN 1662-5153. [cit. 2022-12-29]. Dostupné z: https://www.frontiersin.org/articles/10.3389/fnbeh.2018.00195.

[2] Jenkins, R., Dowsett, A. J. a Burton, A. M. How many faces do people know? *Proceedings of the Royal Society B: Biological Sciences*. 2018, sv. 285, č. 1888, s. 20181319. DOI: 10.1098/rspb.2018.1319. [cit. 2023-04-13]. Dostupné z: https://royalsocietypublishing.org/doi/abs/10.1098/rspb.2018.1319.

[3] Trigueros, D. S., Meng, L. a Hartnett, M. Face recognition: From traditional to deep learning methods. *ArXiv preprint arXiv:1811.00116*. 2018.

[4] Roomi, M. a Beham, D. A Review Of Face Recognition Methods. *International Journal of Pattern Recognition and Artificial Intelligence*. Duben 2013, sv. 27. DOI: 10.1142/S0218001413560053.

[5] Lowe, D. Object recognition from local scale-invariant features. In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. 1999, sv. 2, s. 1150–1157 vol.2. DOI: 10.1109/ICCV.1999.790410.

[6] Masi, I., Wu, Y., Hassner, T. a Natarajan, P. Deep Face Recognition: A Survey. In: *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. 2018, s. 471–478. DOI: 10.1109/SIBGRAPI.2018.00067.

[7] Han, J., Kamber, M. a Pei, J. 2 – Getting to Know Your Data. In: Han, J., Kamber, M. a Pei, J., ed. *Data Mining (Third Edition)*. Third Edition. Boston: Morgan Kaufmann, 2012, s. 39–82. The Morgan Kaufmann Series in Data Management Systems. DOI: https://doi.org/10.1016/B978-0-12-381479-1.00002-2. ISBN 978-0-12-381479-1. [cit. 2023-04-11]. Dostupné z: https://www.sciencedirect.com/science/article/pii/B9780123814791000022.

[8] Standards, N. I. of a Technology. *COSDIST: Cosine Distance* [https://www.itl.nist.gov/div898/software/dataplot/refman2/auxillar/cosdist.htm]. 2017. [cit. 2023-04-29].

[9] Davis, J. a Goadrich, M. The Relationship between Precision-Recall and ROC Curves. In: *Proceedings of the 23rd International Conference on Machine Learning*. New York, NY, USA: Association for Computing Machinery, 2006, s. 233–240. ICML '06. DOI: 10.1145/1143844.1143874. ISBN 1595933832. [cit. 2023-04-12]. Dostupné z: https://doi.org/10.1145/1143844.1143874.

[10] Géron, A. *Neural networks and deep learning*. O'Reilly, 2018.

[11] Trojan, S. *Lékařská fyziologie*. Grada Publishing as, 2003.

[12] Du, K.-L. a Swamy, M. N. *Neural networks and statistical learning*. Springer Science & Business Media, 2013.

[13] Meng, Z., Hu, Y. a Ancey, C. Using a data driven approach to predict waves generated by gravity driven mass flows. *Water*. MDPI. 2020, sv. 12, č. 2, s. 600.

[14] John McGonagle, C. W. *Backpropagation*. 2023. [cit. 2023-01-01]. Dostupné z: https://brilliant.org/wiki/backpropagation/.

[15] LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R. et al. Handwritten Digit Recognition with a Back-Propagation Network. In: Touretzky, D., ed. *Advances in Neural Information Processing Systems*. Morgan-Kaufmann, 1989, sv. 2. [cit. 2022-12-29]. Dostupné z: https://proceedings.neurips.cc/paper_files/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf.

[16] Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A. et al. Recent advances in convolutional neural networks. *Pattern Recognition*. 2018, sv. 77, s. 354–377. DOI: https://doi.org/10.1016/j.patcog.2017.10.013. ISSN 0031-3203. [cit. 2023-02-11]. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0031320317304120.

[17] Albawi, S., Mohammed, T. A. a Al Zawi, S. Understanding of a convolutional neural network. In: Ieee. *2017 international conference on engineering and technology (ICET)*. 2017, s. 1–6.

[18] Pei, Z., Xu, H., Zhang, Y., Guo, M. a Yang, Y.-H. Face recognition via deep learning using data augmentation based on orthogonal experiments. *Electronics*. MDPI. 2019, sv. 8, č. 10, s. 1088.

[19] Zbořil, F. V. *Základy umělé inteligence, Studijní text*. 2022.

[20] Nelson, H. *Essential math for AI*. O'Reilly Media, Inc., 2022. [cit. 2022-10-10]. Dostupné z: https://learning.oreilly.com/library/view/essential-math-for/9781098107628/.

[21] *MaxpoolSample2*. Https://computersciencewiki.org, Feb 2018. [cit. 2022-10-18]. Dostupné z: https://computersciencewiki.org/index.php/File:MaxpoolSample2.png.

[22] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. a Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*. JMLR. org. 2014, sv. 15, č. 1, s. 1929–1958.

[23] Guo, Y., Zhang, L., Hu, Y., He, X. a Gao, J. MS-Celeb-1M: A Dataset and Benchmark for Large-Scale Face Recognition. *CoRR*. 2016, abs/1607.08221. [cit. 2022-11-29]. Dostupné z: http://arxiv.org/abs/1607.08221.

[24] Cao, Q., Shen, L., Xie, W., Parkhi, O. M. a Zisserman, A. VGGFace2: A Dataset for Recognising Faces across Pose and Age. In: *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*. 2018, s. 67–74. DOI: 10.1109/FG.2018.00020. [cit. 2023-01-11].

[25] Liu, W., Wen, Y., Yu, Z., Li, M., Raj, B. et al. Sphereface: Deep hypersphere embedding for face recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, s. 212–220.

[26] Huang, G. B., Jain, V. a Learned Miller, E. Unsupervised Joint Alignment of Complex Images. In: *ICCV*. 2007.

[27] Wolf, L., Hassner, T. a Maoz, I. Face recognition in unconstrained videos with matched background similarity. In: IEEE. *CVPR 2011*. 2011, s. 529–534.

[28] Yang, S., Luo, P., Loy, C.-C. a Tang, X. WIDER FACE: A Face Detection Benchmark. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[29] Chen, W., Huang, H., Peng, S., Zhou, C. a Zhang, C. YOLO-face: a real-time face detector. *The Visual Computer*. Springer. 2021, sv. 37, č. 4, s. 805–813.

[30] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S. et al. SSD: Single Shot MultiBox Detector. In: Leibe, B., Matas, J., Sebe, N. a Welling, M., ed. *Computer Vision – ECCV 2016*. Cham: Springer International Publishing, 2016, s. 21–37. ISBN 978-3-319-46448-0.

[31] Schroff, F., Kalenichenko, D. a Philbin, J. Facenet: A unified embedding for face recognition and clustering. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, s. 815–823.

[32] Deng, J., Guo, J., Xue, N. a Zafeiriou, S. Arcface: Additive angular margin loss for deep face recognition. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, s. 4690–4699.

[33] Securiapro.cz. *Základní rozdělení kamerových systémů*. 2019. [cit. 2023-05-01]. Dostupné z: https://www.securiapro.cz/clanek/zakladni-rozdeleni-kamerovych-systemu/.

[34] KamerovySystem.cz. *Akční kamerový systém – porovnání*. [cit. 2023-05-01]. Dostupné z: https://www.kamerovysystem.cz/kamerove-systemy/akcni-kamerovy-system/porovnani.php.

[35] Kaur, A. a Singh, S. A Survey of Streaming Protocols for Video Transmission. In:. New York, NY, USA: Association for Computing Machinery, 2022, s. 186–191. DSMLAI '21'. DOI: 10.1145/3484824.3484892. ISBN 9781450387637. [cit. 2023-03-11]. Dostupné z: https://doi.org/10.1145/3484824.3484892.

[36] *Real Time Streaming Protocol (RTSP)* [Internet Requests for Comments]. RFC 2326. RFC Editor, April 1998. [cit. 2023-03-05]. Dostupné z: https://www.rfc-editor.org/rfc/rfc2326.html#page-5.

[37] Mgr. Jana Skokanová. *Videokonference a streaming multimédií*. 2022. [cit. 2023-03-20]. Dostupné z: https://moodle.vut.cz/pluginfile.php/516428/mod_resource/content/1/videokonference22.pdf.

[38] Hua, H., Li, Y., Wang, T., Dong, N., Li, W. et al. Edge Computing with Artificial Intelligence: A Machine Learning Perspective. *ACM Comput. Surv.* New York, NY, USA: Association for Computing Machinery. jan 2023, sv. 55, č. 9. DOI: 10.1145/3555802. ISSN 0360-0300. [cit. 2023-04-05]. Dostupné z: https://doi.org/10.1145/3555802.

[39] Zhang, Q., Zhang, M., Chen, T., Sun, Z., Ma, Y. et al. Recent advances in convolutional neural network acceleration. *Neurocomputing*. Elsevier. 2019, sv. 323, s. 37–51.

[40] Wang, Z., Li, H., Yue, X. a Meng, L. Briefly Analysis about CNN Accelerator based on FPGA. *Procedia Computer Science*. 2022, sv. 202, s. 277–282. DOI: https://doi.org/10.1016/j.procs.2022.04.036. ISSN 1877-0509. International Conference on Identification, Information and Knowledge in the internet of Things, 2021. Dostupné z: https://www.sciencedirect.com/science/article/pii/S1877050922005701.

[41] Coral. *Products, Helping you bring local AI to applications from prototype to production*. 2022. [cit. 2022-11-28]. Dostupné z: https://coral.ai/products/.

[42] Intel. *Intel® Neural Compute Stick 2 (Intel® NCS2)*. 2022. [cit. 2022-11-19]. Dostupné z: https://www.intel.com/content/www/us/en/developer/articles/tool/neural-compute-stick.html.

[43] Meel, V. *Intel Neural Compute Stick 2 – AI Vision Accelerator Review 2021*. 2021. [cit. 2022-11-19]. Dostupné z: https://viso.ai/edge-ai/intel-neural-compute-stick-2/.

[44] Intel. *OpenVINO dokumentation*. 2022. [cit. 2022-11-28]. Dostupné z: https://docs.openvino.ai/latest/home.html.

[45] Intel. *Intel® Vision Accelerator Design With Intel® Movidius™ Vision Processing Unit (VPU)*. 2022. [cit. 2022-11-20]. Dostupné z: https://www.intel.com/content/www/us/en/developer/topic-technology/edge-5g/hardware/vision-accelerator-movidius-vpu.html.

[46] *NX6412*. 1655361879487388. Maxtang Technology Co. Ltd., červen 2022.

[47] OpenVINO. *Face-detection-retail-0004*. April 2023. [cit. 2023-04-18]. Dostupné z: https://docs.openvino.ai/2022.1/omz_models_model_face_detection_retail_0004.html.

[48] OpenVINO. *Facial-landmarks-35-adas-0002*. April 2023. [cit. 2023-04-18]. Dostupné z: https://docs.openvino.ai/2022.1/omz_models_model_facial_landmarks_35_adas_0002.html.

[49] ONNX. *Arcface*. April 2023. [cit. 2023-04-18]. Dostupné z: https://github.com/onnx/models/tree/main/vision/body_analysis/arcface.

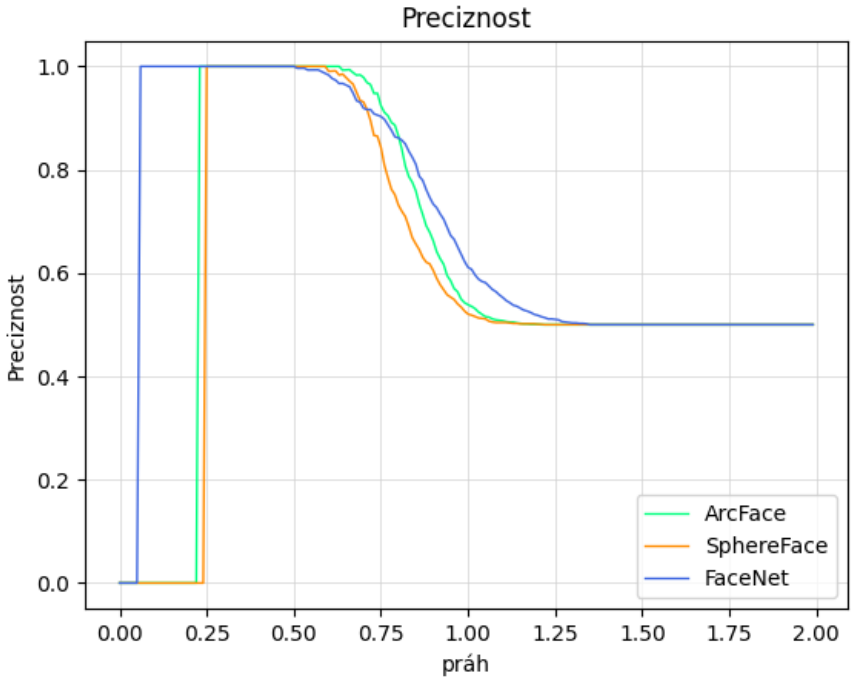
[50] WY1IU. *SphereFace*. April 2023. [cit. 2023-04-18]. Dostupné z: https://github.com/wy1iu/sphereface.

[51] SANBERG, D. *Facenet*. April 2023. [cit. 2023-04-18]. Dostupné z: https://github.com/davidsandberg/facenet.

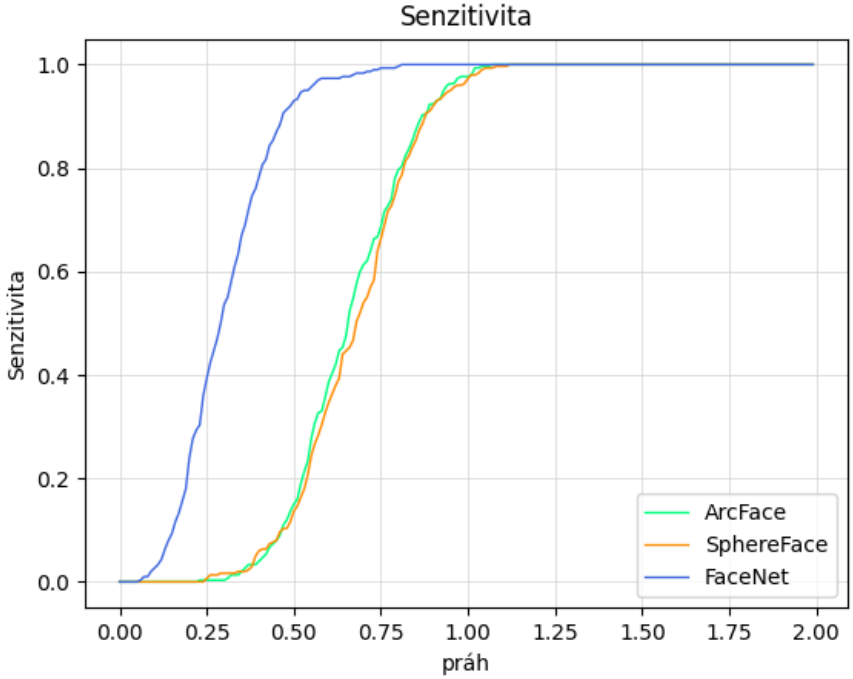
[52] SERENGIL. *Serengil/deepface: A lightweight face recognition and facial attribute analysis (age, gender, emotion and race) library for python*. Serengil, Sefik Ilkin and Ozpinar, Alper. [cit. 2023-04-20]. Dostupné z: https://github.com/serengil/deepface.

Příloha A

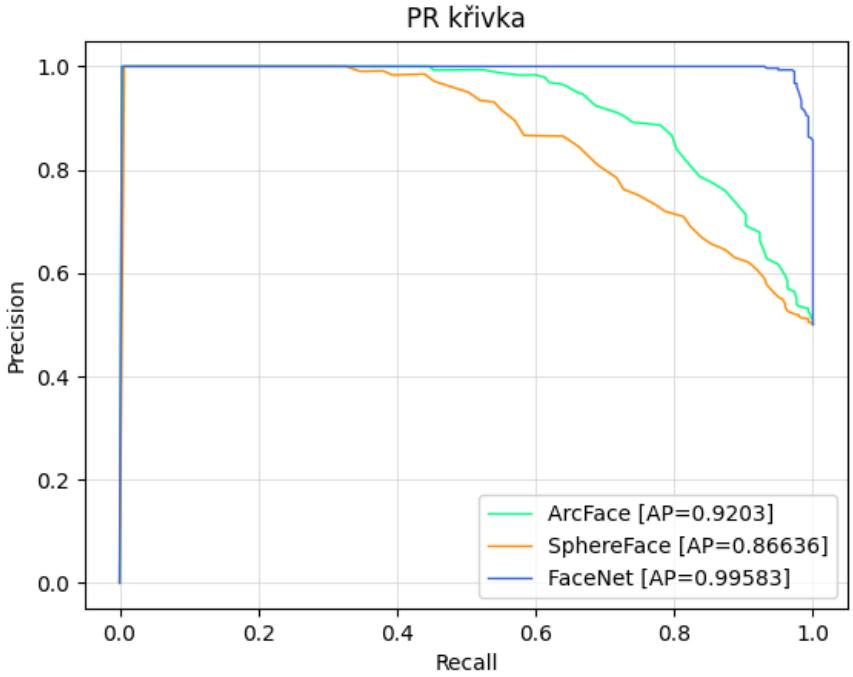
Výsledky validace modelů



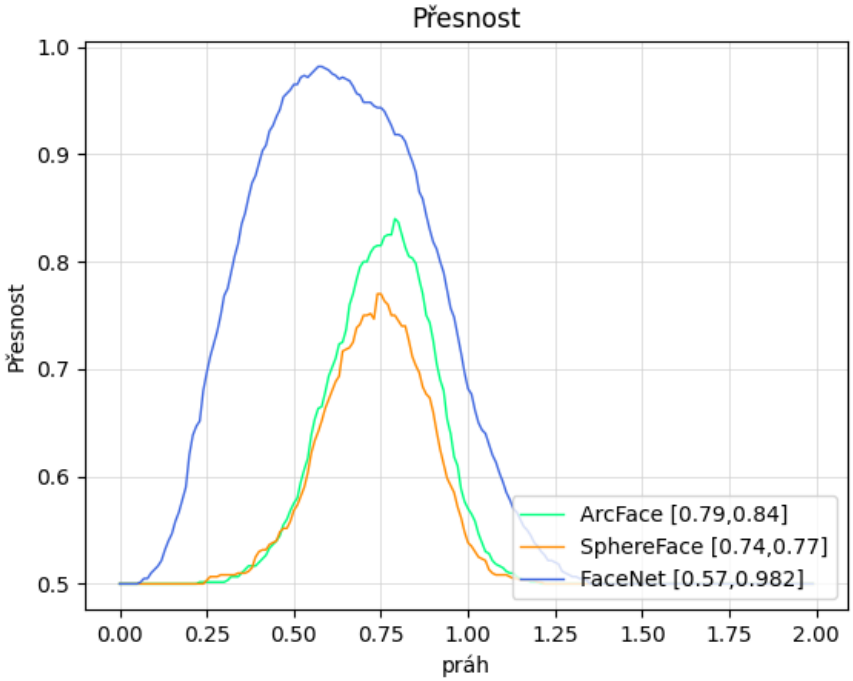
Obrázek A.1: Graf zobrazující preciznost (*precision*) všech validovaných modelů v závislosti na měnícím se prahu.



Obrázek A.2: Graf zobrazující senzitivitu (*recall*) všech validovaných modelů v závislosti na měnícím se prahu.



Obrázek A.3: Graf zobrazující PR (*Precision-Recall*) křivku všech validovaných modelů v závislosti na měnícím se prahu.



Obrázek A.4: Graf zobrazující přesnost (*accuracy*) všech validovaných modelů v závislosti na měnícím se prahu. ArcFace dosáhl nejvyšší přesnosti 0,84 při hodnotě prahu 0,79. SphereFace dosáhl nejvyšší přesnosti 0,77 pro práh 0,74. FaceNet dosáhl nejlepší přesnosti ze všech porovnávaných modelů, a to 0,982 pro hodnotu prahu 0,57.

Příloha B

Obsah přiloženého paměťového média

`README.md` – Obsahuje návod k použití aplikace a popis přiložených souborů

`/latex` – Obsahuje text práce a zdrojové soubory v LaTeXu

`/src` – Adresář se zdrojovými kódy a příslušenstvím

- `/doc` – Obsahuje technickou dokumentaci zdrojových souborů
- `/face_db` – Databáze obličejů s demonstračními identitami
- `/models` – Adresář s modely na detekci a rozpoznávání obličejů
- `/python` – Zdrojový kód aplikace
- `/videos` – Adresář s testovacími videi
- `demo.mp4` – Demonstrační video hotové aplikace
- `requirements.txt` – Závislosti aplikace