Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra kybernetiky

DIPLOMOVÁ PRÁCE

Identifikace obličejů pomocí metod počítačového vidění

Plzeň, 2017

Vypracoval:
David Smolík

Vedoucí práce:
Ing. Marek Hrúz, Ph.D.

Prohlášení

Předkládám tímto k posouzení a obhajobě diplomovou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne:

.....

podpis

Poděkování

Rád bych poděkoval vedoucímu diplomové práce Ing. Marku Hrúzovi, Ph.D. za věcné konzultace a odborné rady k danému tématu.

Děkuji také organizaci MetaCentrum VO za poskytnutí přístupu k jejich výpočetní technice v rámci programu "Projects of Large Research, Development, and Innovations Infrastructures"(CESNET LM2015042).

Abstrakt

Tato práce obecně popisuje problematiku rozpoznávání, identifikace a verifikace tváře. Prozkoumává stávající metody rozpoznávání s podrobnějším popisem některých tradičních a současných vrcholných přístupů. Porovnává existující databáze obličejů vhodné k trénování a testování rozpoznávacího systému. Popisuje roli neuronových sítí v úloze rozpoznávání. Nedílnou součástí je také návrh a implementace vlastního identifikačního systému metodou siamských neuronových sítí, který porovnává identitu dvou tváří a rozhoduje, zda se jedná o shodnou osobu, či nikoliv. Před fází návrhu systému je diskutován výběr databáze obličejů, samotný návrh pak zahrnuje tvorbu datasetu, sestavení architektury sítě, výběr metody učení a volbu parametrů. V závěru je vlastní návrh vyhodnocen a porovnán se stávajícími metodami.

Klíčová slova: rozpoznávání tváře, identifikace tváře, verifikace tváře, databáze obličejů, neuronová síť, konvoluční neuronová síť, siamská neuronová síť, SGD, backpropagation

Abstract

This thesis describes in general terms the issue of face recognition, identification or verification. It examines existing recognition methods with a more detailed description of some traditional and current top approaches, compares existing face databases suitable for training and testing the recognition system and describes a role of neural networks in the recognition task. An integral part is also the design and implementation of the identification system using the siamese neural network, which compares the identity of two faces and decides whether it is the same person or not. Before the design phase is discussed a selection of the face database, the design then involves creating a dataset, building a network architecture, choosing a learning method and selecting parameters. In conclusion, the proposed system is evaluated and compared with the existing methods.

Keywords: face recognition, face identification, face verification, face database, neural network, convolutional neural network, siamese neural network, SGD, backpropagation

Obsah

1 Úvod

S plynoucím časem člověku v jeho životě začínají čím dál více pomáhat různé nástroje a systémy, které jistým způsobem napodobují lidské chování a projevují tím určitou formu inteligence. Od těchto vlastností byl odvozen obor zabývající se těmito systémy, nazvaný umělá inteligence s podoblastí strojového učení. Příkladem mohou být hlasoví asistenti mobilních zařízení, s nimi související systémy pro rozpoznávání a porozumění mluvené řeči, systémy počítačového vidění pro detekci, rozpoznávání a sledování objektů v digitálním obrazu, umělá inteligence v počítačových hrách, systémy autonomního řízení nebo expertní systémy sbírající data a provádějící predikci v oblastech medicíny, marketingu, finanční sféry, apod. Jedním konkrétním příkladem jsou i sytémy počítačového vidění pro rozpoznávání obličeje.

Rozpoznávání (identifikace, verifikace) obličeje je základní lidská schopnost už od narození. Je to jedna z prvních kognitivních funkcí, kterou si člověk osvojuje již ve fázi novorozence. Otázkou bylo, zdali by to dokázala i počítačová technika. Začalo se na tom pracovat už kolem 60. let a během několika posledních dekád se tuto schopnost podařilo úspěšně přenést i do oblasti počítačového vidění a rozpoznávat obličeje s uspokojivou přesností dokáží v dnešní době už i stroje. K problému se přistupuje podobně jako z pohledu člověka. V obličeji se hledají určité charakteristické rysy, které se následně porovnávají s pamětí, konkrétně tedy s charakteristickými rysy známých, naučených tváří. Obličej je přitom reprezentován jako digitální obraz, poskládaný z pixelů. V dnešní době jde o velice populární podoblast počítačového vidění, která se doposud stále vyvíjí a rozšiřuje, a která mimo jiné pomáhá lepšímu porozumnění a analýze digitálního obrazu.

Účelem této práce je mimo jiné proniknout do problematiky rozpoznávání obličeje, prozkoumat a popsat stávající přístupy a navrhnout vlastní rozpoznávací systém předem určenou metodou, což je v tomto případě siamská neuronová síť

(siamská síť). Kromě konkrétní teorie siamských sítí se práce věnuje obecnému popisu neuronových sítí, na kterých jsou siamské sítě postavené. Návrh vlastního systému zahrnuje několik dílčích procesů, které jsou v práci podrobněji popsány a patří mezi ně například výběr databáze obličejů, tvorba datasetu, sestavení architektury sítě nebo výběr metody učení s volbou parametrů. Na závěr je model vlastního návrhu testován a porovnán se stávajícími metodami.

1.1 Motivace

Systémy rozpoznávání obličeje nacházejí čím dál více aplikací v běžném životě. Ať už to jsou systémy pro kontrolu totožnosti podezřelých nebo pohřešovaných osob, pro kontrolu totožnosti osob vcházejících do budovy (školy, úřady, apod.) nebo například systémy pro odemknutí elektronických zařízení (smartphone, notebook, apod.) či systémy pro označování osob na fotografiích v sociálních sítích. O systému rozpoznávání tváře začínají přemýšlet například i obchody společně s poskytovateli platebních služeb v tom smyslu, že člověk zaplatí nákup pouze identifikací jeho obličeje, aniž by k tomu potřeboval platební kartu nebo fyzické peníze. Podobných využití stále přibývá, a to zejména díky zlepšení kvality rozpoznávání v posledních pár letech při aplikaci metod hlubokého učení. Přibližně od roku 2012 se pro rozpoznávání začaly používat konvoluční neuronové sítě, spadající do metod hlubokého učení a od té doby jsou řazeny mezi nejlepší přístupy.

I přes velký úspěch konvolučních neuronových sítí a hlubokého učení nejsou ani současné přístupy stoprocentně přesné, zejména kvůli variabilitě obličeje, světelným podmínkám a podobným komplikacím. Výzvou a zároveň motivací nynějšího výzkumu je tedy zpřesnit rozpoznávání obličeje v různých pózách s různými výrazy, v přirozeném prostředí, v různých scénách, s módními doplňky (brýle, šátek, apod.) nebo například zajistit invarianci vůči změnám v obličeji způsobených věkem.

2 Rozpoznávání a identifikace tváře

Za rozpoznáváním tváří (obličejů) v počítačovém vidění ve většině případů stojí nějaký systém schopný identifikovat, verifikovat osobu na digitálním obraze nebo sekvenci obrazů (videu). Jedním ze způsobů jak to provést, tedy jak poznat identitu obličeje, je porovnání jeho charakteristických rysů s jinými obličeji z vybrané databáze. Tento přístup je pravděpodobně nejvíce intuitivní. Jiným přístupem může být normalizace, komprese obrázku a výběr pouze těch dat (pixelů), které jsou užitečné pro rozpoznávání.

Na systémech rozpoznávání pracoval už v 60. letech W. W. Bledsoe, za první oficiální se ovšem považuje systém, který v roce 1973 představil T. Kanade [1]. Jako charakteristické rysy volí pozici očí, uší nebo nosu (vyjádřeno číselně) a skládá je do tzv. příznakového vektoru. Rozpoznávání pak uskutečňuje výpočtem Euklidovské vzdálenosti mezi příznakovými vektory testovaného a referenčního obrazu. Zároveň poukazuje na to, že nejtěžším úkolem je nalézt v obraze charakteristické rysy, tedy vhodně sestavit příznakový vektor. Tento způsob bylo možné použít na rozsahově menší úlohy, ale pro velké datasety byla informace uložená v příznacích nedostačující. Začaly se tedy hledat nové efektivnější přístupy a algoritmy, které by rozpoznávání vylepšily.

Během dalších let se objevily různé přístupy, které lze klasifikovat jako geometrické nebo statistické. Geometrické přístupy analyzují lokální příznaky v obličeji a geometrické vztahy mezi nimi. Příkladem může být metoda zvnaná Elastic Bunch Graph Matching [2]. Statistické přístupy berou jako příznaky celý obrázek, na začátku je tedy sestaven příznakový vektor o vysoké dimenzi. Vhodnou statistickou metodou se pak snaží dimenzi vekotru snížit, jinými slovy dle dané metody dojde k extrakci pouze informativních příznaků. Mezi statistické metody lze řadit například Eigenfaces - Principal Component Analysis (PCA)

[3], Fisherfaces - Linear Discriminant Analysis (LDA) [4], Independent Component Analysis (ICA) [5] nebo kernelové metody jako Kernel PCA, Kernel LDA [6]. Jinými přístupy mohou být například metody zvané Gabor Wavelets [2] nebo Local Binary Patterns (LBP) [7], které z obrazu získávají pouze příznaky lokálních oblastí o nízké dimenzi. V mnoha případech jsou tyto metody různě modifikovány nebo kombinovány pro dosažení lepší kvality rozpoznávání. Nevýhodou všech těchto přístupů je, že jsou velice citlivé na transformace a variabilitu ve vstupním obrazu (posunutí, rotace, změna výrazu obličeje, apod.). Bez využití apriorní informace o těchto jevech potom jejich kvalita radikálně klesá. Více informací v [8].

V současnosti se na vrchol popularity dostaly algoritmy hlubokého učení (deep learning), založené na konvolučních neuronových sítích, které svou kvalitou překonávají všechny předešlé. Oproti tradičním metodám mají výhodu v tom, že jsou invariantní vůči transformacím, variabilitě obrazu a nepotřebují k tomu apriorní informaci, neboť se na tyto jevy adaptují za běhu.

2.1 Tradiční metody

Následující tradiční metody patří mezi ty nejznámější a jsou volně dostupné například v knihovně počítačového vidění OpenCV [9].

2.1.1 Eigenfaces

Tato metoda spadá mezi statistické přístupy a jak už bylo popsáno výše, snaží se vhodně snížit dimenzi příznakového vektoru. Problém je v tom, že na začátku je obrázek reprezentován vektorem příznaků o vysoké dimenzi, poskládaný ze všech pixelů vstupního obrazu. Pro vstupní obraz o velikosti $p \times q$ dostaneme vektor o dimenzi $m = pq$. Každá dimenze ale obsahuje jiné množství informace, některá méně, některá více. Hlavní myšlenkou potom je, že dimenze, které obsahují nejméně informace jsou nejméně užitečné a lze je jednoduše vypustit. K tomu slouží technika zvaná Principal Component Analysis (PCA), která nalézá v datech směry s největší variancí, tedy s největší informativností a nazývá je hlavní komponenty.

Nechť $X = \{x_1, x_2, \ldots, x_n\}$ je množina trénovacích dat, pak algoritmus této metody vypadá následovně.

1. Spočte se střední hodnota μ

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{2.1}$$

2. Spočte se kovarianční matice S

$$S = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T \tag{2.2}$$

3. Vypočtou se vlastní čísla λ_i a vlastní vektory v_i matice S

$$Sv_i = \lambda_i v_i, \quad i = 1, 2, \ldots, n \tag{2.3}$$

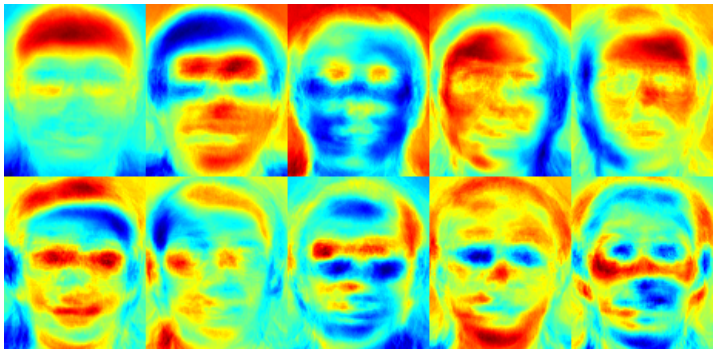
4. Vlastní čísla se seřadí sestupně podle velikosti a k nim se přiřadí odpovídající vlastní vektory. Hlavní komponenty jsou potom ty vlastní vektory, které odpovídají největším vlastním číslům a jejich počet je k .

   Souřadnice bodu v prostoru o nižší dimenzi (nového příznakového vektoru) je pak dána vztahem

$$y_i = W^T x_i\,, \tag{2.4}$$

   kde $W = [v_1, v_2, \ldots, v_k]$.

Tímto způsobem se transformují všechny příklady trénovací množiny a zároveň neznámý, testovaný příklad do společného podprostoru. Klasifikace je pak uskutečněna nalezením nejbližšího souseda neznámého příkladu v daném podprostoru.



Obrázek 2.1: Obličeje po aplikaci metody Eigenfaces, zobrazené v barevné škále, převzato z [8].

2.1.2 Fisherfaces

Fisherfaces spadá také mezi statistické metody a k redukci dimenze využívá techniku zvanou Linear Discriminant Analysis (LDA). S touto technikou přišel Sir R. A. Fisher a poprvé ji aplikoval na klasifikaci rostlin. Hlavní myšlenkou je lépe od sebe oddělit jednotlivé třídy. A právě k tomu je určena technika LDA, která má za úkol maximalizovat poměr mezitřídní a vnitrotřídní variance. Jinými slovy příklady ze stejné třídy by se měly shlukovat blízko u sebe, zatímco jednotlivé třídy by měly být od sebe co nejdále.

Nechť $X = \{X_1, X_2, \ldots, X_c\}$ je množina příkladů získaných z c tříd.

$$X_i = \{x_1, x_2, \ldots, x_n\} \tag{2.5}$$

1. Spočte se mezitřídní kovarianční matice S_B a vnitrotřídní kovarianční matice S_W

$$S_B = \sum_{i=1}^{c} N_i(\mu_i - \mu)(\mu_i - \mu)^T \tag{2.6}$$

$$S_W = \sum_{i=1}^{c} \sum_{x_j \in X_i} (x_j - \mu_i)(x_i - \mu_i)^T, \tag{2.7}$$

kdeN_i je počet příkladů dané třídy, μ je střední hodnota všech dat,

$$\mu = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{2.8}$$

a μ_i je střední hodnota i-té třídy, $i \in \{1, \ldots, c\}$,

$$\mu_i = \frac{1}{|X_i|} \sum_{x_j \in X_i} x_j \tag{2.9}$$

2. Dále se postupuje jako v případě metody Eigenfaces, vypočtou se vlastní čísla λ_i a vlastní vektory w_i matice S_B a S_W,

$$S_B v_i = \lambda_i S_W w_i, \quad i = 1, 2, \ldots, c-1 \tag{2.10}$$

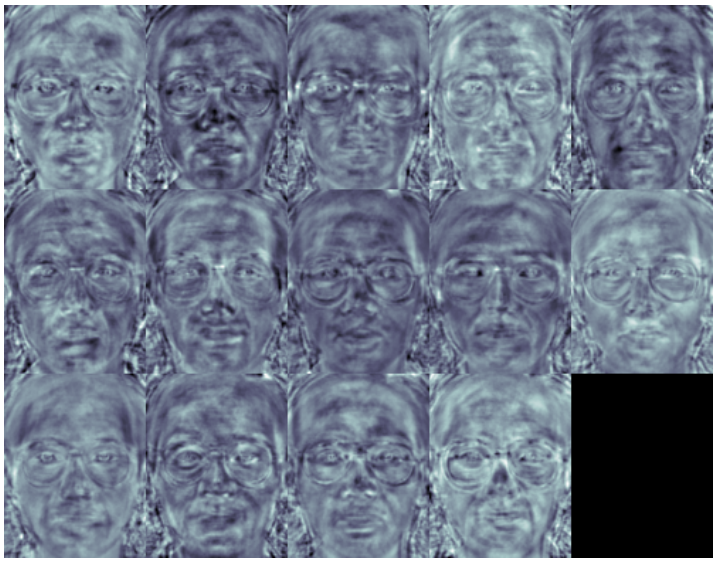
3. Z vlastních vektorů se sestaví projekce W_{opt}, která maximalizuje separabilitu tříd,

$$W_{opt} = argmax_W \frac{|W^T S_B W|}{|W^T S_W W|} \tag{2.11}$$

$$= [w_1, w_2, \ldots, w_k], \tag{2.12}$$

kde vlastní vektory w_i odpovídají největším vlastním číslům, jejichž počet je k. Souřadnice bodu v prostoru o nižší dimenzi (nového příznakového vektoru) je pak opět dána vztahem

$$y_i = W_{opt}^T x_i \,. \tag{2.13}$$

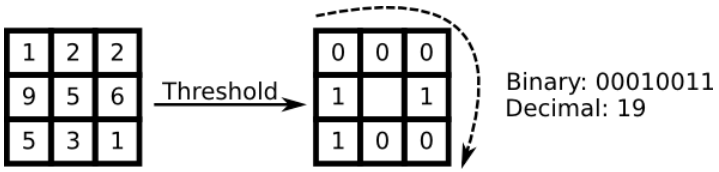


Obrázek 2.2: Obličeje po aplikaci metody Fisherfaces, převzato z [8].

2.1.3 Local Binary Patterns

Local Binary Patterns (LBP) spadá mezi metody založené na lokálních příznacích. Hlavním rozdílem oproti statistickým metodám je skutečnost, že příznakovým vektorem není celý obrázek, ale pouze lokální příznaky objektu. Výhodou těchto lokálních příznaků je jejich nízká dimenze a snadná manipulovatelnost. Základní myšlenkou LBP je zachytit lokální strukturu porovnáním každého pixelu s jeho okolím. Každý pixel se vezme jako střed, podle něhož jsou prahovány hodnoty

okolí. Pokud je intenzita sousedního pixelu větší než středového, označíme ho číslem 1, v opačném případě číslem 0. Pokud uvažujeme osmiokolí, každý pixel je pak definován osmi hodnotami, které lze chápat jako binární číslo. Toto číslo potom bývá nazýváno jako lokální binární vzor (LBP).



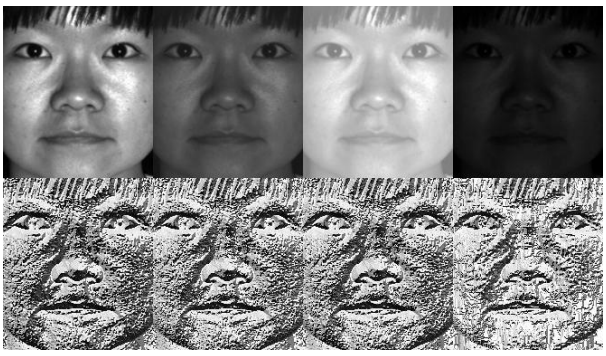
Obrázek 2.3: Princip prahování v metodě LBP, převzato z [8].

Formálně lze popsat LBP následujícím vztahem,

$$LBP(x_c, y_c) = \sum_{p=0}^{P-1} 2^p s(i_p - i_c)\,, \tag{2.14}$$

kde (x_c, y_c) je středový pixel s intenzitou i_c, i_n je intenzita sousedního pixelu a s je funkce definovaná následujícím vztahem,

$$s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0\,. \end{cases} \tag{2.15}$$



Obrázek 2.4: Obličeje před (horní řádek) a po (dolní řádek) aplikaci metody LBP, převzato z [8].

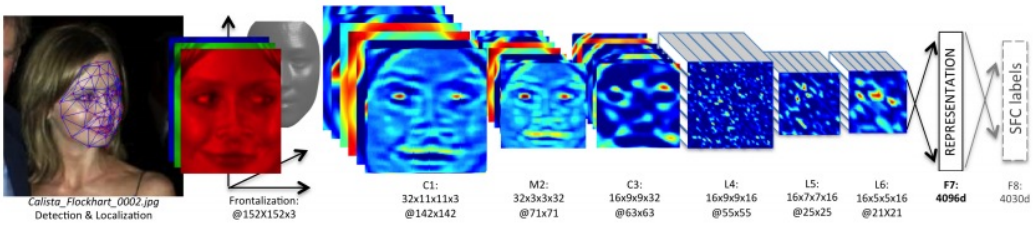
Tuto metodu dále rozšířil Ahonen [7], který rozdělil LBP obrázek na několik lokálních oblastí a vyčíslil histogram od každé z nich. Příznakový vektor pak vznikl spojením všech lokálních histogramů, pojmenovaných jako Local Binary Patterns Histograms.

2.2 Současné metody

V současné době se v oblasti rozpoznávání obličeje ukazuje síla hlubokého učení a neuronových sítí. Následující metody patří k tomu nejlepšímu, co bylo za posledních pár let publikováno. Zároveň tyto tzv. "State of the art"metody ukazují, že zájem o danou problematiku mají i velké korporace.

2.2.1 DeepFace

DeepFace je metoda vyvinutá výzkumnými pracovníky společnosti Facebook, představená v roce 2014 [10]. Používá 9-ti vrstvou konvoluční neuronovou síť obsahující více než 120 milionů parametrů a oproti standartním konvolučním sítím nesdílí váhy a obsahuje tzv. lokálně propojené vrstvy. Jedním z velkých vylepšení je také zarovnání obličeje do přímého směru, což je uskutečněno sofistikovanou metodou 3D modelování. K trénování využila společnost data ze své sociální sítě od více než 4000 osob. Vznikl tedy rozsáhlý dataset s přibližně 4 miliony obrázků, nazvaný Social Face Classification (SFC) dataset. Kvalitu autoři otestovali na známých datasetetch Labeled Faces in the Wild (LFW) a YouTube Faces (YTF). Při verifikaci obličeje jejich metoda dosáhla přesnosti 97.35% u LFW a 91.4% u YTF.

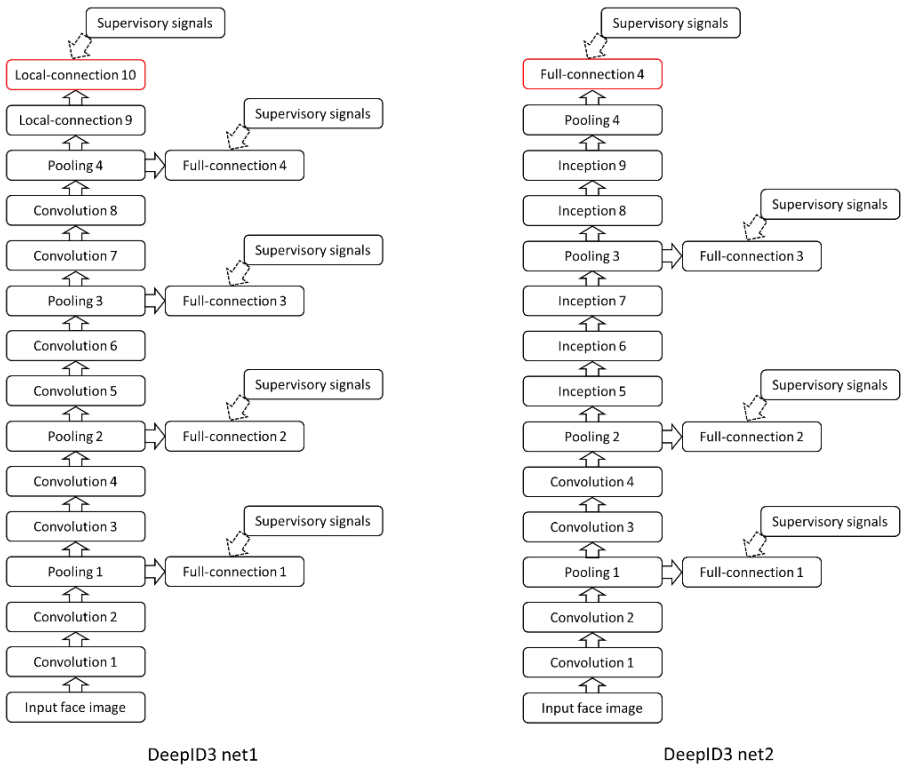


Obrázek 2.5: Architektura neuronové sítě a postup metody DeepFace, převzato z [10].

Na začátku algoritmu je v obraze detekována a oříznuta oblast obličeje. Pomocí metod 3D modelování je obličej zarovnán do přímého směru a následně předán konvoluční síti jako RGB obraz. Počátek sítě je dán třemi vrstvami, po řadě konvoluční-poolingová-konvoluční, následují tři lokálně propojené a dvě plně propojené vrstvy. Na výstupu je obličej reprezentován příznakovým vektorem o velikosti 4096.

2.2.2 DeepID3

Metodu DeepID3 vyvinuli na univerzitě v Hong Kongu v roce 2015 [11]. K získání příznaků využili dvě architektury konvoluční neuronové sítě, jedna z nich vychází z VGG Net [12] a druhá z GoogLeNet [13]. Dimenze příznakového vektoru je potom snižována technikou PCA. Za účelem naučení kvalitnějších příznaků nejsou sdíleny váhy ve vzdálenějších vrstvách sítě a je přidána informace od učitele do některých vnitřních vrstev. Každé poolingové vrstvě předchází dvě konvoluční vrstvy, což také zlepšuje kvalitu příznaků a navíc se tím redukuje počet parametrů. Poslední dvě konvoluční vrstvy byly v první síti nahrazeny lokálně propojenými vrstvami. Druhá síť nahrazuje konvoluční vrstvy na konci sítě tzv. inception vrstvami. Ve všech vrstvách vyjma poolingových byla zvolena aktivační funkce ReLU.



Obrázek 2.6: Architektury konvolučních neuronových sítí metody DeepID3, převzato z [11].

Sítě byly následně trénovány na 25 podoblastech obličeje s více než 300 tisíci trénovacími příklady získaných kombinací datasetu CelebFaces [14] a WDRef. Příznaky všech podoblastí jednoho obrázku jsou pak spojeny do jednoho vektoru

o velikosti přibližně 30 000. Technikou PCA je dimenze redukována přibližně na číslo 300 a rozpoznávání je pak uskutečněno metodou Joint Bayesian. Společně s oběma sítěmi dosáhli přesnosti verifikace 99.53% na datasetu LFW.

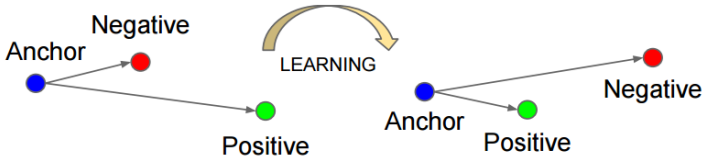
2.2.3 FaceNet

Metodu FaceNet představili v roce 2015 výzkumní pracovníci společnosti Google [15]. Je založena na mapování obrázku do kompaktního Euklidovského prostoru, kde vzdálenost přímo koresponduje s podobností obličejů. Ve chvíli kdy se systém naučí tento Euklidovský prostor, jakýkoli obrázek může být reprezentován jako bod v tomto prostoru a k úlohám klasifikace nebo verifikace lze pak použít standartní techniky strojového učení. Opět je základem konvoluční neuronová síť a její trénování je uskutečněno pomocí tzv. tripletů. Triplet je v tomto případě trojice obrázků, která byla vytvořena výběrem libovolného obličeje (Anchor) a k němu shodného (Positive) a různého (Negative). Tyto trojice jsou přitom vybírány během trénování inovativní tripletovou metodou, založenou na tzv. "hard negative/positive mining"přístupu, při kterém jsou za běhu hledány nejhorší/nejlepší případy z trénovací množiny, vyhodnocené validací.

Tripletová ztráta je dána jako

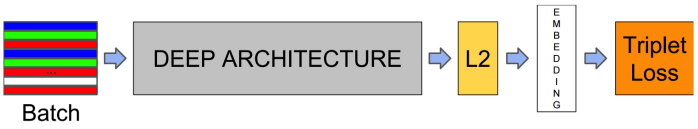
$$\sum_{i}^{N} \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right], \tag{2.16}$$

kde $f(x_i^a), f(x_i^p), f(x_i^n)$ jsou příznakové vektory tripletu v Euklidovském prostoru, α je hranice mezi pozitivním a negativním párem a N je celkový počet trénovacích tripletů.



Obrázek 2.7: Ilustrace tripletové ztrátové funkce, která během učení minimalizuje vzdálenost mezi dvojicí Anchor-Positive a maximalizuje vzdálenost mezi Anchor-Negative, převzato z [15].

Při trénování autoři porovnávají dvě architektury konvoluční sítě, první vychází z modelu ZF Net [16] a druhá z modelu GoogLeNet [13]. Experimentovali s datasety o různých velikostech (2,6 až 260 milionů obrázků). Finální trénovací sada obsahuje 100 až 200 milionů obrázků od přibližně 8 milionů různých osob. V každém obrázku byl detekován a oříznut obličej a následně byla změněna velikost s ohledem na vstupní požadavky konvoluční sítě. Velikost vstupního obrazu se pohybuje v rozmezí od 96×96 do 224×224 pixelů.



Obrázek 2.8: Struktura metody FaceNet, převzato z [15].

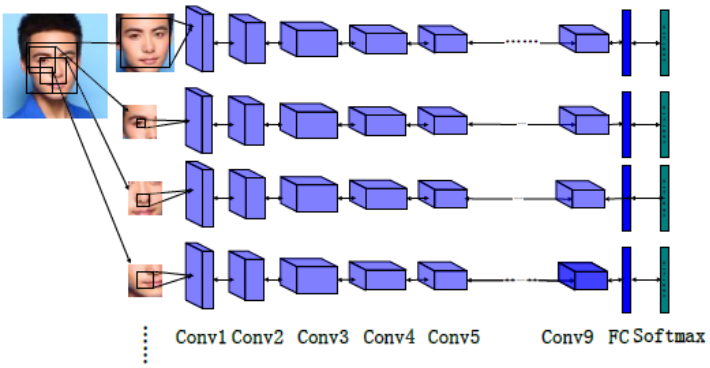
Výhodou této metody je vysoká efektivnost, ke kvalitnímu rozpoznávání stačí, aby reprezentace obrázku (příznakový vektor) měla velikost 128 bajtů. Oproti předchozí metodě nepoužívá žádné techniky pro zarovnání obličeje, je sama o sobě invariantní vůči různým pózám a různým světelným podmínkám. Testováním autoři dosáhli přesnosti 99.63% na datasetu LFW a 95.12% na datasetu YTF.

2.2.4 Baidu

Tuto metodu popsal v roce 2015 čínský výzkumný institut hlubokého učení Baidu Research [17]. Kombinuje dva přístupy zvané multi-patch CNN a redukce dimenze příznakového prostoru, čímž získává velmi diskriminativní příznakové vektory. Autoři se mimo jiné zaměřují na to, jak ovlivňuje velikost datasetu výsledné rozpoznávání. Podle jejich experimentů lze navýšením počtu tváří a identit kdykoli zvýšit přesnost rozpoznávání.

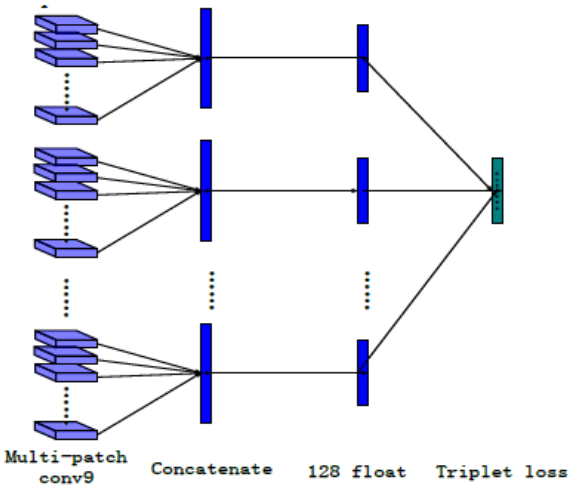
Základem je 9-ti vrstvá konvoluční síť určená pro tradiční klasifikaci do tříd. Vstupem je RGB obraz detekovaného a zarovnaného obličeje do přímého směru. Vstupní obraz je dále rozdělen na překrývající se oblasti (multi-patches), zaměřené na významné části obličeje (oči, nos, rty, apod.) a na každou zvlášť je aplikována stejná konvoluční síť jako v případě celého obličeje. Jednotlivé sítě jsou trénovány odděleně a dílčím výstupem každé z nich je příznakový vektor. Celkový výstup je potom vytvořen jednoduchým napojením dílčích výstupů do

jednoho příznakového vektoru o vysoké dimenzi. K redukci dimenze a zvýšení diskriminativnosti je potom použita tripletová ztrátová funkce jako v případě 2.2.3, jenž snižuje vzdálenost příkladů shodné identity a naopak zvyšuje vzdálenost příkladů různých identit.



Obrázek 2.9: Struktura konvoluční neuronové sítě s technikou multi-patch, převzato z [17].

Pro trénování byl vytvořen dataset shromažďováním obrázků celebrit z internetu, ale byly z něj vymazány osobnosti, které jsou zároveň v datasetu LFW. Výsledkem je dataset obsahující 1.2 milionu obrázků od přibližně 18 000 osob. Při experimentech bylo zjištěno, že lokální oblasti jsou mnohem více robustní vůči pózám a výrazům obličeje. Testování ukázalo, že tato metoda na datasetu LFW svou kvalitou překonává jak DeepFace tak FaceNet dosažením úspěšnosti verifikace 99.77% .



Obrázek 2.10: Učení s tripletovou ztrátovou funkcí a redukce příznaků, převzato z [17].

2.3 Databáze obličejů

K trénování identifikačního systému je samozřejmě potřeba získat dostatečně velké množství snímků s obličeji. Jednou z možností jak to provést je vytvořit si vlastní databázi. Výhodnější je ale použít již nějakou existující, neboť v dnešní době jsou takovéto databáze velice rozsáhlé a obsahují různorodá data, získaná například shromažďováním obličejů z internetu. Z provedené rešerše byly vybrány databáze popsané v následujících podkapitolách. Výběr vhodné databáze pro vlastní návrh bude diskutován v 4.2.

2.3.1 AR Face Database

Databázi AR vytvořili Aleix Martinez a Robert Benavente koncem 90. let [18]. Obsahuje více než 4000 obrázků od 126 lidí (70 mužů a 56 žen). Od každé osoby bylo pořízeno několik přímých pohledů s různými výrazy ve tváři, různými světelnými podmínkami a různými doplňky (sluneční brýle, šátek, atd.). Každá osoba byla navíc vyfocena ve dvou různých obdobích (shodná sada snímků), které od sebe dělí 14 dní.

Tato databáze je volně dostupná, ale pouze pro akademické účely. Pro experimenty v úloze verifikace tváře ji využili například autoři referenčního článku [19].

Obrázek 2.11: Příklad realizace jedné osoby z AR databáze. Horní řádek koresponduje s prvním obdobím, spodní s následujícím obdobím [18].

2.3.2 AT&T The Database of Faces

Databáze AT&T je formálně známá spíše jako ORL, byla vytvořena ve laboratořích AT&T v Cambridge a obsahuje obrázky tváří pořízených mezi roky 1992 a 1994 [20]. Skládá se z 10 různých obrázků od každé ze 40 osob, celkově jich je tedy 400.

U několika osob byly snímky pořízeny v různých obdobích s variací světelných podmínek, výrazů ve tváři a doplňků. Všechny tváře byly vyfoceny před tmavým pozadím s důrazem na zachování přímého pohledu a v databázi jsou uloženy jako šedotónový obraz. Stejně jako AR byla tato databáze použita v [19].

Obrázek 2.12: Příklad realizace jedné osoby z AT&T databáze [20].

2.3.3 CelebFaces Attributes

CelebFaces Attributes (CelebA) je rozsáhlá databáze s více než 200 000 (202 599) obličeji celebrit [14]. Obrázky tváří náleží 10 177 osobám. Ke každé osobě existuje kolem 20 realizací (obrázků). Příklady v databázi zahrnují velké změny jak v pózách obličeje tak v pozadí. Data jsou navíc velice pečlivě anotována.

Obrázek 2.13: Příklady tváří celebrit z databáze CelebA, od každé osoby jedna realizace [14].

2.3.4 Face Recognition Data

Tuto databázi vytvořil Libor Spacek z univerzity v Essexu [21]. Obsahuje 20 obrázků od každé z 395 osob, celkem tedy 7900. Zastoupeni jsou muži i ženy různého rasového původu. Věk osob se pohybuje v rozmezí 18 až 20 let, ale najdou se zde i starší. Některé osoby se od ostatních odlišují brýlemi nebo svými vousy.

Databáze je rozdělena do několika podmnožin, které se různě liší, např. pozadím (barva, homogennost). Jedna podmnožina je využita speciálně na různé výrazy ve tváři. Autor tuto databázi volně poskytuje jen pro výzkumné účely.

Obrázek 2.14: Příklad realizace jedné osoby databáze Face Recognition Data [21].

2.3.5 FaceScrub

Databáze FaceScrub byla vytvořena s využitím přístupu pro automatickou detekci obličeje ve veřejných obrázcích na internetu, což je blíže popsáno Stefanem Winklerem v [22].

Celkem obsahuje 106 863 obrázků, které náleží 530 osobám (265 mužů a 265 žen). Od každé osoby je v databázi průměrně kolem 200 realizací. Všechny obrázky byly získány z internetu jako tzv. "hrubá data", následovala automatická detekce obličeje, a poté manuální kontrola a čištění výsledků.

Obrázek 2.15: Příklady z databáze FaceScrub, na obrázcích jsou dvě celebrity (Lexi Ainsworth a Matt Damon) a od každé je vyobrazeno 12 realizací [22].

2.3.6 FEI Face Database

Tato databáze (zkráceně FEI) pochází z Brazílie a její snímky byly pořízeny mezi roky 2005 a 2006 na oddělení umělé inteligence univerzity FEI v Sao Paulu [23].

Obsahuje 14 obrázků od každé z 200 osob, celkem tedy 2800. Všechny snímky jsou pořízeny před bílým homogenním pozadím. Snímky jedné osoby se liší rotací hlavy v rámci úhlu 180°, od levého profilu přes přímý pohled až k pravému profilu, dále různými výrazy obličeje a světelnými podmínkami.

Všechny obličeje náleží studentům a zaměstnancům univerzity FEI, jejichž věk je v rozmezí 19 až 40 let. Zastoupení mužů a žen je v poměru 1:1, v obou případech tedy 100. Databáze je dostupná pouze pro výzkumné účely.

Obrázek 2.16: Příklad realizace jedné osoby databáze FEI [23].

2.3.7 FERET

Databáze FERET byla shromažďována celkem v 15 cyklech v období mezi roky 1993 a 1996 [24]. Hlavní podíl na tom mají Dr.Wechsler a Dr. Phillips z univerzity George Mason ve Virginii (USA). Celkem obsahuje 14 126 obrázků, které náleží 1199 osobám. Každá osoba byla vyfocena vícekrát (5 až 11 realizací) v různých obdobích, některé například až po několika letech. U jedné osoby tedy mohou existovat podstatné rozdíly v jejím vzhledu.

Obrázek 2.17: Příklad realizace jedné osoby databáze FERET [24].

2.3.8 Labeled Faces in the Wild

Tato databáze je zkráceně nazývána LFW a obsahuje více než 13 000 obrázků tváří získaných z webu, celkem od 5749 lidí [25]. Každá tvář byla označena jménem dané osoby a 1680 lidí je v databázi zachyceno na dvou nebo více snímcích. Na "hrubá data"z webu byl aplikován detektor tváří Viola-Jones.

Databáze se dělí na čtyři různé soubory, první obsahuje originální obrázky, další tři obsahují stejná data, ovšem nějakým způsobem upravená (oříznutí, zarovnání, atd.). Zarovnaná či oříznutá data poté dávají lepší výsledky než originální.

Obrázek 2.18: Příklady z databáze LFW, 3 realizace od 10 osob [25].

2.3.9 SCface

Obrázky této databáze byly pořízeny na elektrotechnické fakultě univerzity v Záhřebu, v neupravených vnitřních prostorech s využitím pěti videokamer různých kvalit [26]. Celkem obsahuje 4160 obrázků (viditelné a infračervené spektrum), na nichž je zachyceno 130 osob. Od každé osoby je v databázi 9 realizací. Různé kvality videokamer simulují podmínky reálného světa a zajišťují robustnost při testování kvality rozpoznávacího nebo identifikačního systému. Databáze je dostupná pro komunity zabývající se výzkumem.

Obrázek 2.19: Příklad realizace jedné osoby databáze SCface [26].

2.3.10 YouTube Faces

YouTube faces je databáze obličejů získáných z videí portálu YouTube [27]. Jako svůj vzor používá databázi LFW a hledá totožné osoby ve videích.

Obsahuje celkem 3425 videí s 1595 různými lidmi. Ke každé osobě existují v průměru 2 videa (minimálně 1, maximálně 6), nejkratší má 48 a nejdelší 6070 snímků (framů). Průměrná délka videa je 181.3 snímků. Podobně jako v LFW existuje ke každému videu označení (label) indikující identitu dané osoby.

Obrázek 2.20: Příklad z databáze YouTube faces, Woody Allen na snímcích jednoho videa [27].

3 Umělé neuronové sítě v úloze rozpoznávání

Umělá neuronová síť (neuronová síť) je jedním z přístupů, jenž se snaží napodobit činnost lidského mozku. V lidském mozku nicméně probíhají složité paralelní, nelineární procesy, které dosud nebyly dostatečně vědecky vysvětleny a popsány. Proto se i dnes oblast neuronových sítí neustále vyvíjí a zároveň se hledají nové přístupy, které by zmíněné složitosti mozku namodelovaly co nejlépe. Aplikace neuronových sítí byla rozšířena zejména v několika posledních dekádách, díky rozvoji softwarových a hardwarových technologií. Jejich architektura se skládá z vrstev neuronů, které umožňují tzv. mapovat vstupy a výstupy do určitého metrického prostoru dle zvoleného učícího algoritmu. Během let byla vyvinuta řada modelů, z nichž nejvýznamnější a nejčastěji využívaný je vícevrstvý perceptron. Potenciál neuronových sítí může být uplatněn v různých úlohách, příkladem můžou být aproximační metody, regresní metody nebo metody klasifikace. Jelikož je tato práce zaměřena na klasifikaci, budu se v následujících odstavcích soustředit právě na tuto oblast.

3.1 Historie

První kroky umělých neuronových sítí vedou až do roku 1943 a jsou spojeny se jmény Warren McCulloch a Walter Pitts. Neurofyziolog a matematik, kteří v té době představili umělé neurony s prahem [28]. Ty fungovaly tím způsobem, že došlo k jejich aktivaci (vyslání elektrického impulzu) jen tehdy, pokud vstup do neuronu byl větší než hodnota prahu.

$$y = \begin{cases} 1, & u \geq \theta \quad (3.1a) \\ 0, & u < \theta \quad (3.1b) \end{cases}$$

Přičemž y je výstup neuronu, u je vstup do neuronu a θ prahová hodnota.

V roce 1949 se zrodilo vůbec první pravidlo pro učení neuronových sítí, nazvané *Hebbovo pravidlo (Hebb's rule)*, navržené psychologem Donaldem Hebbem [29]. Jeho předpokladem byl fakt, že mezi dvěma sousedícími současně aktivními neurony by mělo být zesíleno jejich spojení.

$$w_{ij} = x_i x_j \quad (3.2)$$

Vztah (3.2) je jednou z možných formulací Hebbova pravidla, kde w_{ij} značí váhu spojení mezi neurony i a j, x_i (x_j) je vstup do neuronu i (j).

V roce 1958 přichází Frank Rosenblatt s *perceptronem*, prvním modelem neuronu a zároveň algoritmem pro rozpoznávání vzorů (příznakových vektorů) [30]. Je založený na učení s učitelem a využívá jednoduché sčítání a odčítání. Jeho první implementace byla vydána samostatně ve formě softwaru, větší úspěch se však dostavil později, kdy byl algoritmus implementován do hardwaru postaveného na míru. Tím byl například *Mark 1 perceptron*, stroj navržený pro rozpoznávání obrazu (konkrétně znaků), propojený s fotografickým přístrojem, který pořizoval snímky o rozměrech 20 x 20 pixelů. Dva roky po uvedení perceptronu bylo představeno zařízení modelující jednovrstvou neuronovou síť zvané *ADALINE*. Tento model byl navržen profesorem Stanfordské univerzity Bernardem Widrowem a jeho studentem Tedem Hoffem.

Kolem počátku 70. let vývoj stagnoval, což bylo zapříčiněno dvěma tehdejšími problémy spojenými s neuronovými sítěmi. Prvním byly omezené možnosti perceptronu, který dokázal řešit pouze lineárně separabilní úlohy a nebyl schopen namodelovat například funkci XOR. Druhým problémem byl nedostatek výpočetního výkonu, který neuronové sítě vyžadovaly.

Během několika dalších let prošel hardware i software významným vývojem a s ním logicky i teorie neuronových sítí. Zcela zásadním posunem vpřed byl příchod algoritmu *backpropagation*, který si dokázal poradit s problémem XOR a s rychlostí trénování vícevrstvých sítí. Poprvé ho uvedl v roce 1975 Paul J. Werbos ve své disertační práci. Díky tomu se začaly objevovat další různé typy

neuronových sítí jako samoorganizující se mapy, známé také pod pojmem Kohonenovy mapy. Ty se řadí do skupiny samoučících se sítí (učení bez učitele). Dále se objevily například asociativní sítě nebo RBF sítě.

Na nějaký čas se neuronové sítě opět ukryly do pozadí. V té době je zastínily klasické metody strojového učení jako lineární klasifikátory, SVM a jiné. Oblibu si získaly zejména díky tomu, že byly oproti sítím rychlé a méně náročné na výpočetní výkon. Do popředí se neuronové sítě znovu dostaly až s možností výpočtu na grafických procesorech (GPU), čímž se několikanásobně zvýšila rychlost jejich trénování. Oproti klasickým CPU jednotkám mají GPU jednotky výrazně větší počet jader a větší propustnost. Díky tomu jsou vhodnější k násobení matic o velkých rozměrech a hodí se i pro práci s neuronovými sítěmi.

Z blízké historie lze za zásadní milník považovat například rekurentní neuronové sítě vyvinuté mezi roky 2009 a 2012 švýcarskou výzkumnou institucí *Swiss AI Lab IDSIA*. Jejich sítě se ukázaly jako velice kvalitní například při rozpoznávání ručně psaných číslic známého datasetu MNIST nebo při rozpoznávání dopravních značek. Velký úspěch zaznamenal také Alex Krizhevsky s kolegy, jenž v roce 2012 vytvořil konvoluční nerunovou síť, která dominovala v soutěži ILSVRC na datasetu *ImageNet* [31].

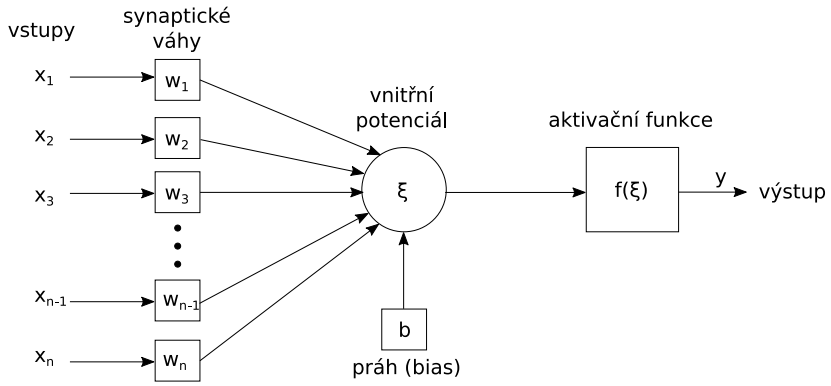
V dnešní době jsou neuronové sítě využívány v různých vědních oborech a průmyslových odvětvích. Mimo jiné také ke klasifikaci obrazových dat, což je předmětem této práce.

3.2 Matematický model neuronu

Základním stavebním prvkem biologických neuronových sítí je neuron. Biologický neuron je specializovaná nervová buňka, schopná přenášet, zpracovávat a ukládat informace. Důležité části buňky jsou z tohoto pohledu synaptická spojení, tělo buňky (soma) a axon.

V teorii umělých neuronových sítí je biologický neuron jakožto základní stavební prvek nahrazen jeho matematickým modelem (umělým neuronem), který je poměrně zjednodušen.

Obecně do těla umělého neuronu vstupuje několik reálných vstupů, které dohromady tvoří vstupní vektor $x = [x_1, \ldots, x_n]$. Každý vstup je ohodnocen synaptickou váhou, dohromady $w = [w_1, \ldots, w_n]$. Jednotlivé váhy mohou být jak kladné tak záporné a reprezentují synaptická spojení. Dalším vstupem neu-



Obrázek 3.1: Model neuronu.

ronu je konstantní hodnota prahu b, též nazývaná jako bias. Pro zjednodušení může být práh začleněn do vztahu přidáním $x_0 = 1$ do vstupního vektoru x, tedy $x = [1, x_1, \ldots, x_n]$. Potom lze práh považovat za váhu $w_0 = b$, tedy $w = [b, w_1, \ldots, w_n]$. V těle neuronu je váženým součtem vstupů získán jeho vnitřní potenciál ξ.

$$\xi \;=\; \sum_{i=0}^{n} w_i x_i \;=\; \sum_{i=1}^{n} w_i x_i \;+\; b \tag{3.3}$$

Pokud jeho hodnota překročí prahovou hodnotu b, indukuje (aktivuje) se výstup y, který reprezentuje impuls axonu. Indukce (aktivace) výstupu je realizována tzv. aktivační přenosovou funkcí $f(\xi)$, která provádí nelineární transformaci vnitřního potenciálu na reálnou hodnotu. Jde tedy o zobrazení z $\{R\}^n \to R$. Jednou z nejjednodušších aktivačních funkcí je ostrá nelinearita, která je dána vztahem

$$f(\xi) = \begin{cases} 1, & \xi \geq 0 \\ 0, & \xi < 0\,. \end{cases} \tag{3.4}$$

Celkový výstup neuronu y můžeme vyjádřit jako

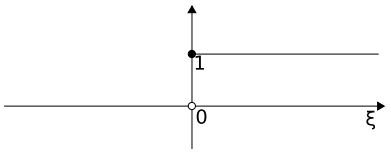
$$y \;=\; f(\xi) \;=\; f\left(\sum_{i=0}^{n} w_i x_i\right)\,. \tag{3.5}$$

V geometrii lze umělý neuron obecně chápat jako nadrovinu, která dělí vstupní prostor na dva poloprostory. Váhy jsou koeficienty této nadroviny a její rovnici můžeme zapsat ve tvaru

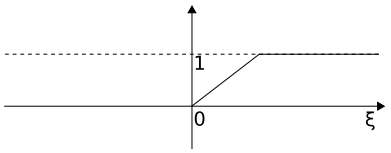
$$\sum_{i=0}^{n} w_i x_i \;=\; 0\,. \tag{3.6}$$

Vstupní vektor x si představíme jako bod v prostoru o stejné dimenzi jako nadrovina. Pokud tento bod leží nad touto nadrovinou nebo přímo na ní, platí nerovnost $\sum_{i=0}^{n} w_i x_i \geq 0$ a výstup neuronu $y = 1$. Pokud leží pod ní, platí nerovnost $\sum_{i=0}^{n} w_i x_i < 0$ a výstup $y = 0$.

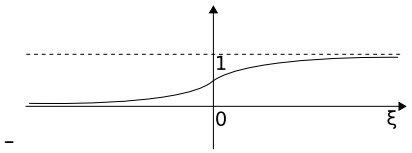
Aktivačních funkcí existuje celá řada. V úlohách, kde nastavujeme parametry sítě přímým výpočtem, můžeme použít i nespojité, nediferencovatelné funkce jako je skoková funkce nebo saturovaná lineární funkce. Naopak tam, kde chceme použít učící algoritmy založené na gradientních metodách, by měly být aktivační funkce diferencovatelné. Diferencovatelnost splňují například logistická sigmoida nebo hyperbolický tangens. Průběhy těchto základních aktivačních funkcí jsou znázorněny na následujících obrázcích.



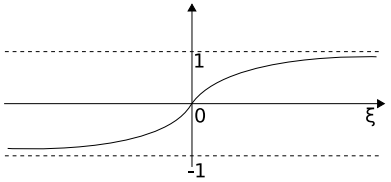
Obrázek 3.2: Skoková funkce.



Obrázek 3.3: Saturovaná linerární funkce.



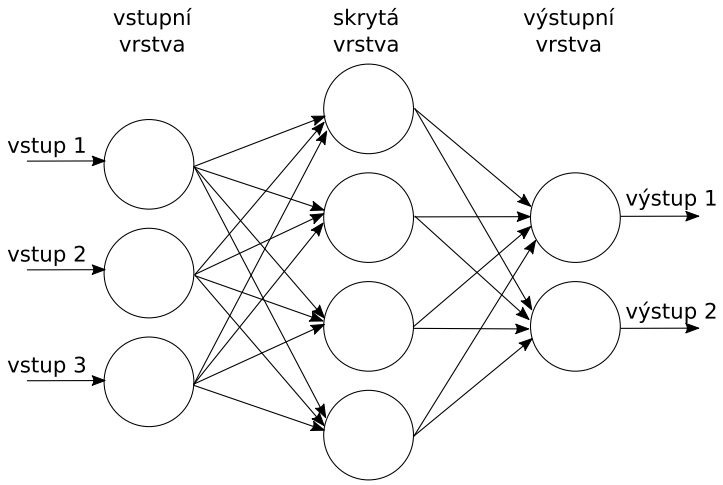
Obrázek 3.4: Sigmoidální funkce.



Obrázek 3.5: Hyperbolický tangens.

3.3 Vícevrstvá architektura

Vícevrstvá architektura vychází z modelu vícevrstvého perceptronu a je nejpoužívanějším typem neuronových sítí. Je tvořena větším množstvím umělých neuronů, které jsou topologicky uspořádány do vrstev. Zatímco u neuronů v rámci jedné vrstvy neexistují žádná spojení, neurony sousedních vrstev jsou plně propojeny a komunikují mezi sebou. Spoje mezi neurony jsou orientované a ohodnocené vahou, přesně jak je tomu u základního modelu neuronu. První vrstva se nazývá vstupní a poslední výstupní. Jakákoli vrstva mezi těmito dvěma se pak nazývá skrytá. Někdy se tento typ sítí označuje také jako dopředná síť, neboť signál se šíří jedním směrem od vstupu k výstupům.



Obrázek 3.6: Příklad třívrstvé architektury.

Během svého nasazení prochází síť několika fázemi. První z nich je fáze učení (adaptační dynamika), během které se nastavují parametry sítě tak, aby odpovídala požadovanému chování. Ve výjimečných případech může být během této fáze změněna i topologie. V další fázi jsou produkovány výstupy sítě na základě předkládání neznámých vstupů (aktivní dynamika).

Mimo nejčastěji užívanou dopřednou topologii, existuje celá řada dalších. Mohou se lišit například zavedením zpětné vazby nebo propojením všech neuronů, tj. každý s každým.

3.4 Backpropagation, učení

Fáze učení neuronových sítí je podmíněna učícím algoritmem a ačkoliv jich existuje několik, právě backpropagation je jedním z nejrozšířenějších. Do češtiny je někdy překládán jako algoritmus zpětného šíření chyby.

Ještě než začne fáze učení (trénování), je třeba vybrat vhodnou architekturu sítě, tzn. počet vstupních a výstupních neuronů, počet skrytých vrstev, atd. Počet neuronů ve vstupní a výstupní vrstvě vyplyne z charakteristiky úlohy, složitější bývá odhadnout počet skrytých neuronů ve skrytých vrstvách. Ve většině případů zatím neexistuje pravidlo, jak nalézt optimum. Řešením je postupovat experimentálně, vyzkoušet různé architektury a vybrat tu s nejmenší chybovostí.

Backpropagation spadá do skupiny učení s učitelem, tudíž je to metoda založená na trénování pomocí párů *vstup-očekávaný výstup*. Tyto páry pak společně tvoří trénovací množinu.

Na začátku každého učení je síť inicializována, to spočívá v náhodném nastavení synaptických vah. Často se volí velmi malá čísla např. z intervalu $\langle -0.05, 0.05 \rangle$. Poté síť zpracuje všechna data z trénovací množiny, porovná jejich výstup s informací od učitele (s požadovaným výstupem) a spočte chybu. Ve chvíli, kdy sítí projdou všechny příklady z trénovací množiny, dochází k modifikaci synaptických vah a optimalizaci sítě. K tomu se většinou využívá přístup nazvaný pokles gradientu (gradient descent), váhy jsou iterativně modifikovány a postupně je tím snižována celková chyba. Signál přitom prochází sítí nazpět, podle toho byl také odvozen název zpětná propagace chyby. K optimalizaci se někdy používají i modifikované metody poklesu gradientu, např. AdaDelta, AdaDelta, Adam, Nesterov (NAG) nebo RMSProp, blíže popsané v [32].

3.4.1 Algoritmus backpropagation

1. **Inicializace**, při které se vybere vhodná architektura sítě, provede se počáteční nastavení vah, zvolí se konstanta učení, popř. momentum apod. Jak již bylo popsáno výše, váhy se většinou volí jako náhodná malá čísla z rovnoměrného rozložení s nulovou střední hodnotou.

2. **Dopředné šíření** (forward pass), při kterém projdou všechny prvky trénovací množiny sítí od vstupu na výstup.

   Předpokládá se, že trénovací množina obsahuje páry *vstup-očekávaný výstup*.

Každý vstup projde sítí dle pravidel matematického modelu neuronu, tj. pro každý neuron v každé vrstvě se spočítá jeho výstup, který je zároveň vstupem do vrstvy následující. Výstup jakéhokoli neuronu v síti lze spočítat dle vztahu

$$y_k^v = f\left(\sum_{i=1}^{n} w_{ik}^v y_i^{v-1}\right) , \tag{3.7}$$

kde y_k^v je výstup k-tého neuronu ve vrstvě v, w_{ik}^v je i-tá váha k-tého neuronu ve vrstvě v, y_i^{v-1} je výstup i-tého neuronu v předchozí vrstvě $v-1$ a f aktivační funkce.

Poté co i-tý prvek trénovací množiny projde všemi vrstvami sítě, vyprodukuje se odezva, tj. výstupní vektor y_i a je spočtena chyba e_i mezi očekávaným výstupem y_i' a skutečným výstupem y_i, příkladem může být střední kvadratická chyba daná vztahem 3.8.

$$e_i = (y_i' - y_i)^2 \tag{3.8}$$

3. **Zpětné šíření** (backward pass), při kterém se mění hodnoty vah dle vzniklé chyby a dochází tedy k učení sítě.

   Obecně lze říci, že zjišťujeme jak velkou částí se podílí každá váha sítě na vzniklé chybě a dle toho pak jednotlivé váhy individuálně modifikujeme za účelem snížení této chyby. Vycházíme z předpokladu, že snížení celkové chyby e je dáno poklesem gradientu chybové funkce. Hledáme tedy globální minimum této funkce, což je ideální případ, v reálných úlohách většinou nalezneme jedno z lokálních minim.

$$\nabla e = \left(\frac{\partial e}{\partial w_1}, \frac{\partial e}{\partial w_2}, \ldots, \frac{\partial e}{\partial w_z}\right) \tag{3.9}$$

   Změna každé váhy v síti je pak dána vztahem

$$\Delta w_i = -c\frac{\partial e}{\partial w_i} , \tag{3.10}$$

   kde c reprezentuje konstantu učení.

Vnitřní potenciál k-tého neuronu ve vrstvě v je dán vztahem

$$\xi_k \;=\; \sum_{i=0}^{n} w_{ik}^{v} y_i^{v-1} \;. \tag{3.11}$$

Pro neurony výstupní vrstvy definujeme

$$\delta_k^y = e_k f'\left(\xi_k\right) \;, \tag{3.12}$$

kde e_k je k-tý prvek chybového vektoru a f' je derivace aktivační funkce.

Pro neurony ve skrytých vrstvách definujeme

$$\delta_k^h = f'\left(\xi_k\right) \sum_{l=1}^{L} \delta_l^{v+1} w_{kl} \;, \tag{3.13}$$

kde počítáme sumu přes všechny neurony l následující vrstvy, w_{kl} je pak váha spojující neurony k a l.

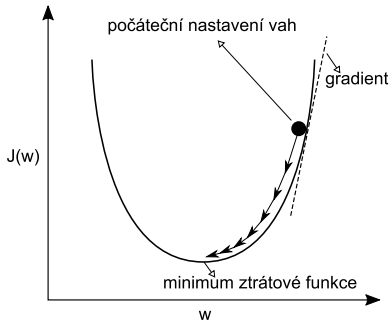
Po spočtení všech hodnot δ přichází na řadu aktualizace vah.

$$\Delta w_{kl}(t) \;=\; c\,\delta_k\, y_k^{v-1} \;+\; m\left(w_{kl}(t) \;-\; w_{kl}(t-1)\right) \tag{3.14}$$

$$w_{kl}(t+1) \;=\; w_{kl}(t) \;+\; \Delta w_{kl}(t) \;, \tag{3.15}$$

kde c je konstanta učení, y_k^{v-1} k-tý neuron předchozí vrstvy a m momentum.

Body 2. a 3. se opakují dokud se chyba neustálí nebo dokud nedosáhneme zastavovacích podmínek, což může být počet iterací.



Obrázek 3.7: Metoda poklesu gradientu, hledání minima ztrátové (chybové) funkce.

Podrobnější informace o teorii neuronových sítí v [33].

3.5 Konvoluční neuronové sítě

Konvoluční neuronové sítě (konvoluční sítě) jsou speciálním typem neuronových sítí. Jejich hlavním účelem je už od doby jejich vzniku rozpoznávání obrazových dat a aplikace v počítačovém vidění.

Historie sahá přibližně do roku 1980, kdy Kunihiko Fukushima vytvořil síť *Neocognitron*. Významného nasazení se ale dočkaly až v posledních několika letech, zejména díky úspěchu Alexe Krizhevského. Ten se svými kolegy vytvořil v roce 2012 konvoluční neuronovou síť, která byla velice úspěšná v soutěži ILSVRC na datasetu *ImageNet*, jak už bylo řečeno v 1.1.

Z poslední doby stojí za zmínku například jejich umělecké využití v mobilní aplikaci *Prisma*. Tato aplikace zpracuje libovolnou fotografii tím způsobem, že vypadá jako by ji namaloval zkušený malíř, jehož styl jsme si vybrali. Těchto efetktů je dosaženo extrakcí vysokoúrovňových příznaků právě pomocí konvolučních sítí.

Obrázek 3.8: Ukázka úpravy fotografie mobilní aplikací Prisma [34].

Jelikož konvoluční sítě využívám ve své práci k identifikaci tváří, bude v následujících odstavcích popsána jejich struktura a funkčnost.

3.5.1 Vlastnosti

Konvoluční neuronové sítě jsou typem dopředných sítí a podobně jako klasické neuronové sítě jsou složeny z neuronů s učícími se váhami. Každý neuron obdrží několik vstupů a ty zpracuje jako vážený součin, který je poté parametrem aktivační funkce. Cílem je opět namapovat vstup (v tomto případě nejčastěji obrázek) do nějakého prostoru o dimenzi n, kde ho lze snadno klasifikovat.

Rozdíl oproti klasickým sítím je pouze na vstupu. Architektura konvolučních sítí předpokládá, že vstupem jsou celé obrázky, jinak řečeno všechny jeho pixely, což nám umožňuje do sítě zakódovat určité "hlubší"vlastnosti. Ve spojení s konvolucí a podvzorkováním (tzv. poolingem) se zajistí větší efektivnost a možnost redukovat počet parametrů sítě. Můžeme to chápat tak, že v síti je několik kopií každého neuronu, tím je umožněn vznik rozsáhlých modelů. Zatímco nastavované parametry (váhy) jsou sdílené a jejich počet zůstává nízký. Počet výpočetních operací je tedy mnohem nižší než při aplikaci klasických neuronových sítí na stejný problém.

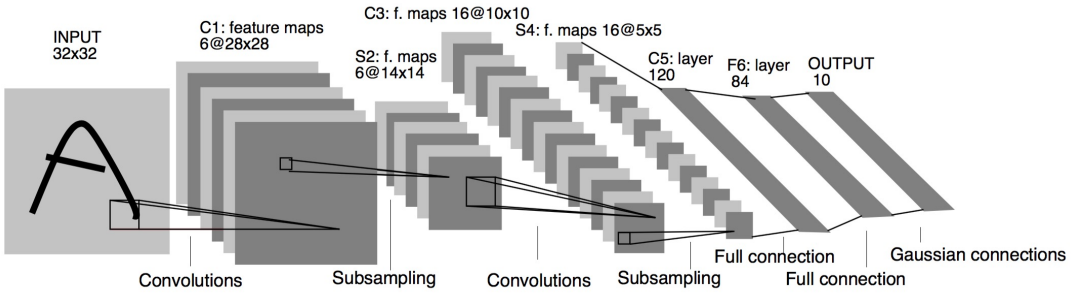
Učení konvolučních sítí probíhá stejně jako u klasických neuronových sítí. Ve většině případů tedy aplikací algoritmu backpropagation, který byl popsán v 1.4. Učenými parametry jsou tomto případě konvoluční jádra.

Obrovskou výhodou konvolučních sítí je, že díky jejich vlastnostem zůstává odezva do jisté míry invariantní vůči geometrickým transformacím vstupních dat.

Více obecných informací o konvolučních sítích v [35].

3.5.2 Architektura

Architektura sítě je tvořena několika vrstvami s odlišnou funkcí, přičemž vstupem sítě je obraz o velikosti $m \times n \times r$, kde $m \times n$ je výška a šířka obrazu a r počet kanálů.



Obrázek 3.9: Typická architektura konvoluční sítě (LeNet), převzato z [36].

Nejdůležitější z vrstev jsou konvoluční vrstva a vrstva podvzorkování (subsamplingová, poolingová), kterých je v síti několik a většinou se pravidelně střídají, ale není to podmínkou. Před výstupem sítě je pak plně propojená (fully connected) vrstva, která zajišťuje úplné spojení s neurony předchozí vrstvy. Tyto vrstvy najdeme v každé konvoluční síti a jsou popsány v následujících podkapi-

tolách. Jako další mohou být použity například ztrátová, aktivační, normalizační, dropout a jiné vrstvy. Ty se ale někdy považují jako součást již zmíněných prvních tří.

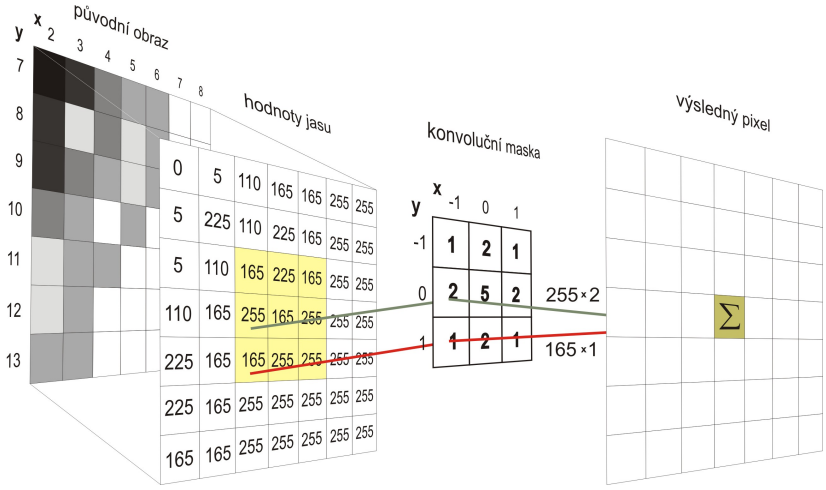
Konvoluční vrstva

Neurony jsou v této vrstvě organizovány do rovin, které sdílí stejné hodnoty vah. Výstupem jsou tzv. příznakové mapy. Všechny neurony dané příznakové mapy provádí stejnou výpočetní operaci (konvoluci), ale v různých podoblastech obrázku. Každý neuron v konvoluční vrstvě je tedy propojený s malou podoblastí zpracovávaného obrázku v předcházející vrstvě (příznakové mapě). Sdílení vah přitom značně zmenšuje počet parametrů, potřebných k naučení sítě.

Váhy jsou zde reprezentovány konvolučními jádry (filtry). Předem se volí jejich počet, velikost a posun. Každé jádro (filtr) potom hledá jiné příznaky (hrany, rohy, atd.) a to tak, že prochází postupně po obrázku a provede konvoluci s danou podoblastí. Počet jader koresponduje s počtem příznakových map v následující vrstvě. Velikost jader by měla být úměrná velikosti obrázku (čím větší vstupní obrázek, tím větší velikost jádra). Posun by měl být volen tak, aby docházelo k překryvu podoblastí při pohybu jádra po obrázku.

Konvoluci počítáme dle vzorce

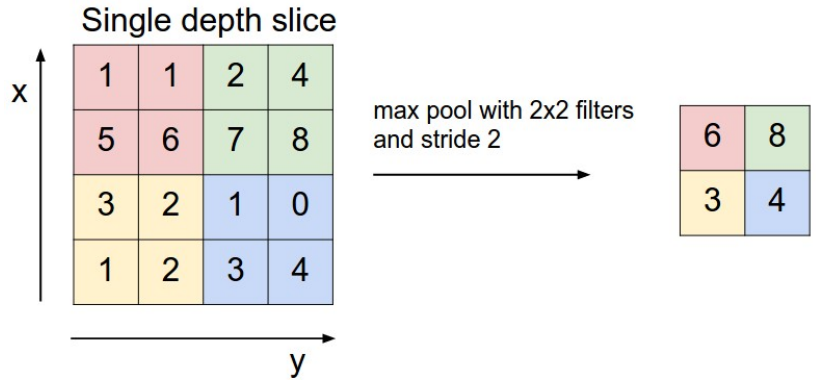
$$(f * h)(x, y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} f(x - i, y - j) \cdot h(i, j) \ . \tag{3.16}$$



Obrázek 3.10: Princip diskrétní konvoluce, převzato z [37].

Vrstva podvzorkování

Vrstva podvzorkování (subsamplingová, poolingová) se běžně vkládá za konvoluční vrstvu a na výstupu obsahuje stejný počet příznakových map jako vrstva předchozí. Jejím hlavním účelem je zmenšit velikost příznakových map vzniklých po konvoluci a tím redukovat celkový počet parametrů a výpočetních operací sítě. Dalším účelem je zabránit tzv. přetrénování sítě a zajistit nalezení lepších, robustnějších příznaků. Poolingová vrstva pracuje nezávisle na každé příznakové mapě, jenž je napojena na odpovídající mapu z předchozí vrstvy. Každý neuron je napojen na určitou podoblast mapy z předchozí vrstvy. Opět se volí velikost a posun jádra, počet je daný počtem příznakových map. Rozdíl oproti konvoluční vrstvě je v tom, že jednotlivé podoblasti se nepřekrývají (dle toho musí být volen posun) a jejich minimální velikost je 2×2. Z těchto podoblastí se následně buď spočte průměr (mean pooling) nebo se vezme maximální hodnota (max pooling), jenž putuje do následující vrstvy. Tímto procesem dochází k tzv. nelineárnímu podvzorkování, zjednodušeně řečeno ke zmenšení velikosti původní příznakové mapy.



Obrázek 3.11: Ukázka max poolingu s jádrem 2×2 a posunem 2, převzato z [35].

Plně propojená vrstva

Plně propojená vrstva (fully connected) následuje za všemi konvolučními a subsamplingovými, může jich být i více za sebou. Vezme úplně všechny neurony předchozí vrstvy a spojí je s každým neuronem ve vlastní vrstvě. Na výstupu je tedy vektor o velikosti n. Pokud se jedná o poslední vrstvu a úlohou je klasifikace do tříd, počet neuronů (dimenze výstupního vektoru) odpovídá počtu tříd.

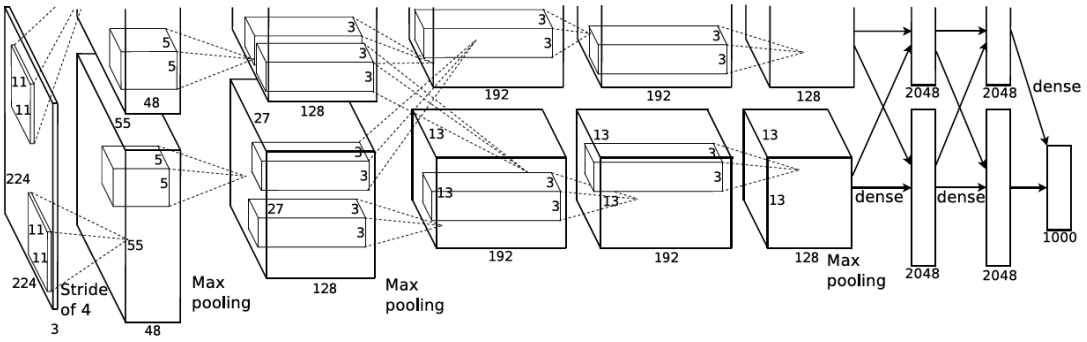
3.5.3 Přehled významných architektur

LeNet

Jedna z prvních úspěšných architektur, kterou v 90. letech vytvořil Yann LeCun [36]. Poprvé byla aplikována na rozpoznávání ručně psaných číslic poštovních směrovacích čísel. Viz obrázek 3.9

AlexNet

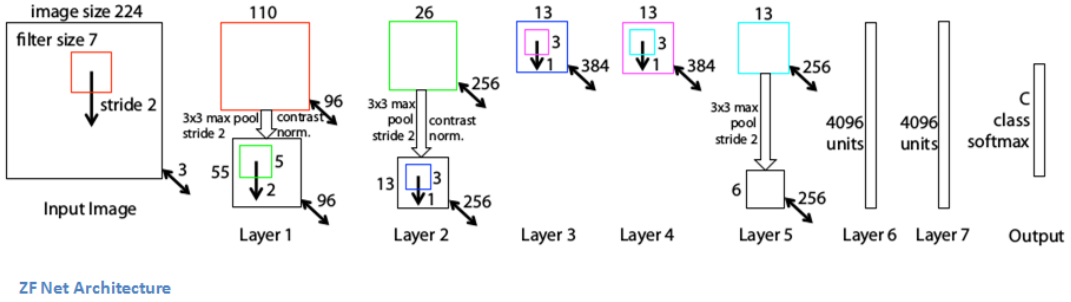
Architektura AlexNet se stala pro využití konvolučních sítí zcela zásadní, po jejím představení následoval obrovský zájem o nasazení konvolučních sítí v počítačovém vidění. Vyvinul ji Alex Krizhevsky společně se svými kolegy a v roce 2012 s ní dosáhl vynikajících výsledků v soutěži ILSVRC (Imagenet Large Scale Visual Recognition Challange), viz [31]. Velmi se podobá architektuře LeNet, ale je hlubší, větší a konvoluční vrstvy jsou zapojené za sebou (není mezi nimi pooligolvá vrstva).



Obrázek 3.12: Architektura AlexNet, převzato z [31].

ZF Net

V roce 2013 vyhráli soutěž ILSVRC s touto architekturou Matthew Zeiler a Rob Fergus, viz [16]. Vychází z architektury AlexNet a vylepšuje ji zvětšením velikosti příznakových map uprostřed sítě a zmenšením posunu a velikosti jader v počáteční vrstvě.V příznakových mapách po konvoluci zůstane více originálních pixelů a tím i "více informace".



Obrázek 3.13: Architektura ZF Net, převzato z [16].

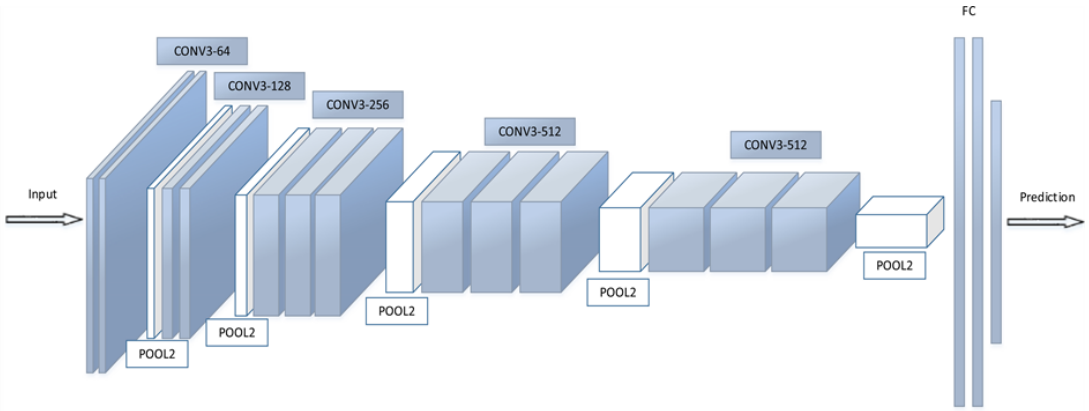
GoogLeNet

GoogLeNet architekturu vytvořila společnost Google, resp. její výzkumný pracovník Christian Szegedy, viz [13]. Vyhrála soutěž ILSVRC v roce 2014 a její hlavní výhodou je, že výrazně snižuje počet nastavovaných parametrů sítě. V porovnání s AlexNet jsou to 4 miliony oproti 60 milionům. Architekura je znázorněna na obrázku 1 v příloze A.

VGGNet

Architekturu VGGNet vytvořili výzkumníci z Oxfordu Karen Simonyan a Andrew Zisserman v roce 2014, viz [12]. Poukazují na to, že dobré výsledky ovlivňuje především hloubka sítě. Experimentovali s několika architekturami s různou hloubkou a nejlepších výsledků dosáhli se sítí, která obsahovala 16 konvolučních vrstev, 5 vrstev podvzorkování a 3 plně propojené vrstvy. Pro získání příznaků použili v celé síti stejnou velikost konvolučních i poolingových jader, konkrétně 3×3, resp. 2×2.

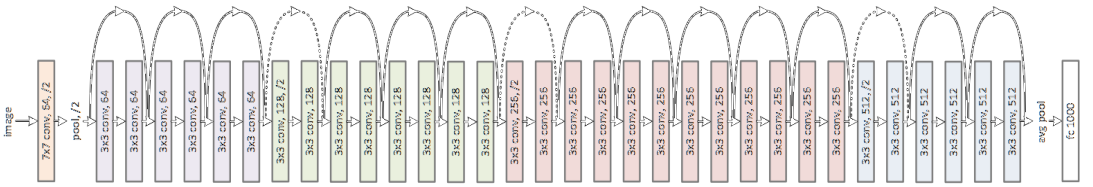
Nevýhodou této architektury je velký počet trénovaných parametrů, kterých je okolo 140 milionů. Nicméně většina těchto parametrů je obsažena v první plně propojené vrstvě a bylo zjištěno, že její odstranění nemá výrazný vliv na kvalitu výsledků. Jejím vynecháním pak dojde ke snížení parametrů sítě.



Obrázek 3.14: Architektura VGG Net s třinácti konvolučními vrstvami, převzato z [38].

ResNet

Reziduální síť vyhrála soutěž ILSVRC v roce 2015 a vyvinul ji Kaiming He se svými kolegy v rámci výzkumu společnosti Microsoft [39]. Tato architektura ignoruje některé vazby, používá normalizaci trénovacích podmnožin (dávek) a zcela úplně postrádá plně propojené vrstvy na konci sítě. Momentálně jde o jeden z nejlepších přístupů.



Obrázek 3.15: Architektura ResNet, převzato z [39].

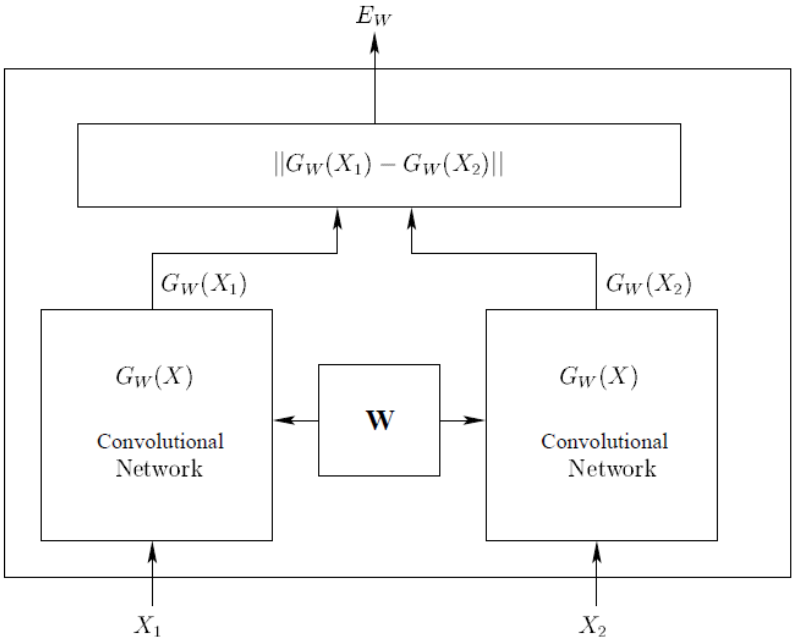
3.6 Siamské neuronové sítě

Siamská neuronová síť (siamská síť) je speciálním typem neuronových sítí a od ostatních se liší především svou strukturou a využitím. Tento přístup představil v 90. letech Yann LeCun se svými kolegy a jeho funkčnost ukázal v úloze ověřování podpisu [40]. Struktura siamské sítě je rozdělena do dvou paralelních větví s totožnou architekturou, přičemž obě větve sdílí shodné parametry (váhy). Každá větev má svůj odlišný vstup, oba vstupy jsou paralelně zpracovány a výstupy obou větví jsou pak na konci sítě spojeny k jejich vzájemnému porovnání. Můžeme to chápat jako dvě samostatné neuronové sítě, jejichž výstupy jsou předány nějaké funkci, která je dále zpracovává (nejčastěji počítá míru podobnosti mezi nimi). Větve tedy počítají příznaky a mapují vstupy do příznakového prostoru a konec sítě binárně rozhoduje o podobnosti těchto vstupů. Míra podobnosti se většinou vyjadřuje na základě Euklidovské vzdálenosti v příznakovém prostoru.

Učící proces je založený na optimalizaci vzdálenosti vstupního páru v příznakovém prostoru. Minimalizuje vzdálenost párů ze stejné třídy a maximalizuje vzdálenost párů z různých tříd. Malou vzdálenost párů ze stejné třídy zajišťuje sdílení vah a shodná architektura obou větví. Kdybychom tedy na vstup dali dva totožné příklady, oba budou mapovány do stejného místa v příznakovém prostoru, tj. vzdálenost mezi nimi bude nulová.

Jak již z výše uvedeného vyplývá, siamské sítě se obecně využívají v úloze nalezení podobnosti a vzájemného vztahu mezi dvěma porovnatelnými vstupy. Výborně se tedy hodí i na problematiku této práce, tj. na identifikaci (verifikaci) tváře porovnáním referenčního a neznámého obrazu. Proto byl tento přístup zvolen jako hlavní pro další experimenty.

Obrázek 3.16 ukazuje architekturu siamské sítě, která je tvořena konvolučními sítěmi. Kde X_1 a X_2 je vstupní pár z trénovací množiny, W jsou sdílené váhy, $G_W(X)$ je shodná architektura obou konvolučních sítí, $G_W(X_1)$ a $G_W(X_2)$ jsou dva body v prostoru o nízké dimenzi, které byly vytvořeny mapováním vstupů X_1 a X_2. E_W je výsledná energie (např. Euklidovská vzdálenost) mezi X_1 a X_2.



Obrázek 3.16: Siamská architektura s konvolučními sítěmi, převzato z [19].

3.7 Siamské sítě a identifikace obličeje

Následující teorie vychází z článku [19], který v roce 2005 publikovali Yann LeCun, Sumit Chopra a Raia Hadsell. Základní myšlenkou je naučit síť jakýsi podobnostní prostor z trénovacích příkladů, do kterého budou následně mapovány. Tím, že systému budeme postupně předkládat dvojice obličejů stejných a různých osob ho zmíněný prostor naučíme. V něm pak budeme schopni binárně rozhodnout, zda je na obou obrázcích tatáž osoba (identita). V naučeném prostoru pak mohou být porovnávány i příklady, které nebyly nikdy předtím sítí viděny (tváře lidí, které nebyly použity při trénování, učení). Koncepce metody je založená na tom, že ji lze aplikovat na klasifikační problém, kde počet tříd (osob) je vysoký, ale počet příkladů v každé třídě (obličejů jedné osoby) je naopak nízký.

Úkolem je najít funkci, která namapuje vstupní vzory do cílového prostoru, ve kterém je tzv. sémantická vzdálenost vstupů aproximována prostorovou vzdáleností (např. Euklidovskou vzdáleností). Pokud bychom to chtěli popsat formálně, máme dán soubor funkcí $G_W(X)$ parametrizovaný váhami W. Hledáme poté takové nastavení parametru (vah) W, které nám zajistí, že vzdálenost páru ze stejné třídy bude malá a páru z různých tříd bude velká. Síť je proto trénovaná pomocí párů z trénovací množiny a parametry jsou optimalizovány na základě minimalizace

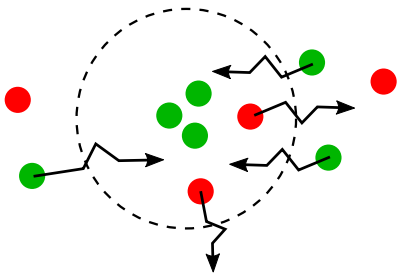
tzv. ztrátové funkce.

V aplikaci identifikace obličeje chceme tedy dle předchozího natrénovat model sítě, který bude produkovat výstupní vektory blízko vedle sebe pro tváře stejné osoby a daleko pro tváře různých osob. Natrénovaný model pak bude schopný určit podobnost i mezi obličeji osob, které nebyly použity při trénovaní (síť je nikdy neviděla). Důležitým aspektem je výběr vhodné architektury, tedy funkce $G_W(X)$. V posledních letech se ukázalo, že vhodným nástrojem (architekturou) pro získávání příznaků z obrázku jsou konvoluční neuronové sítě, které byly vysvětleny v 3.5. Proto byly zvoleny i v tomto případě. V úloze identifikace (verifikace) je navíc vyžadována určitá robustnost vůči geometrickým transformacím vstupu (různým pózám obličeje), což konvoluční sítě splňují.

3.7.1 Obecné vlastnosti

Trénování tohoto modelu je postavené na minimalizaci ztrátové funkce, která je odvozena od modelů založených na energii (EBM - energy based models). Mapování vstupů do prostoru o nižší dimenzi je prováděno nelineárně. Modelování tohoto prostoru může být chápáno jako přiřazování energie $E_W(X_1, X_2)$, každé dvojici (X_1, X_2) z trénovací množiny. Energie je v tomto případě ekvivalentem vzdálenosti. Výhodou modelů založených na energii je skutečnost, že nemusíme odhadovat normalizované pravděpodobnostní rozdělení vstupního prostoru, jako je tomu u pravděpodobnostních modelů (Bayes apod.).

Zásadní roli hraje volba ztrátové funkce. Ta musí zajistit dvě podmínky, energie párů ze stejné třídy musí být malá a naopak energie párů z různých tříd musí být velká. To splňuje tzv. kontrastní ztrátová funkce, která byla použita v modelu této práce.



Obrázek 3.17: Princip učení s kontrastní ztrátovou funkcí, zelené body představují shodné páry jedné identity (osoby), červené body rozdílné páry různých identit (osob) a kružnice prahovou hodnotu m.

Hlavní úkol identifikace obličeje spočívá v binárním rozhodnutí, tj. přijmutí (akceptování) nebo zamítnutí (neakceptování) požadované identity neznámé osoby na obrázku. Kvalita je pak hodnocena na základě dvou měření, procenta špatně přijatých a špatně zamítnutých případů. Cílem je samozřejmě minimalizace obou těchto případů, kdy systém špatně rozhodl.

3.7.2 Funkce pro výpočet energie

Nechť X_1, X_2 je libovolný pár obrázků z trénovací množiny, Y je informace od učitele (label), $Y = 0$ pokud je na obrázku X_1 a X_2 stejná osoba, $Y = 1$ pokud se jedná o různé osoby. W je sdílený vektor parametrů (vah), jenž je předmětem učení. $G_W(X_1)$ a $G_W(X_2)$ jsou body v prostoru o nízké dimenzi vzniklé mapováním vstupů X_1, X_2. Potom se lze dívat na výpočet energie jako na skalární funkci $E_W(X_1, X_2)$, která počítá vzájemný vztah obou z dané dvojice X_1, X_2. Energie se spočte dle vztahu

$$E_W(X_1, X_2) = \|G_W(X_1) - G_W(X_2)\|, \tag{3.17}$$

tedy jako $L2-norma$, ale není to podmínkou, místo $L2$ lze použít i $L1-normu$.

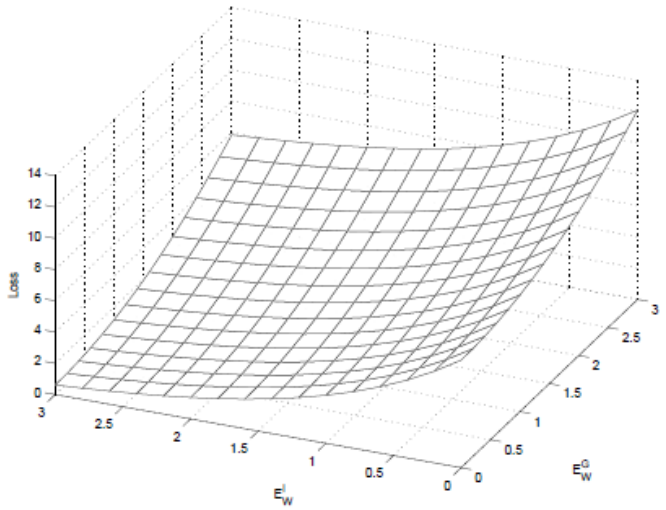
3.7.3 Kontrastní ztrátová funkce

Dle autorů článku [19] je vztah pro kontrastní ztrátovou funkci následující

$$\begin{aligned}
\mathcal{L}(W) &= \sum_{i=1}^{P} L(W, (Y, X_1, X_2)^i) \\
L(W, (Y, X_1, X_2)^i) &= (1-Y)L_G(E_W(X_1, X_2)^i) + YL_I(E_W(X_1, X_2)^i) \\
L(W, Y, X_1, X_2) &= (1-Y)L_G(E_W) + YL_I(E_W) \\
&= (1-Y)\frac{2}{Q}(E_W)^2 + (Y)2Qe^{\frac{-2,77}{Q}E_W},
\end{aligned} \tag{3.18}$$

kde $(Y, X_1, X_2)^i$ je i-tý trénovací příklad, složený z páru obrázků a informace od učitele. Q je horní hranice E_W. L_G, E_W^G je část ztrátové funkce, resp. energie náležící shodnému páru (stejná osoba), L_I, E_W^I pak část ztrátové funkce, resp. energie náležící rozdílnému páru (různé osoby), P je počet příkladů v trénovací množině. L_G a L_I by měli být navrženy tak, aby při minimalizaci L docházelo

ke snižovaní energie mezi shodnými páry a zvyšování mezi rozdílnými páry.



Obrázek 3.18: Graf ztrátové funkce ve 3D, převzato z [19].

Podmínkou je existence jakési hranice (prahu) m, která oděluje shodné a rozdílné páry, viz obrázek 3.17. Formálně pro m platí

$$E_W^G + m < E_W^I \tag{3.19}$$

Platnost těchto vztahů podrobněji ověřují autoři v [19].

4 Návrh identifikačního systému metodou siamských sítí

Základní myšlenkou identifikace je navrhnout systém, kterému na vstupu předložíme dva obrázky s obličeji a na výstupu dostaneme informaci o tom, zda jde o stejnou osobu (identitu) či nikoliv. Komplexní problém rozpoznávání tedy na konci procesu přejde na poměrně jednoduché binární rozhodování. V této kapitole bude implementována identifikace obličeje dle teorie popsané v kapitole 3.

Obrázek 4.1: Příklad negativního páru (vlevo) a vstupního páru (vpravo).

4.1 Pozadí návrhu identifikačního systému

V dnešní době existuje mnoho nástrojů, knihoven a frameworků, které umí pracovat s neuronovými sítěmi a tím nám výrazně ulehčují práci. Dovolují nám jednoduchým způsobem pracovat s datasetem, navrhnout architekturu sítě, nastavit parametry učení apod. Důležitou součástí většiny z nich je také možnost spustit trénování sítě na grafické kartě, která to umožňuje. Trénování je časově náročný proces a procesorové jednotky ho výrazně zpomalují. Jsou totiž omezeny svou architekturou, která pro tento typ výpočtů není stavěná. Proto se postupně

začalo přecházet ke grafickým kartám, jejichž architektura (stovky až tisíce jader) se pro tyto výpočty hodí mnohem více. Příkladem frameworků pro učení neuronových sítí mohou být TensorFlow, Keras, Torch nebo Caffe.

Vzhledem k okolnostem jsem pro svou práci zvolil framework Caffe [41], vyvinutý institutem Berkeley Vision and Learning Center. Hlavním důvodem výběru je fakt, že podporuje architekturu siamské sítě a umožňuje ji implementovat. Dalšími důvody jsou podpora výpočtu jak na procesoru tak na grafických kartách využívajících technologii CUDA (ve spojení s knihovnou cuDNN) a volná dostupnost (open source).

K výpočtům byl použit hardware celonárodní sítě zvané MetaCentrum, kterou spravuje virtuální organizace MetaCentrum VO. Ta poskytuje tyto gridové a cloudové služby zdarma všem akademickým pracovníkům a studentům. Hlavním důvodem této volby byla skutečnost, že MetaCentrum má ve svém portfoliu stroje s vysoce výkonnými grafickými kartami podporující technologii CUDA a umožňuje výpočet na několika strojích najednou.

Veškeré skripty pro vedlejší úkoly jako je úprava obrázků, rozdělení datasetu nebo vizualizace dílčích výsledků byly napsány v jazyce Python. Ten byl vybrán mimo jiné proto, že je kompatibilní s frameworkem Caffe.

4.2 Výběr databáze obličejů

Před samotným návrhem identifikačního systému je nutné získat vhodnou databázi (dataset) trénovacích a testovacích dat. Tato metoda má oproti ostatním výhodu, že nevyžaduje velké množství příkladů od každé třídy, zatímco počet tříd může být vysoký. Nabízejí se dvě možnosti, buď si takovou databázi vytvořit nebo vyhledat nějakou existující. Rozhodl jsem se pro druhou možnost databázi nalézt.

Obecně pro tuto metodu platí, čím větší počet tříd, tím větší kvalita systému. Při hledání tedy upřednostňuji databáze s velkým počtem osob a nižším počtem fotografií od každé z nich, neboť od jedné osoby nám stačí jen několik různých realizací obličeje, konkrétně jednotky až desítky.

Pro trénování i testování vlastního modelu byla vybrána databáze FEI, blíže popsaná v 2.3.6. Byla zvolena z toho důvodu, že obsahuje přiměřené množství dat pro tuto práci a zcela odpovídá jejím požadavkům, tj. relativně velký počet osob (200) a nižší počet realizací od každé z nich (14). Dalším důležitým aspektem,

který vedl k jejímu zvolení byla různorodost v realizaci obličeje jedné osoby. Každá osoba je vyfocena jak z přímého (čelního) pohledu tak z profilu (pohled vlevo i vpravo), což jsou extrémní případy k porovnávání. Takovéto případy se v úloze verifikace neobjevují tak často a to pro mě bylo motivací. Proto jsem se rozhodl navrhnout model právě s touto databází a zjistit, jak kvalitní bude identifikace, pokud vedle ostatních budu porovnávat např. přímý pohled a profil obličeje.

Při návrhu byla kromě FEI paralelně použita také databáze AT&T, popsaná v 2.3.2. Ta posloužila jako výchozí k porovnání kvality navrženého modelu a modelu použitého v [19], z něhož vlastní návrh vychází.

4.3 Tvorba datasetu

V běžné klasifikační úloze s neuronovými sítěmi se dataset obvykle dělí na tři podmnožiny - trénovací, validační a testovací, z nichž každá má odlišnou funkci. Jak už názvy napovídají, trénovací slouží k natrénování (naučení) sítě a validační ke kontrole tzv. přetrénování sítě během jejího učení. Pokud je zjištěno, že v daném okamžiku začíná být síť přetrénovaná, je výhodné proces učení zastavit. Testovací podmnožina pak slouží k vyhodnocení kvality naučené sítě. Aby byly výsledky relevantní, provádí se několik vyhodnocení nad stejným datasetem, ale rozdělení do tří podmnožin se v každém z případů liší. Jednotlivá vyhodnocení se pak zprůměrují. Tento přístup vyhodnocování je nazýván křížová validace (cross-validation).

Jelikož databáze obsahují "syrová"neanotovaná data, bylo nutné je upravit a dataset vytvořit. V rámci úspory paměti byly vytvořené datasety uloženy do formátu LevelDB, který data komprimuje a je podporován frameworkem Caffe.

4.3.1 Předzpracování dat

FEI

Databáze FEI obsahuje barevné obrázky ve formátu JPEG o velikosti 640 x 480 pixelů a v této originální podobě by trénování sítě poněkud ztěžovaly. Proto bylo provedeno několik úprav, které trénování do jisté míry usnadňují. První úpravou bylo oříznutí. Snímky obsahují mnoho nadbytečného, nežádoucího prostoru kolem obličeje, a proto byly tyto plochy odstraněny (konstantně pro všechny

obrázky). Velikost po oříznutí pak činí 320 x 420 pixelů. Takovéto rozlišení je pořád příliš velké a výrazně by nám prodlužovalo dobu trénování. Obecně platí čím menší rozlišení, tím méně výpočtů a tím kratší doba trénování. Při zmenšení, podvzorkování obrázku dochází navíc k žádoucí redukci šumu, na druhou stranu ale také k nežádoucímu úbytku určité informace. Musí se tedy dbát na to, aby zmenšení nebylo příliš radikální.

Další úpravou tedy bylo zmešení velikosti (podvzorkování). Konkrétně na velikost 80 x 105 pixelů. Poslední úpravou bylo převedení obrázku do šedotónu (1-kanálový místo 3-kanálového). Tím se opět redukoval počet matematických operací.

Obrázek 4.2: Příklad předzpracování obrázku databáze FEI. Vlevo originál o velikosti 640x480, vpravo upravený obrázek o velikosti 80x105.

AT&T

Data v této databázi zůstaly beze změny. Obrázky jsou již v originálu šedotónové, uložené ve frormátu PGM a jejich velikost je 92 x 112 pixelů.

4.3.2 Rozdělení datasetu

Trénování a testování siamských sítí probíhá předkládáním shodných (pozitivních) a různých (negativních) párů osob. Tyto páry je nutné nejdříve vytvořit a přidat k nim označení (label), zda se jedná o tutéž identitu nebo ne. Následně je možné provést rozdělení na tři zmíněné podmnožiny - trénovací, validační a testovací. Pokud označíme každý pár jako dvojici (X_1, X_2) a odpovídající label jako Y, pak každá podmnožina datasetu obsahuje seznam trojic (X_1, X_2, Y).

FEI

Databáze FEI obsahuje 14 obrázků od každé z 200 osob. Teoreticky lze tedy vytvořit 39 200 pozitivních a 7 800 800 negativních párů. Vynecháním případu, kdy vedle sebe máme dvě identické fotky ($X_1 = X_2$) změníme počet pozitivních párů na 36 400, počet negativních párů se nemění. Pokud navíc vynecháme případy, kdy se jedná o tentýž pár, ale v prohozeném pořadí (X_1, X_2) *a* (X_2, X_1), dostáváme 18 200 pozitivních, 3 900 400 negativních párů, a právě z těchto hodnot vychází následující rozdělení. Chceme-li systém robustní je nutné dbát na poměr pozitivních a negativních párů při trénování, ten by měl být vyrovnaný, tedy 1:1.

Pro trénovací a validační podmnožinu bylo použito 185 osob, zbylých 15 bylo použito pro testovací podmnožinu. Trénovací sada v tomto případě obsahuje 29 670 párů (14 835 pozitivních/negativních), validační sada 4 000 párů (2 000 pozitivních/negativních) a testovací sada 5 000 párů (1 365 pozitivních, 3 635 negativních). Dataset byl takto zrealizován desetkrát pro následnou křížovou validaci.

FEI	Trénovací	Validační	Testovací
Počet osob	185		15
Pozitivních párů	14 835	2 000	1 365
Negativních párů	14 835	2 000	3 635

Tabulka 4.1: Rozdělení datasetu FEI do 3 základních podmnožin.

AT&T

Rozdělení AT&T koresponduje s článkem [19]. V databázi je vždy 10 obrázků od každé ze 40 osob. Vytvořit lze teoreticky 4 000 pozitivních a 156 000 negativních párů. V tomto případě nebyly vynechány žádné páry a v datasetu se tedy objevují i duplikáty.

Prvních 35 osob bylo použito pro trénovací a validační podmnožinu, pro testovací bylo použito zbylých 5 osob. Trénovací sada obsahuje 6 000 párů (3 000 pozitivních/negativních), validační sada 1 000 párů (500 pozitivních/negativních) a testovací 4500 párů (500 pozitivních, 4 500 negativních). Negativní páry v tes-

tovací sadě byly smíchány i s některými, které byly zahozeny při tvorbě trénovací a validační sady, to znamená, že nebyly použity při trénování.

AT&T	Trénovací	Validační	Testovací
Počet osob	35		5
Pozitivních párů	3 000	500	500
Negativních párů	3 000	500	4500

Tabulka 4.2: Rozdělení datasetu AT&T do 3 základních podmnožin.

4.4 Návrh architektury a trénování

Architektura siamské sítě pro identifikaci obličeje byla obecně popsána v 3.7. Pro její návrh byl použit framework Caffe, ve kterém se architektura definuje jednoduše v textovém souboru.

Zatím neexistuje pravidlo jak sestavit vhodnou architekturu, proto byla na začátku zvolena výchozí a ta byla dále optimalizována. Do obou větví byla dosazena shodná konvoluční síť vycházející z architektury LeNet 3.5.3, sestavená ze 2 konvolučních, 2 poolingových a 3 plně propojených vrstev. Kvalitu architektury ovlivňuje několik faktorů, především počet konvolučních a poolingových vrstev, počet a velikost konvolučních jader, velikost posunu nebo velikost výstupního vektoru. Na základě experimentů s těmito faktory byla vybrána architektura s nejlepšími výsledky, která bude nyní popsána. Pokud si označíme konvoluční vrstvu jako C, poolingovou vrstvu jako P a plně propojenou vrstvu F, pak je nejlepší získaná architektura (v rámci jedné větvě siamské sítě) dána následující posloupností: $C_1 - P_1 - C_2 - P_2 - C_3 - F_1 - F_2 - F_3$. Architektura je znázorněna na obrázku 2 v příloze B.

- C_1 - 20 konvolučních jader o velikosti 5x5, posun 1
- P_1 - velikost jádra 2x2, posun 2
- C_2 - 50 konvolučních jader o velikosti 5x5, posun 1
- P_2 - velikost jádra 2x2, posun 2
- C_3 - 100 konvolučních jader o velikosti 3x3, posun 1
- F_1 - 500 neuronů
- F_2 - 200 neuronů
- F_3 - 50 neuronů

Důležité je zmínit, že na začátku sítě je každý trénovací pár automaticky normalizován z intervalu [0,255] na [0,1]. Pár je dále předán rozdělující vrstvě, která pošle obrázky zvlášť do každé větve a výstupem větví jsou pak příznakové vektory. Na konci sítě se příznakové vektory společně s labelem předají kontrastní ztrátové funkci, která spočte aktuální ztrátu.

Z navržené architektury můžeme pro názornost jednoduše vyčíslit počet neuronů a trénovaných parametrů v jednotlivých vrstvách. Ukažme si to nyní v případě, kdy na vstupu bude obrázek datasetu FEI o velikosti 80 x 105 pixelů. Počet neuronů P v následující vrstvě po konvoluci se vypočte následovně

$$\begin{aligned} W_O &= \frac{W_I - W_K}{S} + 1 \\ H_O &= \frac{H_I - H_K}{S} + 1 \\ P &= W_O \cdot H_O \cdot K, \end{aligned} \tag{4.1}$$

kde W_I, H_I je šířka a výška vstupního obrázku (mapy), W_K, H_K je šířka a výška konvolučního jádra, S je velikost posunu, K je počet konvolučních jader a W_O, H_O šířka a výška výstupní (nové) příznakové mapy.

Počet trénovatelných parametrů se běžně vypočítá jako součin počtu neuronů a počtu vah (s biasy) dané vrstvy. Kdybychom to spočetli pro první vrstvu, vyšlo by nám $153520 * (5 * 5 + 1) = 3991520$ parametrů. Ale díky tomu, že jsou parametry sdílené, stejně jako v tomto případě, se jejich počet výrazně redukuje.

Počet sdílených parametrů T se pak spočte jako

$$T = M_I \cdot W_K \cdot H_K \cdot M_O + B\,, \tag{4.2}$$

kde W_K, H_K je šířka a výška konvolučního jádra, B je počet biasů, M_I je počet příznakových map předchozí vrstvy a M_O je počet příznakových map aktuální vrstvy. Pro vstupní obrázek datasetu FEI tedy platí vyčíslení v následujících bodech.

- C_1 - 153 520 neuronů, 520 trénovatelných parametrů, 20 příznakových map o velikosti 76×101
- P_1 - 38 760 neuronů, 20 příznakových map o velikosti 38×51
- C_2 - 75 900 neuronů, 25 050 trénovatelných parametrů, 50 příznakových map o velikosti 34×47
- P_2 - 20 400 neuronů, 50 příznakových map o velikosti 17×24
- C_3 - 33 000 neuronů, 45 100 trénovatelných parametrů, 100 příznakových map o velikosti 15×22
- F_1 - 500 neuronů, 16 500 500 trénovatelných parametrů, 500 příznakových map o velikosti 1×1
- F_2 - 200 neuronů, 100 200 trénovatelných parametrů, 200 příznakových map o velikosti 1×1
- F_3 - 50 neuronů, 10 050 trénovatelných parametrů, 50 příznakových map o velikosti 1×1

4.4.1 Metoda učení

Při procesu učení je hlavním cílem minimalizace ztrátové funkce vhodným nastavením vah a k tomu slouží několik optimalizačních metod. Pro návrh modelu byla primárně použita metoda Stochastic Gradient Descent (SGD), přičemž experimentováno bylo i s metodami Adaptive Gradient (AdaGrad) a Nesterov's Accelerated Gradient (NAG). Ještě před začátkem procesu trénování je nutné váhy inicializovat. To bylo provedeno metodou Xavier, která generuje náhodná malá čísla a zároveň zajišťuje zachování stejné variance v celé síti.

SGD

Dle frameworku Caffe metoda SGD aktualizuje váhy W na základě lineární kombinace záporného gradientu $\nabla L(W)$ a předchozích přírůstků vah V_t. Konstanta učení α je váha záporného gradientu a momentum μ váha předchozí hodnoty V_t.

Nové váhy W_{t+1} a jejich přírůstky V_{t+1} získáme ze vztahu

$$\begin{aligned} V_{t+1} &= \mu V_t - \alpha \nabla L(W_t) \\ W_{t+1} &= W_t + V_{t+1}\,, \end{aligned} \tag{4.3}$$

kde V_t značí přírůstek váhy v předešlé iteraci, W_t váhu z předešlé iterace, μ momentum, α konstantu učení , L ztrátovou funkci a index t číslo iterace.

AdaGrad

V této optimalizační metodě se ukládá informace ze všech předchozích iterací $(\nabla L(W))_{t'}$ pro $t' \in \{1, 2, \ldots, t\}$ [32]. Aktualizace vah pak probíhá zvlášť pro každou komponentu i váhového vektoru W dle vztahu

$$(W_{t+1})_i = (W_t)_i - \alpha \frac{(\nabla L(W_t))_i}{\sqrt{\sum_{t'=1}^{t} (\nabla L(W_t'))_i^2}}\,, \tag{4.4}$$

NAG

Metoda NAG zajišťuje v některých případech lepší konvergenci [32]. Aktualizace vah je podobná jako u SGD a počítá se dle vztahu

$$\begin{aligned} V_{t+1} &= \mu V_t - \alpha \nabla L(W_t + \mu V_t) \\ W_{t+1} &= W_t + V_{t+1}\,. \end{aligned} \tag{4.5}$$

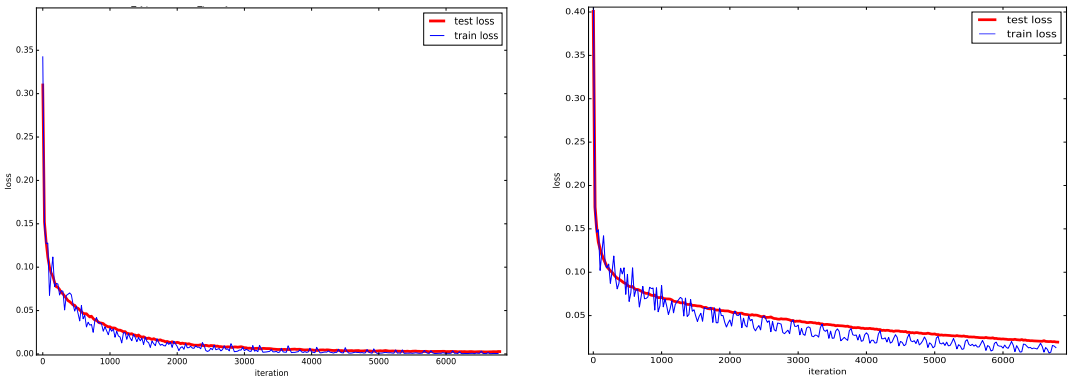
4.4.2 Parametry učení

Následující parametry jsou platné pro metodu SGD, jenž byla použita jako hlavní.

Konstanta učení

Konstantu učení je z odborné praxe doporučeno inicializovat fixovanou hodnotou 0.01, postupně ji snižovat přenásobením o konstantní faktor (např. 10^{-1}) a pozorovat změny ztrátové funkce během trénování. Díky těmto experimentům lze potom volit takové počáteční nastavení, které zajistí optimální průběh trénování. Optimálním průběhem je myšlena konvergence ztrátové funkce v co nejkratším čase.

Dodržením tohoto postupu jsem došel k závěru, že optimální počáteční hodnotou konstanty učení je v tomto případě 0.01. Při testování nižších hodnot ztrátová funkce konvergovala pomaleji a naopak při vyšších hodnotách divergovala do nekonečna.



Obrázek 4.3: Průběhy ztrátové funkce s fixovanou konstantou učení na datasetu FEI, vlevo s konstantou 0.01, vpravo 0.001 .

Ukázkou mohou být průběhy s hodnotami 0.01 a 0.001 na obrázku 4.3. Průběh divergence bohužel nemohl být vykreslen, neboť framework Caffe v této situaci neukládá číselné hodnoty.

Po získání optimální počáteční hodnoty je výhodné konstantu učení během procesu trénování postupně měnit (snižovat/zvyšovat). Zpočátku je dobré volit konstantu větší, postupně ji snižovat a v ideálním případě se dostat do globálního minima.V reálném případě dokonvergujeme spíše do některého z lokálních minim. Dynamickou změnou konstanty docílíme větší robustnosti v konvergenci a snadněji "doputujeme"do minima ztrátové funkce. Pokud bychom nechali konstantu fixovanou, mohlo by dojít k oscilacím hodnoty ztráty a vůbec bychom se nemuseli do tohoto minima dostat.

K dynamické změně konstanty učení slouží specifické funkce, které jsou již implementovány ve frameworku Caffe. Nejznámější je skoková funkce (v Caffe frameworku jako "step"), která mění hodnotu konstanty skokově po předem zvoleném počtu iterací. Existují ale i funkce, které mění konstantu učení po každé iteraci, přičemž jednu takovou jsem použil i ve svém návrhu. Její název by se dal volně přeložit jako funkce inverzního rozpadu (v Caffe frameworku jako "inv") a hodnotu konstanty α počítá následovně

$$\alpha = \alpha_0 \cdot (1 + \gamma \cdot i)^{-p}, \tag{4.6}$$

kde α_0 je počáteční hodnota konstanty učení, i je číslo iterace, γ a p (v Caffe "gamma"a "power") jsou další volené parametry, ovlivňující velikost změny konstanty. Dle doporučení Caffe frameworku jsem jim ponechal výchozí hodnoty, tedy γ jako 0.0001 a p jako 0.75.

- α_0 : 0.01
- γ : 0.0001
- p : 0.75

Momentum

Momentum je parametr, který ve vztahu pro výpočet nového přírůstku váhy přidává důležitost váhovému přírůstku z předchozí iterace. Jinými slovy momentum vyjadřuje, jaký vliv bude mít váha z předchozí iterace na výpočet nové váhy aktuální iterace. Urychluje trénování a zabraňuje uvíznutí v lokálním extrému.

Pokud se váhy mění ve stejném směru (stejný směr gradientu), je možné urychlit trénování tím, že v daném okamžiku zvýšíme váhové přírůstky. K čemuž slouží právě momentum, jenž se doporučuje volit v okolí hodnoty 0.9. Tato hodnota se ukázala jako nejlepší během několika let praxe.

- μ : 0.9

Batch size

Pojem batch můžeme chápat jako určitou podmnožinu trénovací (validační, testovací) množiny a batch size jako její velikost.Tímto parametrem je vyjádřeno kolik příkladů z trénovací množiny bude použito pro jednu iteraci učícího algoritmu. V praxi to znamená, že vezmeme najednou určitý počet trénovacích příkladů, s nimi uskutečníme dopředné šíření, zpětné šíření a poté co sítí projde poslední příklad z batche provedeme aktualizaci vah. Kvůli tomu se také metoda učení označuje jako stochastická (Stochastic gradient descent) a do jisté míry odhaduje skutečné hodnoty vah, které by se jinak získali aktualizací po každém příkladu z trénovací množiny. Výhodou batchů je, že spotřebují méně paměti než situace, kdybychom použili celou trénovací množinu, což by v mnohých případech kvůli její velikosti ani nešlo. Síť se navíc trénuje rychleji díky častější aktualizaci vah. Nevýhodou je méně přesný odhad gradientu. Čím je velikost batche menší, tím je odhad méně kvalitní. Obdobné je to pro validační nebo testovací množinu.

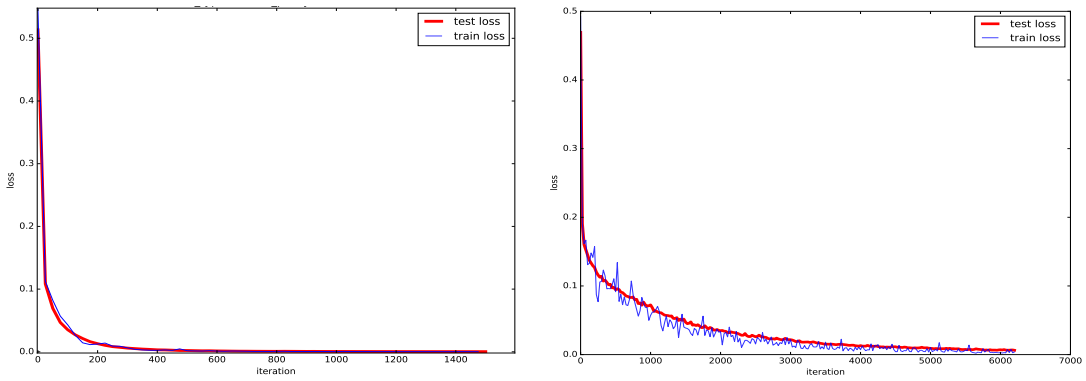
- Batch size (trénovací): 64
- Batch size (validační): 100

Počet iterací

Počet iterací je zastavovací podmínka trénování a ideálně se volí taková hodnota, při které můžeme říci, že ztrátová funkce je ustálená. Do takového stavu se většinou dostaneme učením sítě opakovaným předkládáním trénovací množiny. Jeden průchod celé trénovací množiny sítí se označuje jako epocha a tímto parametrem můžeme určit, kolik epoch se při trénování provede. Pro získání ideální hodnoty je nutné provést několik takovýchto experimentů a vyhodnotit průběh ztrátové funkce. Za ustálenou ztrátové funkce jsem považoval stav, kdy se hodnota ztráty začala pohybovat v intervalu $[0, 0.02]$. Pro dataset $AT\&T$ se hodnota

ztrátové funkce ustaluje přibližně po 1500 iteracích, pro dataset FEI přibližně po 6000 iteracích.

- Počet iterací ($AT\&T$): 1500 ($\sim$ 16 epoch)
- Počet iterací (FEI): 6000 ($\sim$ 13 epoch)



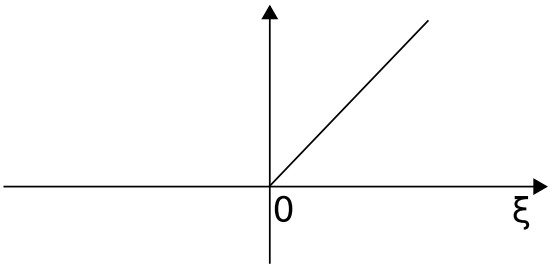
Obrázek 4.4: Průběh ztrátové funkce s datasetem AT&T po 1500 iteracích (vlevo) a s datasetem FEI po 6000 iteracích (vpravo), oba průběhy s dynamickou konstantou učení.

4.4.3 Aktivační funkce

Jako aktivační funkci jsem zvolil ReLU (Rectified Linear Unit), která se počítá dle vztahu

$$f(\xi) = max(0, \xi). \tag{4.7}$$

V posledních letech se stala velmi populární a to díky svým vlastnostem. Bylo zjištěno, že při použití ReLU trénování konverguje mnohem rychleji v porovnání s běžnými aktivačními funkcemi jako je sigmoida nebo hyperbolický tangens. Běžné funkce totiž omezí výstupní prostor na velmi malý rozsah (např. 0 až 1) , to ovlivňuje velikost gradientu, jehož hodnoty mohou být v tu chvíli velice nízké, a tím se zpomalí celý proces trénování. Funkce ReLU je také mnohem jednodušší a pouze prahuje vstupní hodnoty. Pokud je hodnota větší než 0 dojde k aktivaci, v ostatních případech ne.



Obrázek 4.5: Ilustrace aktivační funkce ReLU.

4.4.4 Kontrastní ztrátová funkce

Kontrastní ztrátová funkce byla obecně popsána v 3.7.3 a základním principem učení je ji minimalizovat. Ve frameworku Caffe je její výpočet mírně modifikován. Rozdíl je v označení (labelu) pozitivních a negativních párů v datasetu. V Caffe jsou pozitivní páry označeny labelem 1 a negativní páry labelem 0, zatímco v teorii 3.7.3 je to naopak. Pokud bychom označení prohodili, výpočet ztráty by byl chybný a síť bychom nenatrénovali správně. Framework Caffe definuje kontrastní ztrátu následujícím vztahem

$$E = \frac{1}{2N}\sum_{n=1}^{N}(y)d^2 + (1 - y)\,max(\text{margin} - d, 0)^2\,, \tag{4.8}$$

kde $d = \|a_n - b_n\|_2$, což je Euklidovská vzdálenost mezi příznakovými vektory a_n, b_n jednoho páru z trénovací množiny. N je počet trénovacích příkladů, y je label a $margin$ zvolená hraniční (prahová) hodnota oddělující pozitivní a negativní páry. Prahová hodnota nijak neovlivňuje kvalitu systému, pouze rozšiřuje nebo smršťuje příznakový prostor, do kterého jsou příklady mapovány. V tomto případě byla nastavena na hodnotu $margin = 1$.

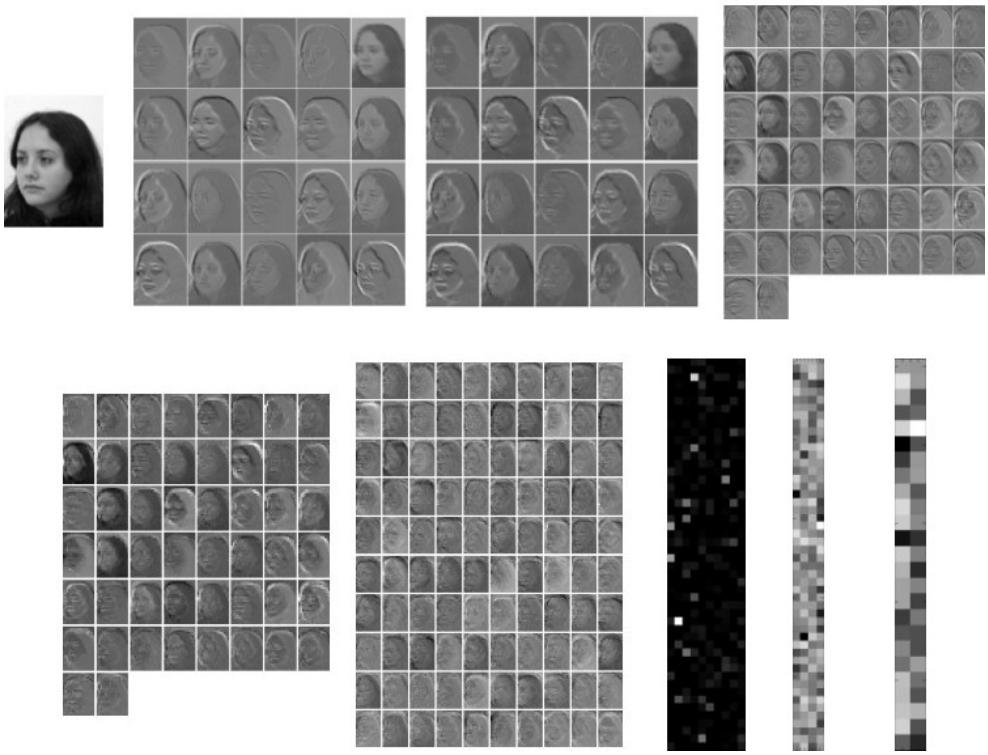
4.5 Testování a vyhodnocení

Pro zjištění kvality identifikačního systému byl vyhodnocován počet případů, ve kterých došlo ke špatnému rozhodnutí. Špatným rozhodnutím se rozumí dva případy. První odpovídá situaci, kdy je na obou obrázcích stejná osoba, ale systém rozhodne, že se jedná o různé osoby (špatně zamítnuté). Druhý případ špatného rozhodnutí nastane, pokud jsou osoby na obrázcích různé, ale systém rozhodne, že se jedná o stejnou osobu (špatně přijaté). Obecně je žádoucí, aby byl počet obou těchto případů minimální. Některé úlohy ale mohou být postaveny tak, že upřednostňují minimální počet pouze špatně přijatých nebo špatně zamítnutých.

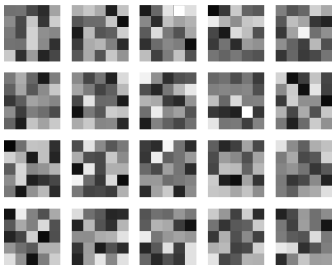
Poměr špatně přijatých a špatně zamítnutých případů lze ovlivnit změnou prahu, který je pomyslnou hranicí oddělující shodné a různé páry. Pokud je Euklidovská vzdálenost testovaného páru menší než práh, řekneme že jde o stejnou osobu, pokud je naopak větší, řekneme že jde o různé osoby. Při testování tedy bylo vybráno několik hodnot prahu, pro které byl následně vyhodnocen počet špatně přijatých a špatně akceptovaných. A jelikož byl systém trénován s prahem 1, ostatní testované prahy byly voleny z intervalu v okolí tohoto bodu.

Aby byly výsledky relevantní je nutné testování několikrát zopakovat metodou cross-validation. Zjednodušeně řečeno je potřeba několikrát promíchat příklady v jednotlivých podmnožinách datasetu (trénovací, validační, testovací) a provést celý proces znovu. Výsledek se pak bere jako průměr z těchto dílčích vyhodnocení. Při testování jsem takto provedl deset opakování.

Pro názornost toho, jak naučená síť zpracovává vstupní data je na obrázku 4.6 ukázán její vnitřní stav pro jeden testovací obrázek. Znázorňuje výstupy jednotlivých vrstev v rámci jedné větve siamské sítě. Na obrázku 4.7 jsou potom zobrazeny naučené filtry z první konvoluční vrstvy.



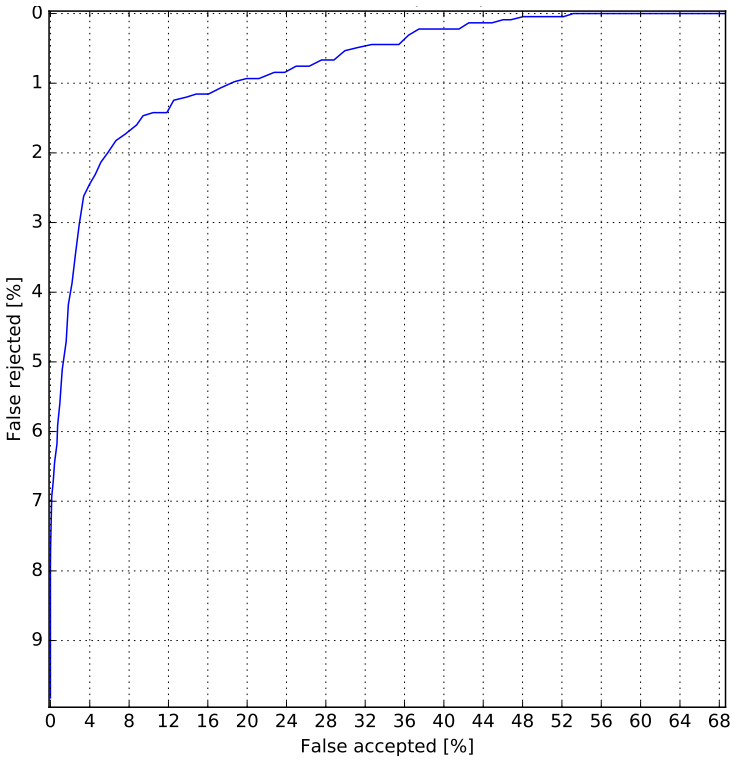
Obrázek 4.6: Ukázka vnitřního stavu naučené sítě pro jeden testovací obrázek datasetu FEI v rámci jedné větve siamské sítě.



Obrázek 4.7: Naučené filtry z první konvoluční vrstvy, 20 filtrů o velikosti 5×5.

4.5.1 Dataset AT&T

Tento dataset posloužil jako reference pro ověření kvality sítě v porovnáním s [19]. Testování bylo provedeno nad testovací podmnožinou s 5000 příklady. Z toho je 500 pozitivních a 4500 negativních párů, které odpovídají 5 osobám, jenž síť ještě neviděla (nebyly použity při trénování). Výstupem je křivka parametrizovaná prahovou hodnotou na obrázku 4.8, vyjadřující procento špatně přijatých vs. špatně zamítnutých případů.

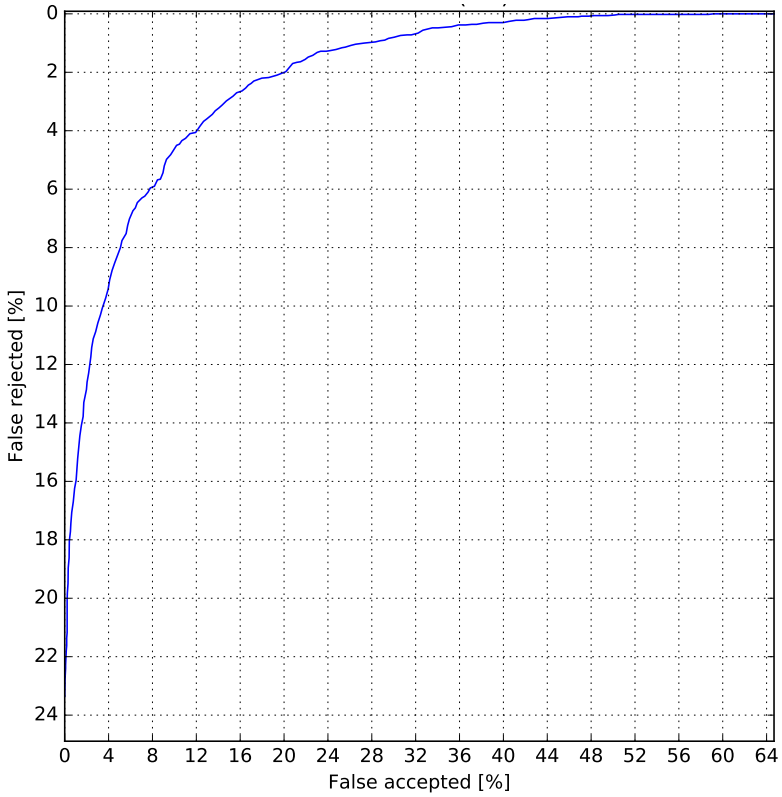


Obrázek 4.8: Procento špatně přijatých vs. špatně zamítnutých případů při testování databáze AT&T.

Nejlepších výsledků, tedy nejnižšího součtu špatně akceptovaných a špatně zamítnutých, bylo docíleno při hodnotách prahu v rozmezí $[0.30, 0.35]$, kdy přesnost identifikace dosáhla v průměru hodnoty 94.05%. Špatně přijatých případů je potom 2.93% a špatně zamítnutých 3.02%. To odpovídá výsledkům v [19], čímž jsem ověřil kvalitu vlastního návrhu.

4.5.2 Dataset FEI

Testování na datasetu FEI bylo uskutečněno s podmnožinou obsahující 5000 příkladů. Oproti datasetu AT&T je tato podmnožina vždy složena z 1365 pozitivních a 3635 negativních párů, které odpovídají 15 neviděným osobám (nebyly použity při trénování). Výsledný poměr počtu špatně přijatých vs. špatně zamítnutých případů s různou volbou prahu je vyjádřen křivkou 4.9.



Obrázek 4.9: Procento špatně přijatých vs. špatně zamítnutých případů při testování databáze FEI.

Nejlepších výsledků bylo dosahováno při hodnotách prahu v rozmezí [0.55, 0.75]. Výsledná průměrná přesnost se v tomto případě dostala na hodnotu 87.08% s 5.9% špatně přijatých a 7.02% špatně zamítnutých případů.

4.5.3 Diskuze a porovnání

Z výsledků je zřejmé, že vlastní návrh zdaleka nedosahuje takové přesnosti jako "state of the art"metody FaceNet nebo Baidu, což může být způsobeno kombinací různých vlivů. Jedním z nich je rozhodně velikost trénovací množiny,

zatímco nejlepší zmíněné metody používají při trénování až desítky tisíc identit (FaceNet dokonce 8 milionů) ve vlastním návrhu jich je u datasetu FEI pouze 185. Některé metody oproti vlastnímu návrhu navíc používají speciální algoritmy pro zarovnání obličeje do přímého směru, což rozpoznávání výrazně zjednodušuje. Vlastní navržený systém obličeje nijak nezarovnává, což se na výsledném ropzpoznávání do určité míry podepisuje. Při vyhodnocování bylo potvrzeno, že nejhoršími případy verifikace jsou ty, kdy vedle sebe porovnáváme obličej v přímém směru a jeho profil nebo oba profilové pohledy (levý, pravý). Vliv na kvalitu rozpoznávání mají jistě i zvolené parametry a architektura sítě, která je u nejlepších metod mnohem hlubší a propracovanější.

	Testovací dataset	Přesnost
Baidu	LFW, 6000 párů	99.77%
FaceNet	LFW, 6000 párů	99.63%
DeepID3	LFW, 6000 párů	99.53%
DeepFace	LFW, 6000 párů	97.35%
Vlastní návrh (AT&T)	AT&T, 5000 párů	94.05%
Vlastní návrh (FEI)	FEI, 5000 párů	87.08%

Tabulka 4.3: Porovnání přesnosti verifikace vlastního návrhu a stávajících metod.

5 Závěr

Cílem této práce bylo nastudovat možnosti strojového rozpoznávání obličeje a na základě toho pak navrhnout vlastní systém pro rozpoznávání. Součástí bylo prozkoumat existující databáze obličejů a vybrat z nich tu, která se hodí pro vlastní návrh. Samozřejmostí pak bylo porovnat vlastní návrh se stávajícími metodami.

V úvodu byla popsána teorie rozpoznávání tváře s ohlédnutím do historie a shrnutím přístupů během let. Ze stávajících přístupů byly podrobněji prozkoumány některé tradiční metody, jenž během vývoje hrály zásadní roli, konkrétně jde o Eigenfaces, Fisherfaces nebo LBP Patterns. V současnosti se v oblasti rozpoznávání obličeje dostaly na vrchol metody založené na hlubokém učení, které svou kvalitou překonávají všechny předešlé. Mezi nedávno publikované, podrobněji popsané v této práci, patří například DeepFace, DeepID3, FaceNet nebo Baidu. Z nichž některé dosahují přesnosti rozpoznávání více než 99%.

Dále byly prozkoumány existující databáze obličejů. Dříve byly takovéto databáze vytvářeny většinou účelovým pořizováním snímků obličejů v laboratořích a v různých ohledech byly omezené, ať už v rozsahu (počtu identit) nebo tím, že byly pořízeny v homogenním prostředí a neodpovídají reálným situacím (různé prostředí, světelné podmínky, apod.). V dnešní době se tyto databáze vytvářejí spíše shromažďováním snímků z internetu, které pocházejí z reálného prostředí a umožňují vytvářet rozsáhlé soubory až s miliony dat. Díky tomu mohou vznikat přesnější a robustnější systémy rozpoznávání.

Jelikož jsou současné nejlepší metody, včetně vlastního návrhu, založené na hlubokém učení, je v další části popsána teorie neuronových sítí, od obecných vlastností, přes algoritmus učení až po konvoluční neuronové sítě s jejich významnými architekturami, proslavenými v oblasti rozpoznávání digitálního obrazu, kterými jsou například AlexNet, GoogleNet nebo ResNet.

Pro návrh vlastního identifikačního systému byla zvolena metoda siamských neuronových sítí, jejíž teorie byla vysvětlena nejprve obecně a poté ve spojení

s rozpoznáváním obličeje. Samotný návrh pak spočíval v několika krocích. Na začátku byl diskutován výběr databáze obličejů. S ohledem na různé pózy obličeje, různé výrazy ve tváři, různé světelné pomínky, ale i náročnost procesu učení byly vybrány dvě databáze menšího rozsahu, primárně FEI a paralelně k tomu AT&T. Následujícím krokem bylo předzpracování dat a tvorba datasetu s rozdělením na trénovací, validační a testovací podmnožinu. Každou podmnožinu datasetu v této metodě tvoří seznam dvojic obličejů různých a shodných identit. Důležité v tomto případě bylo zachovat poměr shodných a různých identit a to hlavně v trénovací podmnožině. Pokud tento poměr nebyl dodržen, síť začala upřednostňovat buď pouze shodné nebo pouze různé páry a to vedlo ke špatným výsledkům při rozhodování. Následujícím krokem byl návrh architektury sítě. Jako výchozí byla zvolena architektura LeNet s níž bylo provedeno několik pokusů, ale přesnost rozpoznávání nedosahovala významných kvalit. S architekturou bylo tedy dále experimentováno, změna spočívala především v počtu konvolučních, poolingových vrstev, velikosti a počtu konvolučních jader, velikosti posunu, apod. S konečnou úpravou architektury, která je uvedena v této práci, pak došlo při testování k výraznému zlepšení výsledného rozpoznávání. V dalších krocích byl optimalizován průběh ztrátové funkce během trénování a to zejména vhodnou volbou parametrů primárně zvolené metody učení SGD. Tyto parametry mají na průběh trénování velký vliv a s jejich experimentováním bylo dosaženo optimální konvergence ztrátové funkce a tím i celého trénovacího procesu. Při testování natrénovaného modelu bylo důležité vhodně zvolit hodnotu prahu, která rozhoduje o shodě identit a ovlivňuje výsledný počet špatných rozhodnutí. Bylo experimentováno s několika hodnotami prahu a za nejlepší výsledek byl považován vždy ten, při kterém byl součet špatně rozhodnutých případů nejnižší. Několikanásobným testováním bylo takto dosaženo průměrné přesnosti 87.08% s datasetem FEI, resp. 94.05% s datasetem AT&T.

Tyto výsledky nicméně nedosahují kvality současných nejlepších metod, tudíž by bylo vhodné se systémem dále pracovat. Vhodným a zřejmě i nejjednodušším vylepšením by bylo razantní navýšení počtu trénovacích dat, což s sebou nese nevýhodu v podobě větších výpočetních nároků. Kvalitu výsledků by mohlo zvýšit například i použití metody pro zarovnání obličeje do přímého směru. S ohledem na nejlepší metody se rovněž nabízí zlepšení v podobě úpravy architektury sítě, použití některé ze zmíněných (GoogLeNet, ResNet, apod.) by výsledné rozhodování určitě zpřesnilo.

Literatura

[1] T. Kanade, *Picture processing system by computer complex and recognition of human faces*. PhD thesis, Kyoto University, November 1973.

[2] L. Wiskott, J. Fellous, N. Krüger, and C. Malsburg, "Face recognition by elastic bunch graph matching," in *IEEE Transactions on pattern Analysis and Machine Intelligence*, IEEE, 1997.

[3] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[4] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 19, no. 7, pp. 711–720, 1997.

[5] M. S. Bartlett, J. R. Movellan, and T. J. Sejnowski, "Face recognition by independent component analysis," *IEEE Transactions on neural networks*, vol. 13, no. 6, pp. 1450–1464, 2002.

[6] M.-H. Yang, "Kernel eigenfaces vs. kernel fisherfaces: Face recognition using kernel methods.," in *Fgr*, vol. 2, p. 215, 2002.

[7] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face recognition with local binary patterns," *Computer vision-eccv 2004*, pp. 469–481, 2004.

[8] OpenCV Dev Team, *Face Recognition with OpenCV [Online]*, April 2017. Available at: `http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html`.

[9] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[10] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1701–1708, 2014.

[11] Y. Sun, D. Liang, X. Wang, and X. Tang, "Deepid3: Face recognition with very deep neural networks," *arXiv preprint arXiv:1502.00873*, 2015.

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015.

[14] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[15] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 815–823, 2015.

[16] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*, pp. 818–833, Springer, 2014.

[17] J. Liu, Y. Deng, T. Bai, Z. Wei, and C. Huang, "Targeting ultimate accuracy: Face recognition via deep embedding," *arXiv preprint arXiv:1506.07310*, 2015.

[18] A. M. Martinez, "The ar face database," *CVC Technical Report24*, 1998.

[19] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. of Computer Vision and Pattern Recognition Conference*, IEEE Press, 2005.

[20] AT&T Laboratories Cambridge, *The Database of Faces [Online]*, 2002. Available at: `http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html`.

[21] L. Spacek, *Face Recognition Data [Online]*, 1997. Available at: `http://cmp.felk.cvut.cz/~spacelib/faces/`.

[22] H.-W. Ng and S. Winkler, "A data-driven approach to cleaning large face datasets," in *Image Processing (ICIP), 2014 IEEE International Conference on*, pp. 343–347, IEEE, 2014.

[23] C. E. Thomaz, *FEI Face Database [Online]*, 2006, Sao Paulo, Brazil. Available at: `http://fei.edu.br/~cet/facedatabase.html`.

[24] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, "The feret evaluation methodology for face-recognition algorithms," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 10, pp. 1090–1104, 2000.

[25] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Tech. Rep. 07-49, University of Massachusetts, Amherst, October 2007.

[26] M. Grgic, K. Delac, and S. Grgic, "Scface–surveillance cameras face database," *Multimedia tools and applications*, vol. 51, no. 3, pp. 863–879, 2011.

[27] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 529–534, IEEE, 2011.

[28] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[29] D. O. Hebb, *The organization of behavior: A neuropsychological approach*. John Wiley & Sons, 1949.

[30] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[32] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[33] E. Volná, "Neuronové sítě 1. 2. vyd," *Ostrava: Ostravská univerzita v Ostravě*, 2008.

[34] Prisma Labs, Inc., *Prisma [Online]*, 2016. Available at: `https://prisma-ai.com/`.

[35] Stanford University CS231n, *Convolutional Neural Networks for Visual Recognition [Online]*, 2017. Available at: `http://cs231n.github.io/convolutional-networks/`.

[36] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[37] Wikipedia, otevřená encyklopedie, *Konvoluce [Online]*, February 2017. Available at: `https://cs.wikipedia.org/wiki/Konvoluce`.

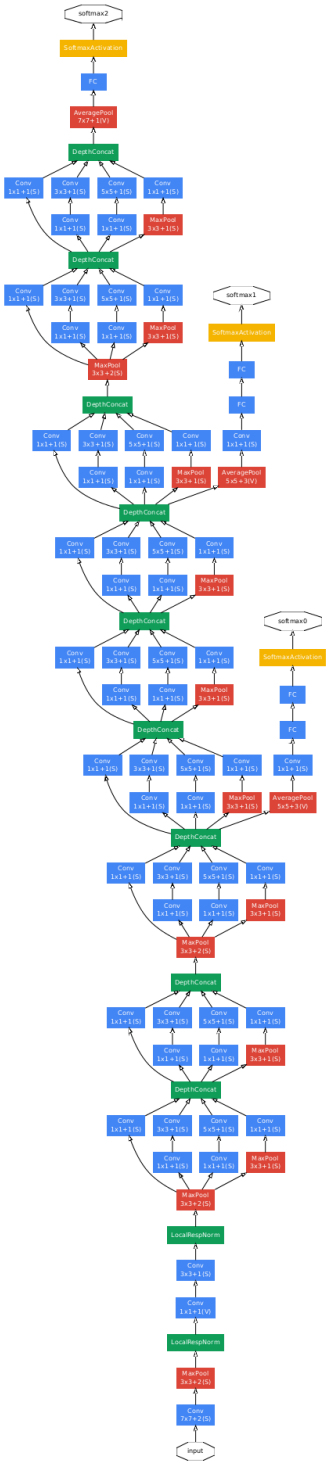
[38] H. El Khiyari, H. Wechsler, *et al.*, "Face recognition across time lapse using convolutional neural networks," *Journal of Information Security*, vol. 7, no. 03, p. 141, 2016.

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, 2016.

[40] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, and R. Shah, "Signature verification using a "siamese"time delay neural network," *IJPRAI*, vol. 7, no. 4, pp. 669–688, 1993.

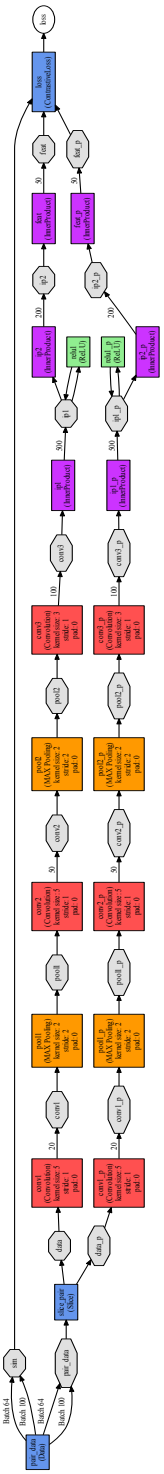
[41] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

Příloha A



Obrázek 1: Architektura GoogLeNet, převzato z [13]

Příloha B



Obrázek 2: Architektura navržené siamské sítě v Caffe frameworku.