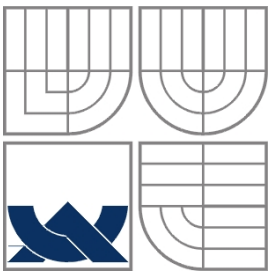
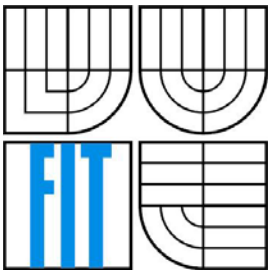


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

NEURONOVÉ SÍTĚ PRO LOKALIZACI LIDSKÉHO OBLIČEJE

APPLICATION OF NEURAL NETWORKS FOR HUMAN FACE LOCALIZATION

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

JAKUB ŽÁK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MIROSLAV ŠVUB

BRNO 2007

Licenční smlouva

Zadání

1. Prostudujte dostupné materiály na téma neuronových sítí pro zpracování obrazu, rozpoznávání, detekci a případně klasifikaci vzorů.
2. Zaměřte se na problematiku detekce a lokalizace lidského obličeje v obraze. Analyzujte možnosti současných metod a neuronových sítí specificky orientovaných na tento problém.
3. Na základě zvolených metod navrhněte detektor lidského obličeje v obraze.
4. Experimentujte s vaší implementací a případně navrhněte vlastní modifikace použitých metod.
5. Diskutujte dosažené výsledky a vlastnosti vybraných neuronových sítí. Navrhněte možnosti budoucího rozšíření vaší práce.
6. Vytvořte stručný plakát prezentující vaši bakalářskou práci, její cíle a výsledky.

Licenční smlouva je uložena v archivu Fakulty informačních technologií.

Abstrakt

Tato práce se zabývá využitím vícevrstvých neuronových sítí na problém lokalizace lidského obličeje ve statickém obraze. Tato metoda má obecně dobré generalizační schopnosti a proto není nutné sestavovat složité modely analyzovaných dat. Je také rozebrána možnost využití neuronové sítě s pozměněnou architekturou.

Klíčová slova

Lokalizace obličeje, umělá neuronová síť, perceptron, MLP, soft-computing, počítačové vidění, obličejová databáze

Abstract

This paper describes aplication of multi layered neural network for solving problem of detection human face in static picture. This Method has good generalizational capabilities in general and there is no need to assembly complex models of analyzed data. There is also mentioned posibility of using neural network with changed architecture in this work.

Keywords

Human face localization, artificial neural network, perceptron, MLP, soft-computing, computer vision, face database

Citace

Jakub Žák: Neuronové sítě pro lokalizaci lidského obličeje, bakalářská práce, Brno, FIT VUT v Brně, 2007

Neuronové sítě pro lokalizaci lidského obličeje

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Miroslava Švuba. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jakub Žák
8.května 2007

Poděkování

Rád bych poděkoval především svému vedoucímu Ing. Švubovi, který mi poskytl cenné rady a odborné vevedení při vypracování tohoto této práce. Další dík patří mé lásce, rodině, přátelům a všem, kteří mě podporovali při mé cestě.

Obsah

Úvod

Počítačová technika je v současné době velmi dynamicky se rozvíjející odvětví. Lidé se pomocí této techniky snaží automatizovat čím dál více úkonů. Nutno říci, že se jim to daří někde ve větší, někde v menší míře. Na počátku této éry byly počítače z dnešního pohledu primitivní a veškerá snaha se soustřeďovala na zvládnutí jednoduchých matematických úkonů. S příchodem poznání se však počítače stávaly složitějšími, ale zároveň zvládaly čím dál náročnější úkony. Jak rostla výpočetní síla, byly vymýšleny náročnější algoritmy a nové přístupy k řešení specifických problémů. Jedním z takových problémů může být i lokalizace lidského obličeje v obraze. Krom jiných přístupů byl ve 40. letech vyvinut koncept *Neuronových sítí*, když pánové McCulloch a Pitts sestavili první umělý *neuron*. Zájem o tuto oblast se velice zvýšil se sestavením modelu *perceptronové sítě* (Rosenblatt, 1962), která byla schopná rozpoznat jednoduché vzory (pro fyzickou realizaci perceptronu použil dvojici elektrického motoru s potenciometrem).

Zájem nicméně opadl po sestavení matematického důkazu, potvrzujícího vysokou chybovost takových sítí (Minsky a Papert, 1969). Renesance zájmu o tento přístup řešení problému rozpoznání vzorů přišla až v době, kdy byly vyvinuty lepší algoritmy pro trénování neuronové sítě. Jedním z hlavních je i algoritmus *back propagation* využitý i v této práci. Pro bližší informace viz[1].

Neuronová síť je síť složená z neuronů, které jsou vzájemně různým způsobem propojeny. Neuron jako základní jednotka neuronové sítě je definován množstvím spojů, které do něj vstupují, které z něj vystupují a svou přenosovou funkcí. Každý vstupní spoj (synapse) má zároveň určitou váhu, která definuje, jak moc bude daný spoj ovlivňovat celkový výstup neuronu. Množství spojů a neuronů i celková topologie sítě pak určuje, jak se bude síť chovat. Pro různé účely se používají různé typy sítí právě podle své topologie a vhodnosti pro daný účel.

Tato práce se zabývá použitím neuronové sítě typu vícevrstvý perceptron pro rozpoznání lidského obličeje ze statického obrázku. Obličej je zároveň ve svislé poloze a otočen směrem k záznamovému zařízení.

Po tomto krátkém uvedení do problematiky, o které pojednává tato práce, se ještě pouze náznakem zmíním o obsahu dalších kapitol. V úvodní kapitole bude pojednáno obecně o umělých neuronových sítích se zaměřením na síť typu vícevrstvý perceptron, která bude využita i v tomto projektu. Druhá kapitola obsahuje motivaci k řešení problému typu „nalezení obličeje v obrázku“. Zároveň jsou ve druhé kapitole zmíněny některé problémy spojené s detekcí obličeje a v závěru jsou náznakem popsány některé další přístupy k řešení tohoto problému. Třetí kapitola obsahuje návrh systému a popisuje některé problémy, které jsou spojeny s dílčími součástmi systému. Ve čtvrté kapitole je rozebrána implementace celého systému a pátá kapitola shrnuje dosažené výsledky.

1 Neuronové sítě

Tato kapitola obsahuje obecné informace o neuronových sítích. Je v ní nastíněn princip fungování neuronových sítí a v podkapitolách jsou pak popsány různé části a vlastnosti umělých neuronových sítí. Začneme hrstkou obecných informací v několika následujících odstavcích a poté přijde na řadu popis samotného neuronu tak, jak jej definovali McCulloch a jeho student Pitts. Po popisu neuronu se můžeme pustit do definice neuronové sítě. V závěru tohoto obecného úvodu bude i detailněji pojednáno o různých principech učení neuronových sítí. Součástí jednotlivých popisů bude také definice základních pojmů, se kterými bude dále pracováno.

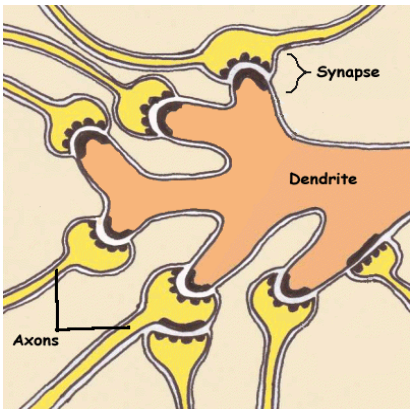
Původní koncept umělých neuronových sítí byl vyvinut na základě zkoumání funkce lidského oka a zkoumání stavby a chování biologické neuronové sítě, která nám zprostředkovává obrazové vjemy z okolí. Umělá neuronová síť (dále jen neuronová síť) je propojená síť základních procesních elementů nazvaných umělé neurony (dále jen neurony) a zároveň je modelem sítě biologické. Je znázorňována ve formě orientovaného grafu, kde uzly jsou jednotlivé neurony. Chování této sítě je závislé na propojení jednotlivých neuronů, stejně tak jako na topologii celé sítě, která zahrnuje počty vstupních a výstupních neuronů, či počet vrstev. Propojení mohou být různá, od jednoduchých dopředných sítí, kde šipky orientovaného grafu vedou pouze směrem od vstupních uzlů k uzlům výstupním, až po rekurentní sítě se zpětnými vazbami.

Síla neuronových sítí, a tedy důvod jejich použití v poměrně rozsáhlé aplikační oblasti, spočívá v jejich schopnosti zobecňování, což znamená, že je můžeme využít v situacích, kdy dopředu neznáme množinu výsledků, kterých může řešení daného problému nabývat. Tento fakt nepřímo souvisí s nutností učení neuronových sítí, které bude popsáno v některé následující podkapitole. Toto je jedno opodstatněné využití neuronových sítí. Jako další vhodné aplikační oblasti se jeví problémy, při jejichž řešení neznáme přesný matematický popis problému, nebo je popis tak složitý, že znemožňuje praktické použití a zároveň zjednodušování modelu nevede k požadovaným výsledkům. Dalším vhodným využitím je situace, kdy máme jistou množinu výsledků a množinu vstupů, ze kterých tyto výsledky vychází, ale neznáme matematický předpis, který převede jednu množinu v druhou.

1.1 Model umělého neuronu

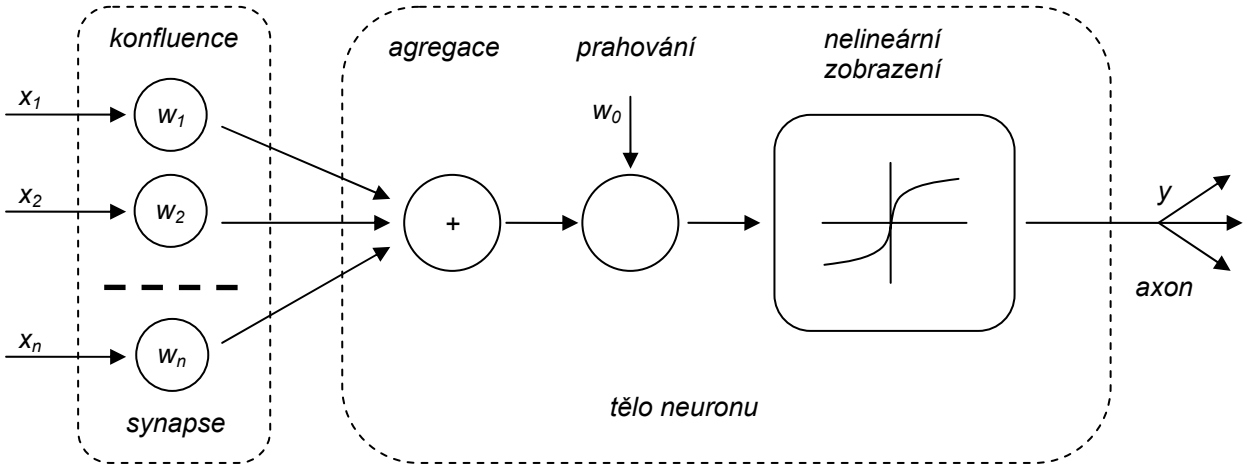
Neuron je základní procesní jednotka neuronové sítě. Jeho model je odvozen od modelu biologického neuronu. Biologický neuron je zařízení, které má mnoho vstupů a jeden jediný výstup. V biologické terminologii jsou vstupy nazývány *dendrity* a výstup je nazýván *axon*. Propojení takové sítě je realizováno napojením výstupního axonu na vstupní dendrity. Protože každý neuron má jediný výstup, který je napojen na mnoho vstupů, napojení je v biologické síti, kterou vymyslela matka

příroda, realizováno napojením mnoha axonů na jeden dendrit v jeho různých částech(viz obr. 1.1). Tato spojení se nazývají *synapse*. V počítačové praxi je synapse realizována rozmnožením výstupu a jeho přivedením na následující vstupy.



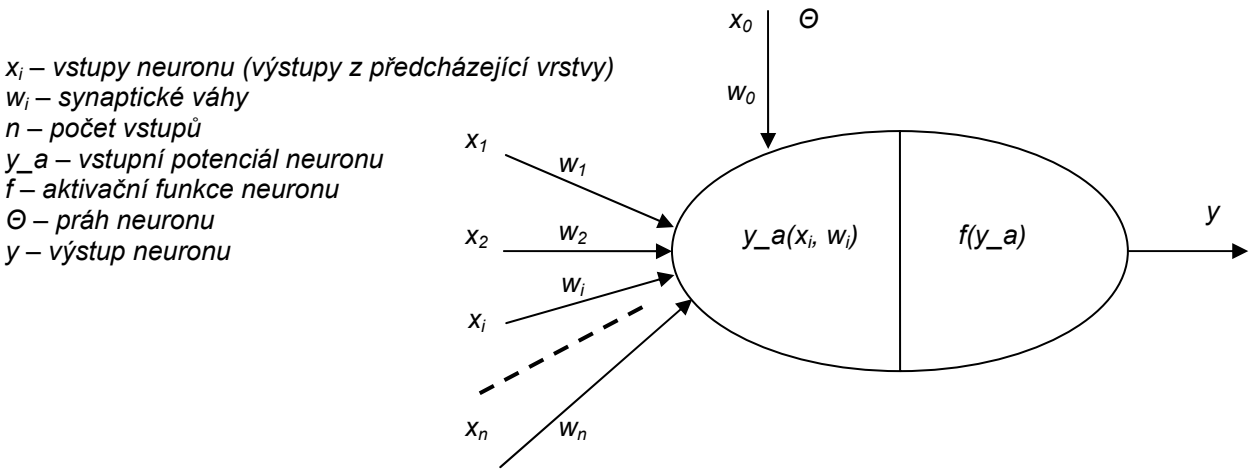
obr. 1.1 Biologický neuron

Neuron se skládá z několika částí, které je potřeba modelovat. Kromě zmíněných vstupů a výstupu musí být v neuronu realizovány váhy jednotlivých vstupů, které vyjadřují relativní důležitost daného vstupu. Druhy neuronů se mohou dělit také podle jeho *přenosové funkce*, která je pro neuron typická. Biologické funkce „pravých" neuronů jsou nahrazeny v umělých neuronech funkcemi matematickými. Dosud v praxi nejčastěji používaným modelem je tzv. *formální model neuronu*. Tento typ bývá také někdy nazýván podle svých stvořitelů McCulloch-Pittsův neuron.



obr. 1.2 Analogie biologického a umělého neuronu

Vstupy x_1, $x_2 \ldots x_n$ modelují dendrity, váhy w_1, $w_2 \ldots w_n$ modelují synapse a výstup y simuluje činnost axonu. Agregace vstupních signálů, jejich porovnání s prahovou hodnotou Θ a následně jejich nelineární zobrazení představují model těla neuronu, tj. vyhodnocení celkového vstupního aktivačního potenciálu a jeho transformaci na výstupní signál.



obr. 1.3 McCulloch-Pittsův model neuronu

Každý umělý neuron obsahuje konečný počet vstupů x_n a jediný výstup y, který je samozřejmě možno rozmnožit do libovolného konečného počtu kopií, jak bylo řečeno výše. V každém neuronu (popisovaný základní model nevyjímaje) se vstupní hodnoty transformují na výstup pomocí minimálně dvou výpočetních procedur. Konkrétně se jedná o výpočet vstupního potenciálu y_a a o tzv. *aktivační funkci f*.

I když neuronové sítě umí rozpoznávat i nečíselné veličiny (jako například v našem případě obrazy), vstupy do neuronů jsou v převážné většině případů realizovány pomocí reálných čísel. V závislosti na poloze neuronu v síti můžeme vstupy rozdělit do dvou kategorií. První skupinu tvoří vstupy předchozích (*presynaptických*) neuronů, v další skupině jsou pak neurony jejichž vstupy tvoří podněty z okolí.

Vektor vstupů $X=[x_1,x_2,...x_n]$ představuje soubor hodnot, které mohou nabývat buď kvalitativních nebo kvantitativních hodnot. Pod kvalitativními hodnotami si můžeme představit booleovské *true* resp. *false* a pod vyjádřením vstupu jako jedné z těchto hodnot si pak můžeme představit, že modelovaný objekt danou vlastnost má respektive nemá. Vyjádření kvantitativní znamená vyjádření míry dané vlastnosti, kterou má objekt, či vyjádření hodnoty měřené veličiny. Kvantitativně jsou vlastnosti či veličiny vyjádřeny reálným číslem.

Další důležitou vlastností neuronu jsou váhy jednotlivých vstupů. Váhy reprezentují citlivost s jakou daný vstup ovlivňuje celkový výstup neuronu a většinou jsou reprezentovány reálnými čísly. Jsou také velmi důležité v procesu učení, kdy se právě jejich úpravou dosahuje shody mezi výstupy sítě a výstupy požadovanými a tím v podstatě natrénování sítě.

Matematicky lze závislost mezi vstupy neuronu a jejich váhami obecně popsat pomocí operace *konfluence* (splynutí). V případě popisu jednoduchého modelu neuronu můžeme obecný operátor

konfluence nahradit lineární hodnotou operátoru, tzn. můžeme konfluenci nahradit prostým násobením. Rovnice závislosti pak může mít tvar:

$$z_i(k) = x_i(k) * w_i(k) \qquad \text{rovnice 1.0}$$

Práh neuronu, resp. prahovou hodnotu neuronu lze definovat jako hranici, kterou musí signál překonat aby byl neuronem „propuštěn“ dál a mohl se dál šířit neuronovou sítí. Hodnota prahu (Θ) tedy určuje, kdy je neuron aktivní, resp. neaktivní. Je-li hodnota vstupního signálu nižší než hodnota prahová, pak signál není šířen dál a naopak je-li hodnota vyšší, pak je signál propuštěn k dalšímu šíření a může se měnit v závislosti na oboru hodnot příslušné aktivační funkce až do určitého maxima.

V modelech umělých neuronů je však práh často používán k tomu, aby „posouval“ signál při vstupu do aktivační funkce. Při tomto přístupu se potom prahová hodnota aktivně podílí na hodnotě výstupu z neuronu. Při této úvaze je vhodné modelovat práh neuronu jako násobek synaptické váhy w_0 a 1 resp. -1. Číslice 1 resp. -1 (dále x_0) představují fiktivní vstup s konstantní hodnotou, podle směru šíření signálu, což záleží na řešené úloze. Neurony, u kterých je práh jediným vstupem, pak označujeme jako *sigma neurony*.

Agregační funkce *y_a* si klade za cíl sloučit vektor vstupního signálu do jedné skalární proměnné, která je přivedena na vstup aktivační funkce *f*. Stejně jako u konfluence zavedeme obecný operátor agregace *G*. Funkce potom přejde do tvaru:

$$y\_a(k) = \overset{n}{G}_{i=1} z_i(k) \qquad \text{rovnice 1.1}$$

Při výše zmíněném základním modelu můžeme opět zjednodušit operaci agregace a nahradit ji operací sumace. Po této úpravě, zavedení prahu a dosazení rovnice 1.1 dostaneme vztah:

$$y\_a(k) = \sum_{i=1}^{n} x_i(k) * w_i(k) + \Theta \qquad \text{rovnice 1.2}$$

Jak bylo řečeno, práh lze považovat za příklad speciálního vstupu a lze jej nahradit substitucí:

$$\Theta = w_0 * x_0 \qquad \text{rovnice1.3}$$

Rovnice agregační funkce se potom zjednoduší do tvaru:

$$y\_a(k) = \sum_{i=0}^{n} x_i(k) * w_i(k) \qquad \text{rovnice1.4}$$

Aktivační (*přenosová*) funkce *f(y_a)* má za úkol převést hodnotu vstupního potenciálu (agregační funkce) na hodnotu výstupní, která už je přímo přivedena na výstup neuronu. Neurony se někdy mohou dělit právě podle své přenosové funkce. Přenosových funkcí může být mnoho různých druhů a výběr této funkce je odvislý od charakteru řešené úlohy, nebo lze volit přenosovou funkci podle polohy daného neuronu v síti. Z toho vyplývá, že můžeme použít jiné funkce pro neurony ve skrytých vrstvách a jiné pro neurony ve vrstvě výstupní. Jen náznakem se můžeme zmínit, že lze použít jak funkce spojité, tak diskrétní, lineární i nelineární.

Z nejčastěji používaných funkcí můžeme jmenovat funkci skokovou, hyperbolický tangens, či funkci gaussovskou.

Vše, co bylo o neuronu řečeno v této kapitole se však týká pouze základního (formálního) modelu neuronu. V teorii umělých neuronových sítí se ale objevují i jiné typy modelů, které se mohou lišit od základního (a samozřejmě i od sebe navzájem) například tvarem agregačních, či aktivačních funkcí. S těmito typy lze modelovat i složitější funkce biologických neuronů, které se základním modelem vytvořit nelze.

Jako názornou ukázku můžeme zmínit funkci, která je používána velmi často jako aktivační a nese jméno *radiální bázová funkce*. Někdy se u ní také používá označení RBF. Nutno zmínit, že některé klasifikační problémy, či aproximace některých funkcí lze s touto funkcí řešit poměrně elegantně. Tato funkce má tvar:

$$y\_a(k) = \sqrt{\sum_{i=1}^{n} [x_i(k) * w_i(k)]^2} \qquad \text{rovnice 1.5}$$

Oproti běžným aktivačním funkcím má tato extrém a to buď maximum nebo minimum. Nejčastěji je jako RBF používána gaussova funkce.

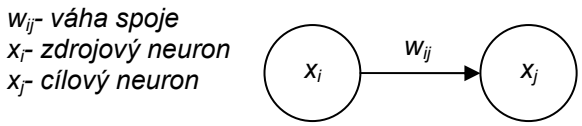
V závěru bych se ještě rád zmínil o dalším dělení neuronů, podle aktivačních funkcí, a to do několika tzv. *generací*. Do prvních dvou generací patří neurony McCullochova-Pittsova typu. První generaci obsadil model s nespojitou přenosovou funkcí, přičemž do druhé patří jeho spojitý kolega. Další generace už jsou tvořeny typy složitějšími, které jsou ovšem schopny popsat složitější chování a jsou komponovány z typů jednodušších. Lze však říci, že pro většinu běžných úloh si vystačíme s modely prvních dvou generací. Pro bližší informace o této kapitole viz [2].

1.2 Model umělé neuronové sítě

Umělá neuronová síť si klade za cíl určitým způsobem napodobit fungování sítě biologické. Protože biologická neuronová síť je tvořena množstvím propojených biologických neuronů, pak i síť umělá musí být sestavena s pomocí neuronů, ale tentokráte už těch umělých (viz předchozí podkapitola). Biologická síť je tvořena miliardami „přírodních“ neuronů, ovšem takto velké počty neuronů jsou v umělých sítích neúnosné z důvodu relativně slabé výpočetní síly dnešních počítačů a hlavně nutnosti provádět veškeré operace sériově. V tomto ohledu má počítač oproti mozku velkou nevýhodu, protože mozek je schopen masivního paralelního zpracování a díky tomu může i při menší rychlosti přenosu informace zpracovat mnohokrát více dat.

Proto je třeba hledat mechanismy, které nám dovolí vytvářet sítě jednoduché avšak takové, které jsou schopny nalézt dané řešení s minimální možnou odchylkou. Nutno ale dodat, že biologická neuronová síť musí umět řešit mnohem větší oblast činností, jako například, u člověka, správnou

funkci orgánů, koordinaci pohybů, či myšlení. V umělé síti máme tu výhodu, že si můžeme zvolit úsek reality, který má naše síť umět napodobit a tento úsek se pak rovná řešenému problému.



obr. 1.4 Spojení neuronů

Umělá neuronová síť (dále jen neuronová síť) může být znázorněna obecně libovolným orientovaným grafem, kde uzly jsou jednotlivé neurony a orientované hrany představují propojení sítě. Příklad spojení dvou neuronů je uveden na obrázku 1.4. Podle spojení neuronů na obrázku můžeme dělit neurony na *presynaptické* (před synapsí), což v našem případě představuje neuron x_i a *postsynaptické* (za synapsí), což je zbývající neuron x_j.

Pod pojmem topologie (nebo také struktura, architektura, geometrie a jiné) si můžeme představit konkrétní rozložení a propojení neuronů dané sítě. Topologie většiny prakticky používaných sítí bývá vrstevnatá, ale můžeme se setkat i s topologií mřížkovou. Síti s takovýmto rozložením neuronů se pak říká mapa.

Při organizaci sítě do vrstev můžeme rozlišovat různé druhy vrstev podle jejich umístění v síti. První vrstvou je vrstva vstupní, do níž přicházejí informace z okolí. Po průchodu informace vstupní vrstvou se dostane do vrstev skrytých. Těchto vrstev může být libovolný počet a zpravidla mají majoritní podíl na chování sítě. Za nimi přijde vrstva výstupní, která už nám dá nějaký konkrétní výsledek. Podle pojmenování vrstev jsou pojmenovány i neurony příslušející dané vrstvě(vstupní, skryté, výstupní). Jak už bylo řečeno výše, právě propojení vrstev a počty neuronů ve vrstvách tvoří architekturu sítě a ovlivňují tak její chování. Vlastní propojení může být realizováno mezi neurony libovolných vrstev, nejčastěji jsou však propojeny neurony buď v jedné vrstvě(*laterální spoj*) nebo ve dvou sousedních vrstvách buď v dopředném směru nebo i ve směru zpět(*rekurentní spoj*).

Vstupní vrstva je tvořena zdrojovými uzly(*vstupní terminály*) a slouží k přijetí signálu z okolí a jeho následnému rozdělení pro přenos do vnitřních vrstev. Můžeme ji chápat jako pasivní, protože signál pouze rozdělí a přepošle do následující, skryté vrstvy. Proto v praxi i podle příslušné české technické normy není započítávána do počtu vrstev sítě. Počet vrstev je počítán od první skryté vrstvy až k vrstvě výstupní.

Vrstvy skryté slouží k co nejlepší aproximaci požadované funkce. Vrstva výstupní slouží k závěrečné úpravě signálu a jeho předání do okolí.

Podle toku signálu můžeme sítě rozdělit na *dopředné*, kde se signál šíří pouze jedním směrem a to od vrstvy vstupní k vrstvě výstupní, tzn. orientovaný graf netvoří smyčky a *rekurentní*, kde se signál může vracet a v grafu se mohou objevit cykly. V rekurentních sítích je někdy obtížné určit vstupní a výstupní vrstvy.

Je potřeba podotknout, že výběr typu sítě je velmi důležitý, protože odráží efektivitu dané sítě pro tu, kterou úlohu. Bohužel zatím neexistuje obecný návod na výběr sítě, či vytvoření nové podle

typu úlohy. Většina sítí je poplatná pouze určitému typu úlohy a výběr vhodné sítě je proto dán doporučením z literatury nebo je to otázka simulačních či jiných experimentů.

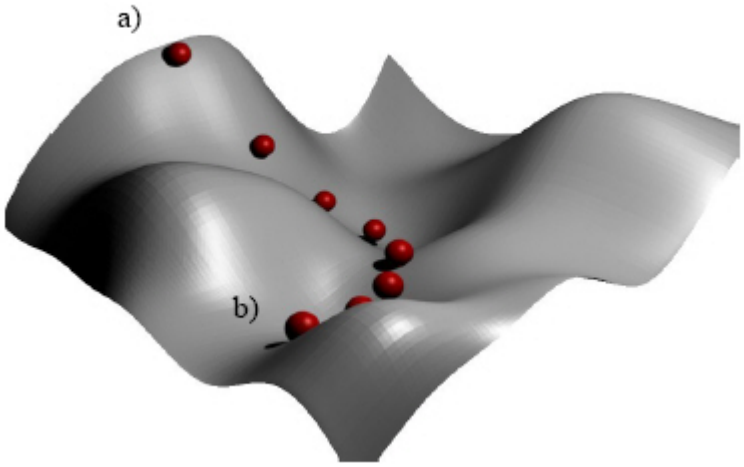
1.2.1 Učení neuronové sítě

Proces učení je pro neuronové sítě typický. Aby byla síť vůbec použitelná pro řešený problém, je potřeba ji po vytvoření naučit správně reagovat na vstupy, které jí předáváme. Učení je dynamický proces, při kterém dochází k úpravě nastavitelných parametrů sítě. Hlavní je úprava jednotlivých vah u spojených neuronů. Upravovat lze i další parametry, jako například strmosti aktivačních funkcí jednotlivých neuronů, či dokonce topologii sítě.

Učení je vlastně proces, při kterém je počítána tzv. energetická funkce(viz 1.2.1.2) a úpravou volitelných parametrů sítě se hledá její globální minimum. Tato funkce je n-rozměrná, kde:

$$n = x + 1$$

V této rovnici x představuje počet vstupů sítě. Hledání minima energetické funkce si lze představit jako průchod tzv. nadplochou(viz obrázek 1.5). Na počátku je výstup sítě(znázorněný červenou kuličkou) v náhodném bodě(a) prostoru a při učení se síť snaží dosáhnout globálního minima této nadplochy(b). Obrázek 1.5 je pouze ilustrativní, ve skutečnosti mívá tato nadplocha obvykle mnoho rozměrů, což na obrázku nelze znázornit.



obr. 1.5 průchod nadplochou při hledání minima energetické funkce

Vlastní průchod je realizován úpravou nastavitelných parametrů sítě, což znamená, že při úpravě parametrů je výsledek energetické funkce jiný(doufejme, že menší). K těmto parametrům patří právě váhy jednotlivých spojení a strmosti aktivačních funkcí neuronů. Strmosti určují, kterým směrem se v prostoru při učení sít vydá. Nastavení moc velkých strmostí vede k tomu, že síť může při hledání „překročit“ globální minimum a pokračovat nesprávným směrem. Naopak nastavení moc

malých strmostí vede k příliš dlouhému učení sítě. S tímto také souvisí parametr *learning rate*, který určuje rychlost učení. Tento parametr lze vysvětlit jako velikost kroku, který urazí síť v jednom kroku učení. Tyto dva parametry majoritním způsobem ovlivňují učení sítě. V některých přístupech k učení se lze setkat s přístupem, kdy se při učení dokonce mění topologie sítě za účelem dosažení lepší odezvy sítě. Učení bývá charakterizováno určitým algoritmem učení. Použitý algoritmus potom vypovídá a o přístupu k učení sítě.

Učení lze provádět induktivní nebo deduktivní metodou. Při učení induktivní metodou vyvozujeme všeobecně platné závěry z pozorování určitých jevů. Potom hovoříme o syntetickém přístupu k učení. V případě deduktivní metody pozorujeme jev jeden a jeho analýzou potom dospějeme k určitému závěru. Toto je potom nazýváno analytický přístup. Nejčastěji je využíváno učení induktivní metodou. Tento přístup se dá ještě rozdělit na učení s učitelem, kde je vyžadována liská podpora a učení bez učitele, kde lidská podpora vyžadována není. V tomto případě se síť dokáže naučit sama a činí tak prostřednictvím analýzy svých výstupů. Před bližším popisem učení s učitelem a bez učitele se ještě zmíním o tzv. Hebbově zákonu učení.

1.2.1.1 Hebbův zákon učení

V současnosti představuje základ všech algoritmů učení. Jeho tvůrcem je kanadský psycholog Donald Hebb a poprvé ho publikoval už v roce 1949. Tento zákon popisuje algoritmus změny synaptických vah spojů mezi neurony v biologických sítích.

V principu je tento zákon založen na předpokladu, že pokud jsou dva neurony aktivní je jejich vazba posílena a jsou-li pasivní je jejich vazba zeslabena. Pokud je aktivní pouze jeden z neuronů, pak se vazba nemění. Toto pravidlo počítá s předpokladem, že neurony dosahují pouze dvou stavů a to aktivní/neaktivní. Matematicky lze toto pravidlo zapsat následujícím předpisem:

$$\Delta w_{ij} = \alpha x_i(k) x_j(k) \qquad \text{rovnice 1.6}$$

x_i v rovnici představuje presynaptický stav neuronu, x_j představuje postsynaptický stav neuronu a parametr α udává zesilující koeficient tzv. rychlost učení(výše zmíněná learning rate).

1.2.1.2 Učení s učitelem

Učení s učitelem se také někdy říká chybové učení. Jedná se o přístup, kdy je do sítě vždy poslána dvojice hodnot. Jsou to hodnoty vstupní a hodnoty výstupní zadané učitelem. Reálný výstup sítě(y_i‘), který vznikne jako odezva sítě na daný vstup, se potom porovnává s výstupem požadovaným(y_i) a podle velikosti odchylky mezi těmito hodnotami se upravují váhy spojů sítě. Při tomto procesu je pochopitelně snahou dosáhnout co nejmenší odchylky mezi výstupem požadovaným a výstupem reálným. Velikost odchylky lze spočítat tzv. *energetickou funkcí*, která je také někdy nazývána funkcí chybovou. Tuto funkci lze zapsat jako:

$$E(s) = \sum_{i=1}^{M} \left[y'_i(s,k) - y_i(k) \right]^2 \qquad \text{rovnice 1.7}$$

M v rovnici je celkový počet vzorů trénovaní množiny, *k* je pořadové číslo vzoru trénovací množiny, *y(k)* je požadovaný výstup a *y(s,k)* je reálný výstup sítě. Slovně lze říci, že funkce počítá celkovou kvadratickou odchylku výstupu sítě.

1.2.1.3 Učení bez učitele

Tento přístup je založen na schopnosti neuronových sítí hledat ve vstupech podobné vlastnosti a třídit pak vstupy podle těchto vlastností. Podobné vektory se potom sdružují do tzv. *shluků* nebo také *map*. Podobné mapy jsou také v jednotlivých částech mozku. Učící algoritmus nemá hodnoty výstupů, protože tyto hodnoty neznáme. Tento princip učení se tedy používá právě v případech, kdy neznáme výstupní hodnoty. Principem učení je výpočet vzdáleností mezi vzory a aktuálními hodnotami vstupu. Jednotlivé vstupy se podle těchto vzdáleností organizují do zmiňovaných shluků.

2 Motivace

Jak bylo zmíněno v úvodu detekce obličeje je relativně složitý problém, jehož řešení není zatím dotaženo do úplného konce a v současnosti není dostupný systém, který by daný problém řešil se stoprocentní úspěšností. Tento úkol přináší také určitou výzvu, která člověka může motivovat k dosažení výsledků a zároveň zdokonalení sebe sama. V této kapitole budou rozebrány problémy při detekci obličeje a zároveň budou zmíněny některé aplikační oblasti, ve kterých je detekce obličeje relevantním problémem. Posledním bodem této kapitoly je pak stručný přehled některých dalších metod pro detekci obličeje.

2.1 Detekce obličeje

Při snaze detekovat obličej v obraze vyvstává řada problémů, které detekci ztěžují a kvůli kterým pravděpodobně neexistuje ucelený systém, který by detekování obličeje řešil ve všech jeho aspektech. Podproblémů, které je potřeba vyřešit, je celá řada a je proto nutné je všechny pochopit a nalézt cestu k úspěšnému řešení, které by v závěru vykazovalo co nejlepší výsledky. V této kapitole budou rozebrány některé problémy, které ukazují komplexnost zadaného úkolu.

2.1.1 Výrazy

Následující obrázky demonstrují, jak je obtížné dopředu definovat, jak bude obličej na obrázku vypadat. Je mnoho faktorů, které se mohou měnit a které dopředu nemůžeme ovlivnit. Nejdůležitějším faktorem je bezpochyby směr, kterým je natočena lidská hlava na obrázku, protože k hlavě natočené do pravého úhlu(viz. žena na obrázku 3.1 v horní řadě vpravo), kterou vidíme z profilu, nelze přistupovat stejně jako k obličeji, který je otočen směrem ke kameře.

obr. 3.1 ukázka různých póz různých jedinců

Tato práce se však soustřeďuje pouze na detekci obličejů otočených směrem ke kameře.

2.1.2 Osvětlení

Předchozí ukázka znázorňovala varianci obličeje vzhledem k jeho otočení. Dalším faktorem je osvětlení. Fotky jsou totiž pořizovány za různého osvětlení, které může radikálně změnit vzhled obličeje. Následuje několik ukázek osvětlení scény.

obr. 3.2 ukázka různého osvětlení

2.1.3 Tvar tváře

Posledním faktorem v rozeznávání je tvar obličeje samotného. Kdyby měl obličej pevně zadaný tvar, mohli bychom celkovou scénu projít a nalézt kandidátní místa, kde by se mohl vyskytovat obličej. Zbytek scény bychom mohli při detekci zanedbat, ale kvůli různým tvarům obličeje, různým

výrazům, kterých je v podstatě také nekonečné množství a v neposlední řadě i kvůli výše zmiňovanému osvětlení nemůžeme pozadí nijak jednoduše zanedbat.

2.2 Využití

Lokalizace obličeje v obrázku skýtá mnoho možností využití. Jako první bych zmínil, že detekce obličeje může sloužit jako předzpracovací krok k rozpoznání obličeje. Tento přístup se dá použít ve všech sledovacích systémech. Při spojení systému na detekci a databáze obličejů, které mají být rozpoznány si lze, poměrně jednoduše, představit výsledný systém sloužící k rozpoznání některých obličejů. Pěkným příkladem je nasazení takovéto aplikace například na letištích, kde se pohybuje velké množství osob a v otázkách bezpečnosti je stále potřeba lidské podpory pro sledování prostoru pomocí kamerového systému. Automatizací detekce, či dokonce rozpoznání pak dokážeme zjednodušit práci lidí. Lidská podpora by pak se omezila pouze na verifikaci výstupů systému.

Pokud odhlédneme od spojení detekce a rozpoznání, můžeme za využitelnou prohlásit i detekci samu o sobě. Tento přístup je pak použitelný v bezpečnosti v situacích, kde nám stačí pouze sledovat monitorovaný prostor a zachytit případného člověka pohybujícího se po scéně.

Tímto byly naznačeny některé možnosti využití systému pro detekci. Možností je jistě více a případné další si jistě každý doplní sám.

2.3 Další metody detekce obličeje

Jak už bylo zmíněno detekce obličeje není triviální problém, ale je zřejmé, že přístup k řešení tohoto problému využívající neuronovou síť je použitelný(viz např. [6]). Co ale ještě nebylo zmíněno je informace, že tento problém se dá řešit i jinými metodami, než pouze neuronovou sítí. Tato kapitola postihuje krátký výčet metod s jejich stručným popisem.

V současné době existuje, kromě neuronových sítí, několik dalších metod, které lze charakterizovat podle přístupů, kterými se snaží vyřešit detekci obličeje. Začal bych tzv. *knowledge-based methods*(vytvoření pravidel, které popisují lidský obličej). Jako další zmíním tzv. *feature invariant approaches*(hledá znaky, které obsahuje každý obličej). Jako poslední bych nastínil princip tzv. *template matching methods*(vytváření šablon pro jednotlivé části obličeje). Názvy nejsou překládány do češtiny, protože by to znalého čtenáře zbytečně rozptylovalo a neznalému čtenáři by to stejně vysvětlení neposkytlo.

Přístupů je ještě více, ale tato práce nemá za úkol popsat vyčerpávajícím způsobem všechny metody detekce obličeje, proto pouze v nastíním princip některých dalších metod. Principy těchto metod jsou vytaženy z [3].

- **knowledge-based methods:** Tento přístup využívá metodu *shora-dolů*. Snaží se popsat lidský obličej pomocí pravidel. Potom při testování zjišťuje zda daný obrázek, resp. výřez

obrázku odpovídá daným pravidlům. Tato metoda je vhodná především pro obrázky, jejichž pozadí není příliš komplexní. Problémy této metody spočívají hlavně v tom, že nejsme schopni popsat naprosto přesně lidský obličej a proto analýza obrázků, které jsou hodně členěné může přinášet mnoho *falešných detekcí*, tzn. že metoda „najde“ obličej, i když v daném místě žádný není. Je také obtížné vyjádřit pravidly všechny možné polohy obličejů vzhledem ke kameře a zároveň je obtížné popsat všechny možné výrazy v obličeji.

- **feature invariant approaches:** Postupuje metodou zdola-nahoru. Snaží se postihnout typické rysy tváře, jako například elipsu obličeje, barvu obličeje, oči a podobné typické rysy. Problém se potom snaží vyřešit hledáním správného geometrického rozložení jednotlivých rysů, přičemž při nalezení každého rysu přidá rys do skupiny, která je verifikována(ověřena) tzv. Bayesovskou sítí. Tato metoda je netečná vůči natočení obličeje, zatímco je velice citlivá na osvětlení, či šum v obraze. Zároveň má problémy s detekcí v komplexním pozadí.

- **template matching methods:** Tento přístup je založen na metodě vytvoření šablony člověkem. Šablona je vytvořena z různých oblastí tváře, které definuje na základě kontur v obličeji. Kontury jsou pozorovány v příkladech tváří při velmi nízkém rozlišení a díky těmto pozorováním je vytvořena příslušná šablona. Při rozpoznávání metoda zkoumá průměrné hodnoty pixelů v blízkém okolí pro test shody s příslušnou částí šablony. Velkou nevýhodou této metody je nutnost inicializace vzoru blízko obličeje z čehož vyplývá nutnost využití nějaké sekundární metody, která buď redukuje okolní prostor obrázku, abychom ho nemuseli procházet celý, nebo najde na obrázku kandidátní místa, která jsou potom pomocí šablony pouze verifikována. Další nevýhodou je opět složitost vyjádření všech možných póz obličejů.

3 Návrh systému

Detektor je systém, který umožňuje detekovat obličej ve vstupním obrázku a zároveň je schopen reagovat příslušným způsobem. V našem případě detektor reaguje na rozeznaný obličej vykreslením čtverečku do místa, kde obličej leží.

Detektor pracuje s neuronovou sítí a využívá některé její příjemné implicitní vlastnosti. Je to schopnost zobecňování, která dělá z neuronové sítě mocný nástroj při řešení problému klasifikace vzorů. Zároveň využíváme další příjemnou vlastnost neuronové sítě a to relativní jednoduchost modelu oproti poměrně náročnému úkolu. Samozřejmostí jsou některé úpravy na datech, které síti pomáhají, protože použití sítě samotné, bez jakékoli přípravy dat, by vykazovalo vysokou chybovost(viz [4]). Proto využijeme některé předzpracovací kroky, které nám upraví data do takové podoby, aby se alespoň částečně odstínily některé nežádoucí efekty popsané v podkapitole 2.1.

3.1 Schéma systému

Toto schéma zjednodušeným způsobem ukazuje, jaký je průběh operací při testování jednoho obrázku a zároveň modeluje architekturu detektoru. V dalším textu budou vysvětleny jednotlivé kroky resp. součásti detektoru.

Následuje navržené schéma systému, k jehož lepšímu pochopení je zaveden jistý způsob značení, který je třeba vysvětlit. Ve schématu jsou oválnými tvary znázorněny veškeré operace, v obdélnících jsou data a kruhově je naznačena síť. Schéma nedokáže v plné šíři vysvětlit fázi testování a proto je nutné toto znázornění podpořit slovním zápisem použitého algoritmu.

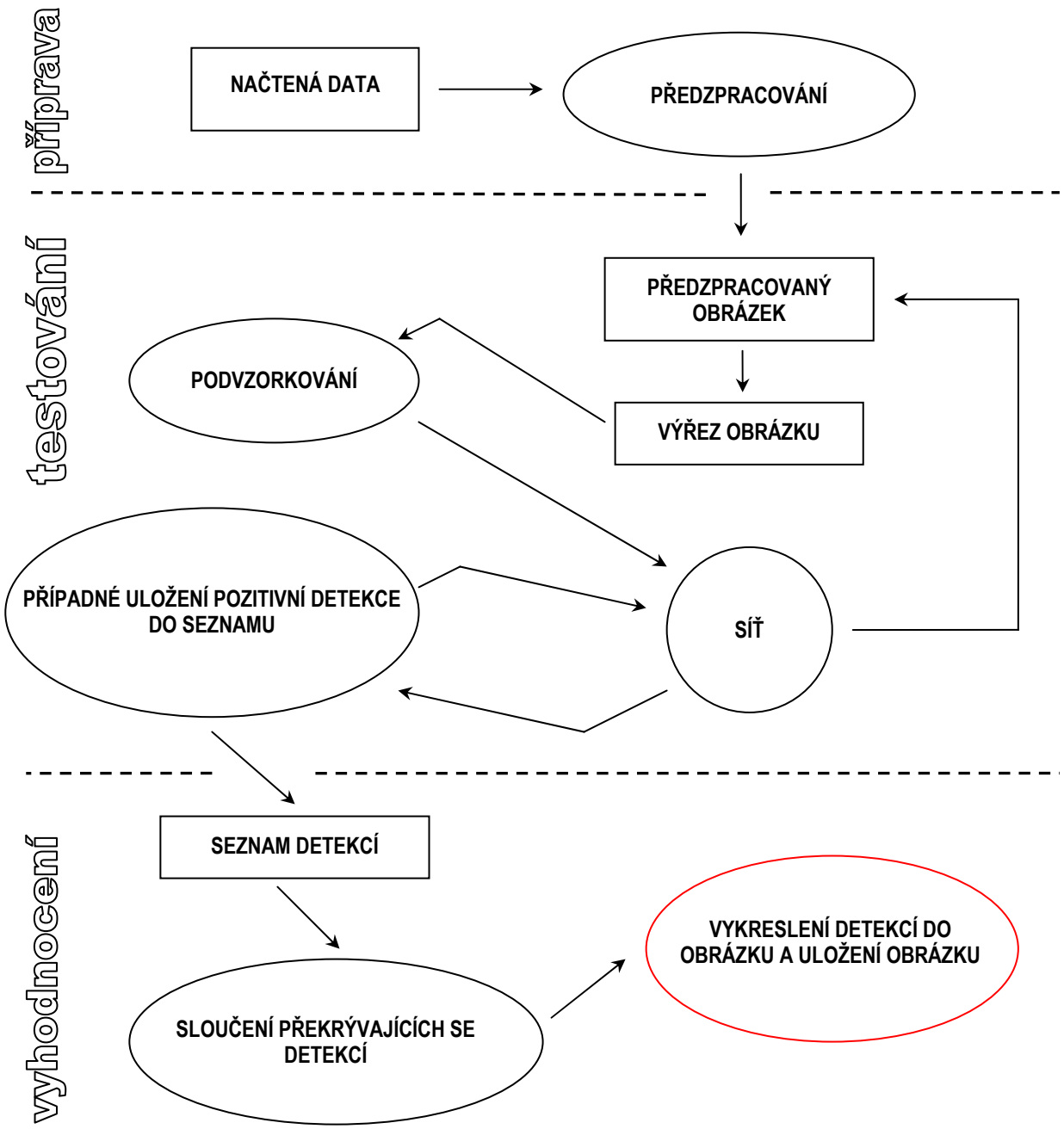


schéma 3.1 – návrh systému

3.2 Algoritmus detekce

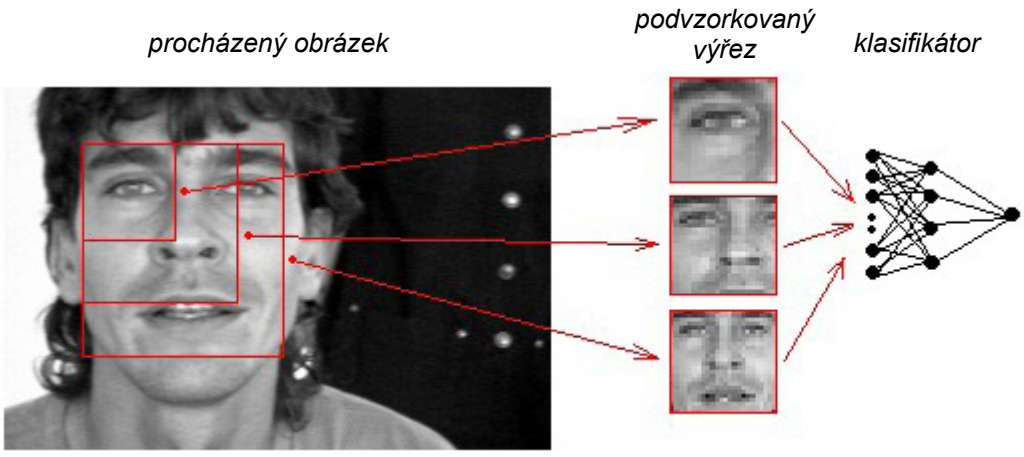
7. inicializuj detekční okénko(čtverec) na levý horní roh obrázku a nastav velikost strany na menší ze stran obrázku
8. pokud nejsi v prvním cyklu, podvzorkuj okénko danou konstantou
9. podvzorkuj výřez v okénku na danou velikost pro síť(19x19)
10. spočítej velikost odezvy a pokud je větší než stanovený práh, ulož do seznamu
11. posuň okénko na další pozici
12. pokud není dosaženo konce obrázku vrať se na bod 3.
13. pokud nemá okénko minimální velikost vrať se na bod 2.

3.3 Podvzorkování

Tato podkapitola popisuje metodu zvanou podvzorkování a je záměrně umístěna hned za pasáží, ve které tento termín poprvé v této práci zaznívá. Myslím si, že ne každý, kdo se s touto prací setká, bude daný termín znát. Pro přehlednost je tedy vysvětlení tohoto termínu(metody) umístěno zde.

Podvzorkování neboli *podsamplování*(resp. *subsampling*) je metoda úpravy obrázku, která nám umožní detekovat obličej větší, než je zvolená velikost okénka, ze kterého posíláme jednotlivé pixely[1] na vstupy sítě.

Princip spočívá ve změně rozlišení obrázku. V našem případě podvzorkujeme vždy výřez daný průchodovým okénkem na stejnou velikost a to 19x19 pixelů, což je velikost, kterou přijímá naše síť. Pro lepší pochopení následuje ukázka obrázku ilustrujícího podvzorkování.



obr. 3.1 ukázka převzorkování různých částí obrázku

[1] Obrazové body, jinak řečeno nejmenší jednotka zobrazení

3.3.1 Interpolace

Tento termín úzce souvisí s termínem podvzorkování. Při podvzorkování se v obrázku na každém řádku i v každém sloupci zmenší počet pixelů a interpolace je metoda, která nám říká, jakým způsobem se budou počítat hodnoty nových pixelů.

Interpolací je několik druhů a v této práci je využita *bikubická* interpolace(viz [5]), což je jeden s nejpoužívanějších druhů. Hodnota nového pixelu se spočítá jako váhovaný průměr hodnot nejbližších 16 pixelů v mřížce 4x4. Vzoreček, pro výpočet hodnoty pixelu je potom:

$$\sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij} x^i y^j \qquad \text{rovnice 3.1}$$

kde x,y značí souřadnice nového bodu(pixelu), a je hodnota jednoho ze 16 pixelů v okolním poli a i a j jsou souřadnice tohoto pixelu.

3.4 Předzpracování

Předzpracování je velice důležitým krokem před samotnou detekcí, protože umožňuje odstranit nebo alespoň redukovat některé problémy související se samotnou detekcí. Způsobů přípravy obrázku pro síť je mnoho, ovšem pouze některé byly vybrány do této práce. Velikou inspiraci při výběru předzpracovacích kroků poskytla [4] a zároveň mě tato práce navedla na pochopení důležitosti vybraných kroků. Nutno říci, že veškeré předzpracovací kroky je potřeba provádět nejen při učení neuronové sítě, ale také při samotném testování. Následuje výčet předzpracovacích kroků spolu s jejich popisem a vyjádřením míry jejich potřebnosti.

3.4.1 Normalizace

Toto je jeden z nejjednodušších předzpracovacích kroků, ale zároveň nesmíme podceňovat jeho důležitost. Normalizací se v našem případě myslí převod rozsahu hodnot, kterých mohou nabývat jednotlivé pixely obrázku.

Už při načítání obrázku do systému je obrázek převeden na *šedotónový*[2], z čehož vyplývá, že systém je schopen dále zpracovávat pouze šedotónové obrázky. Činnost systému to ovšem neomezuje, protože nepracuje s žádnými operacemi, které by zpracovávaly barvu v obrázku. Standardně je tedy načten obrázek pouze s jedním *barevným kanálem*[3], což umožňuje pohodlnější zpracování. Hodnoty tohoto obrázku jsou po načtení v rozmezí 0..255 a je potřeba dostat je do rozsahu jiného a právě v tom spočívá princip normalizace. Jako normalizovaný rozsah byl zvolen rozsah hodnot 0..1.

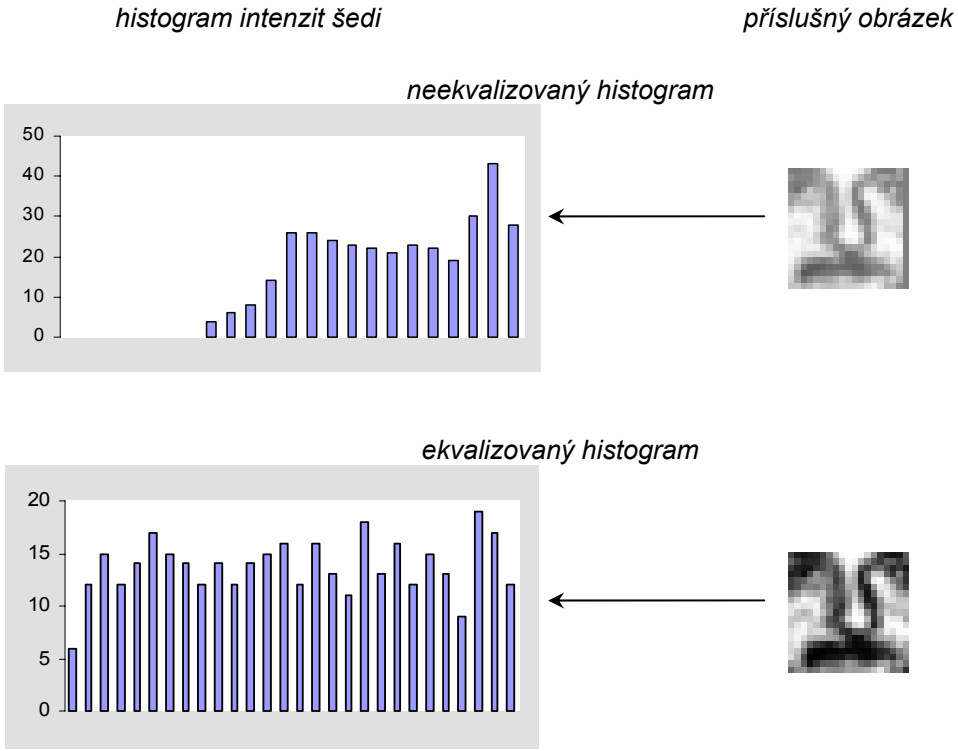
[2] Obrázek, který obsahuje pouze různě intenzivní šedou barvu(a samozřejmě i černou a bílou)
[3] Počet kanálů udává barevnou hloubku

3.4.2 Ekvalizace histogramu

Ekvalizace histogramu je předzpracovací krok, který má na úspěšnost detekce největší dopad.

- **Princip:** Každý obrázek se skládá z pixelů. Počítáme s faktem, že zpracovávané obrázky jsou pouze šedotónové, tzn. že pixel si můžeme představit jako číslo mezi 0..255. Pokud bychom tedy sestavili histogram četností výskytů všech hodnot, dostaneme graf, který nám ukazuje rozložení odstínů v obrázku. Pokud dostaneme obrázek, který je velice tmavý nebo naopak velice světlý, jednotlivé sloupce histogramu pak budou soustředěny na levé respektive pravé straně. Podstata ekvalizace spočívá v rozložení tohoto grafu do celého rozsahu hodnot, v našem případě tedy 0..255.

Zjednodušeně se dá říci, že ekvalizace histogramu zvýší kontrast v obrázku. Pro ilustraci přikládám ukázku ekvalizace na náhodně vybraných trénovacích obličejových datech.

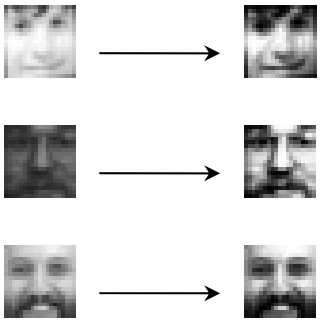


obr. 3.2 ukázka histogramu před a po ekvalizaci

Z předchozího obrázku je vidět vliv ekvalizace histogramu na obrázek a zároveň je znázorněno rozložení intenzit před a po ekvalizaci.

Na konec této podkapitoly je přiloženo několik příkladů obrázků z trénovací množiny a zároveň jsou ukázány jejich ekvalizované verze pro ukázku, jak dokáže metoda ekvalizace histogramu smazat rozdíly mezi světlými a tmavými obrázky.

Původní velikosti obrázků jsou 19x19 pixelů, pro větší názornost jsou však roztaženy na větší velikost.



obr. 3.3 ukázka ekvalizace histogramu pro různě tmavé obličeje

3.5 Neuronová síť

Neuronová síť představuje výkonný prvek samotné detekce a potažmo i celého systému. Za použitý typ sítě byl zvolen vícevrstvý perceptron, nejen díky jeho rozšířenosti a všeobecnému povědomí o tom, že je schopen řešit daný problém, ale také na základě [4].

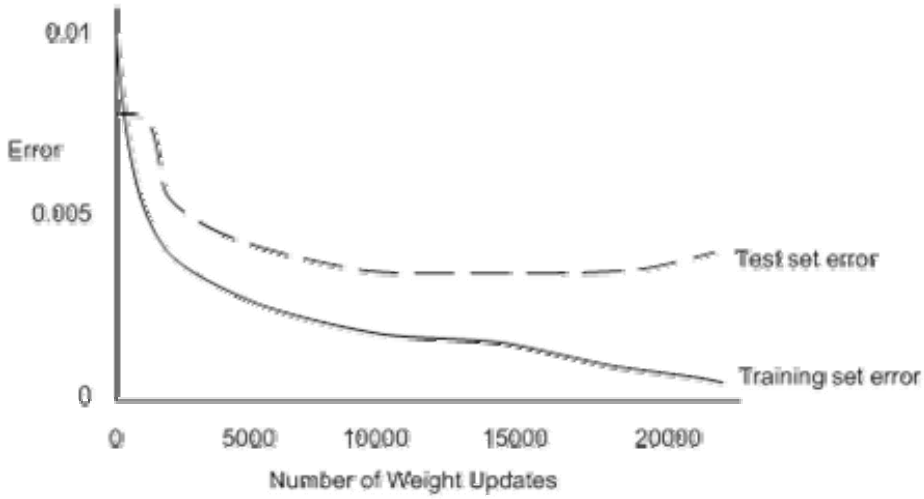
V [4] je také zmíněna architektura sítě, která vykazuje lepší výsledky co se týče rychlosti i schopnosti rozpoznávání. Více bude o této problematice zmíněno v kapitole 4.2.

3.5.1 Natrénování neuronové sítě

Trénování neuronové sítě je klíčovým prvkem, který posléze ovlivňuje úspěšnost celého systému. Na následujícím obrázku(3.4) je ukázka typického průběhu chybové funkce. Obrázek ovšem znázorňuje nejen průběh chyby na trénovacích datech, ale zároveň průběh chyby na testovacích datech.

Z grafu je vidět, že chyba v prvních fázích trénování klesá velice rychle a to jak na trénovacích, tak na testovacích datech. Postupem času se rychlost klesání zpomaluje a jak je vidět, tak za delší dobu už chyba na trénovacích datech klesá velice pomalu a na testovacích datech dokonce roste. Je proto potřeba nalézt při trénování ten správný okamžik a trénování zastavit. Jakmile totiž začne růst chyba na testovacích datech, rapidním tempem začne při testování přibývat falešných detekcí. Při studiu grafu však musíme mít na paměti, že se jedná o typický průběh, což znamená, že tvar křivky

odpovídá studovanému problému, ale velikost chyby i počet epoch se může měnit v závislosti na použitých parametrech sítě.

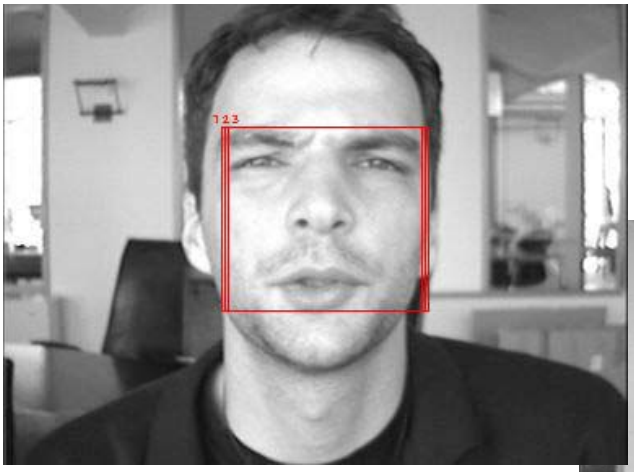


obr. 3.4 typický průběh chybové funkce

3.6 Překrývající se detekce

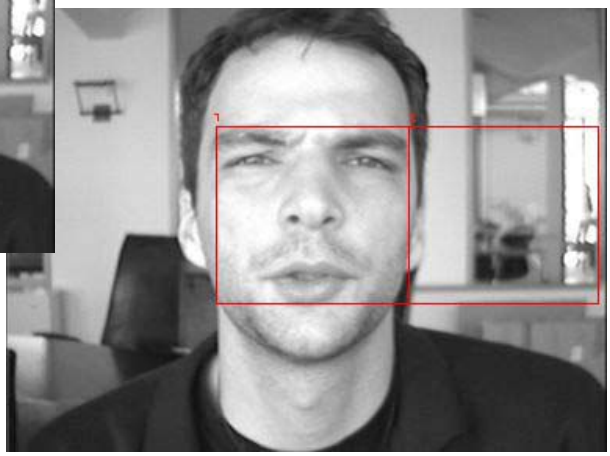
Při řešení vícenásobných detekcí musíme řešit dva typické problémy. První problém spočívá už v samotném procházení obrázku. Obrázek je procházen detekčním okénkem po velmi malých krocích, protože nevíme, kde se obličej může nacházet, musíme projít celou scénu sekvenčně. Krok použitý při tomto průchodu je 2 pixely. Jak je vidět, krok je opravdu velmi malý a tak se lehce stane, že po první detekci obličeje v obrázku, když se detekční okénko posune, detekuje obličej znovu. Odbourání tohoto typu překrývajících se detekcí je velmi snadné a poměrně intuitivní. Stačí se totiž po detekci obličeje posunout ve zkoumaném prostoru o velikost detekčního okénka dál a tento problém zmizí.

Následující obrázek(3.5) ukazuje v postup detekčního okénka v případě kladné odezvy neuronové sítě. V levé části je vidět, že kdyby detekční okénko neustále postupovalo stejným krokem, do výsledku by se zaneslo mnoho zbytečných detekcí. Na pravé straně je oproti tomu ukázka jednoduchého odbourání tohoto problému.



původní přístup

modifikovaný přístup

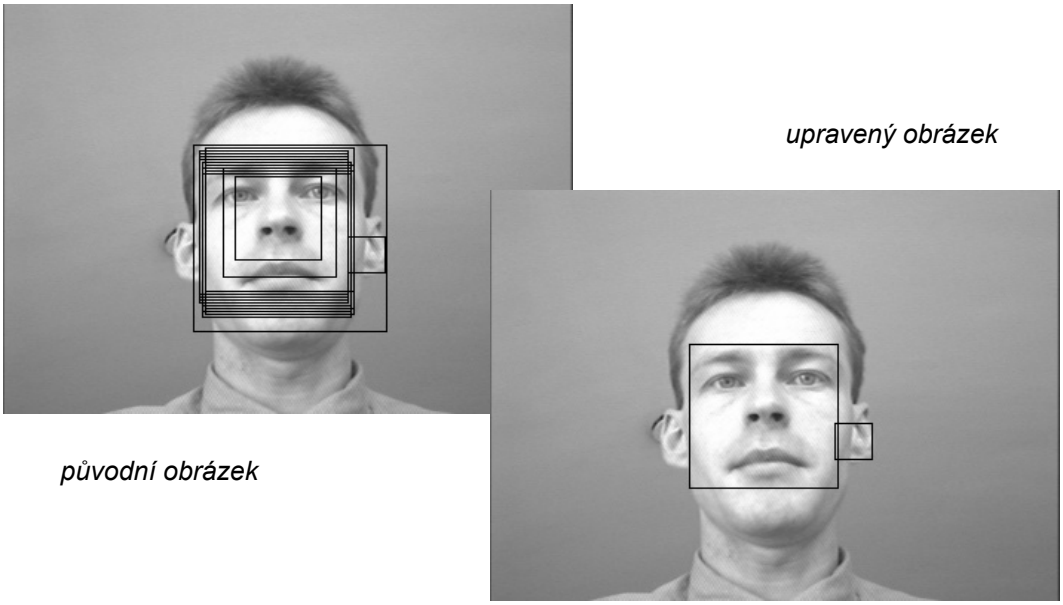


obr 3.5 překročení o velikost okénka

Druhým typickým problémem jsou vícenásobné detekce při podvzorkování(viz podkapitola 3.3). Jak už bylo řečeno, obrázek je nutno podvzorkovat, abychom byli schopni detekovat obličeje větší než detekční okénko. Pokud je ale obličej například téměř přes celý obrázek, může k jeho detekcí dojít přes několik úrovní podvzorkování a máme opět několik překrývajících se detekcí, které je nutno spojit, pokud možno, do jedné výsledné.

Řešení tohoto problému je už poněkud obtížnější, protože jednotlivé detekce si ukládáme tak, jak jdou za sebou a není zaručeno, že budou uloženy ve správném pořadí. Je proto nutné vymyslet metodu, kterou sloučíme co nejvíce k sobě příslušejících detekcí.

Návrh této metody vychází z metody použité ve [4] a spočívá ve sloučení středů, které jsou od sebe vzdálené o určitou, předem stanovenou, toleranci. Zároveň je měněna velikost výsledné detekce a je spočítána jako průměr velikostí všech, ze kterých je počítána. Následuje ukázka obrázku s překrývajícími se detekcemi a ukázka obrázku, na který byla aplikována metoda eliminace podle středů. Oba obrázky jsou zmenšeny od své skutečné velikosti.



obr. 3.6 ukázka odstranění vícenásobných detekcí

3.7 Obličejové databáze

V této práci jsou použity dvě stěžejní databáze obrázků. První je databáze CBCL(Center for Biological & Computational Learning), která byla sestavena na Massachusetts Institute of Technology(MIT) a druhou je databáze společnosti Human Vision, která nese název BioID. Následuje krátký popis obou databází.

3.7.1 Databáze CBCL

Databáze CBCL je databází, která se skládá z malých obrázků(19x19 pixelů) obsahujících jak obličejová, tak neobličejová data a je rozdělena na dvě sekce. První sekce obsahuje data sloužící k natrénování sítě, zatímco druhá sekce nám umožňuje test úspěšnosti sítě.

	obličejové příklady	**neobličejové příklady**
trénovací data	2429	4548
testovací data	472	23573

Na této databázi byla natrénována neuronová síť pro tento projekt. Ukázky z databáze jsou například na obrázku 3.3.

3.7.2 Databáze BioID

Druhou použitou databází je databáze BioID. Jedná se pouze o sekundární databázi, která slouží k testu systému. Databáze obsahuje 1521 obrázků v rozlišení 384x286 pixelů. Obrázky zachycují obličeje lidí s různým výrazem a otočením tváře vůči kameře a při různém osvětlení. Příklad z databáze je na obrázku 3.6.

4 Implementace

Tato kapitola se soustřeďuje na strukturu programu a přibližuje propojení celého výsledného systému. Celý systém je implementován v jazyce C++, který nabízí jak možnost jak objektového, tak procedurálního programování. Pro účely co nejrychlejšího zpracování programu byla zvolena možnost procedurálního programování s tím, že je zároveň možno využít některé příjemné vlastnosti C++, jako například pohodlnější zpracování řetězců než v jazyce C.

V celém systému je dodržena jistá modularita, která od sebe odděluje několik logických celků, ze kterých je program složen. Celky jsou nazvány *moduly* a jsou specifické operacemi, které nabízejí. Moduly lze spodobnit s objekty v OON(objektově orientovaný návrh) s tím omezením, že nabízejí pouze operace, ale nikoliv instance, či vlastnosti. Zjednodušeně lze říci, že každý z modulů je uložen v jednom souboru a tvoří jednotný celek, což významnou měrou zvyšuje schopnost rozšiřitelnosti systému.

Podprogramy znázorněné ve schématu 4.1 potom pouze využívají služeb jednotlivých modulů. Jednotlivé podprogramy jsou přiblíženy v podkapitole 4.3.

4.1 Knihovny

V této práci byly použity dvě externí knihovny, které umožňují funkčnost celého systému. Jedná se o knihovnu pro práci s grafickými elementy a operacemi a druhá knihovna je použita pro neuronovou síť.

Jako první byla vybrána knihovna *openCV* od firmy Intel. Hlavním důvodem je rozmanitost operací, které poskytuje a také jednoduchost a intuitivnost práce s touto knihovnou. Nutno říci, že veškeré operace týkající se obrázkových dat mi poskytla tato knihovna.

`url: http://www.intel.com/technology/computing/opencv/`

Druhou knihovnou je *fann*(fast artificial neural network), což je velmi efektivně napsaná knihovna pro práci s neuronovými sítěmi. Práce s ní je také velmi jednoduchá a umožňuje veškeré operace potřebné k vytvoření, natrénování i testování sítě typu jednoduchý plně, či řídce, propojený perceptron a několika dalších. V této práci je však vyžita pouze síť typu vícevrstvý perceptron.

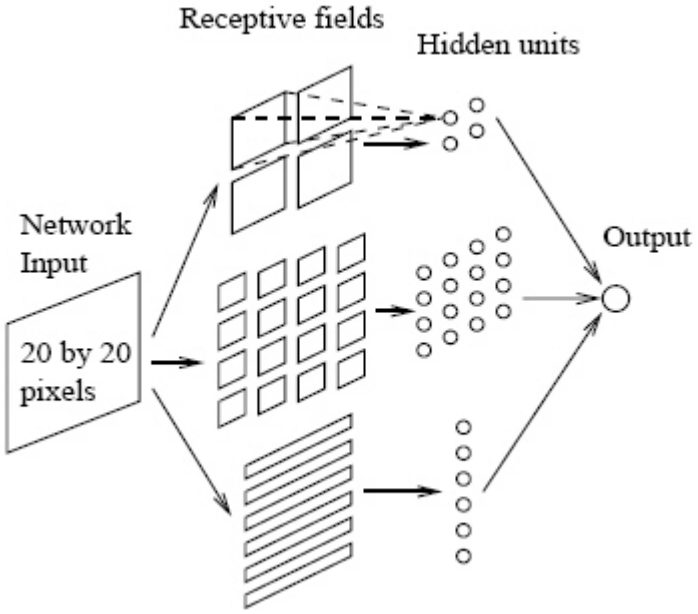
`url: http://leenissen.dk/fann/`

4.2 Architektura neuronové sítě

Tato podkapitola popisuje implementaci sítě typu vícevrstvý perceptron a je diskutována účinnost plně propojené sítě oproti síti s definovanou architekturou. Síť s definovanou architekturou je popsána v [4] a schematicky znázorněna na obrázku 4.1.

Při implementování sítě s architekturou však narážíme na problém s knihovnou fann, která zadanou architekturu bohužel nepodporuje, respektive nepodporuje úpravu architektury standardními prostředky.

Neschopnost knihovny implementovat tuto architekturu lze však obejít ručním přepsáním spojení jednotlivých neuronů. Nevýhoda tohoto přístupu je, že je potřeba ztrojnásobit vstup do sítě(pro každou z vnitřních vrstev jeden). Naopak výhoda tohoto přístupu spočívá v menším počtu spojení, což významnou měrou zvyšuje rychlost celé sítě.



obr. 4.1 síť se změněnou architekturou

Jen pro porovnání, plně propojený perceptron použitý v této práci obsahuje 361 neuronů ve vstupní vrstvě, 30 ve vrstvě skryté a 1 výstupní neuron. Dohromady tedy obsahuje ~10 900 spojení. Perceptron s architekturou upravenou podle obrázku potom obsahuje pouze ~1100 spojení, přičemž vstup je ztrojnásobený, což znamená, že vstupních neuronů je 1083. Neuronů skrytých je 26 a na výstupu je opět 1 neuron.

Jediným problémem takto upravené architektury v knihovně fann je onen ztrojnásobený vstup, protože každé načítání obrázku potom trvá mnohem déle. Snahou o maximální zefektivnění tohoto poměrně významnou měrou zpomalujícího prvku je předpočítání indexů, na které se budou ukládat jednotlivé pixely. Každý pixel se potom načte z obrázku pouze jednou a uloží se na 3 příslušná místa. Tímto se dosáhne maximálního zefektivnění načítání obrázku.

Upravená architektura je implementována jako součást systému, avšak nakonec není využita, a uživatelsky k ní není přístup, ale budoucí vývoj se bude zaměřovat na doladění této architektury a

bude jí věnována mnohem větší pozornost, protože má daleko větší potenciál(hlavně co se týče rychlosti), než plně propojený perceptron.

4.3 Složení programu

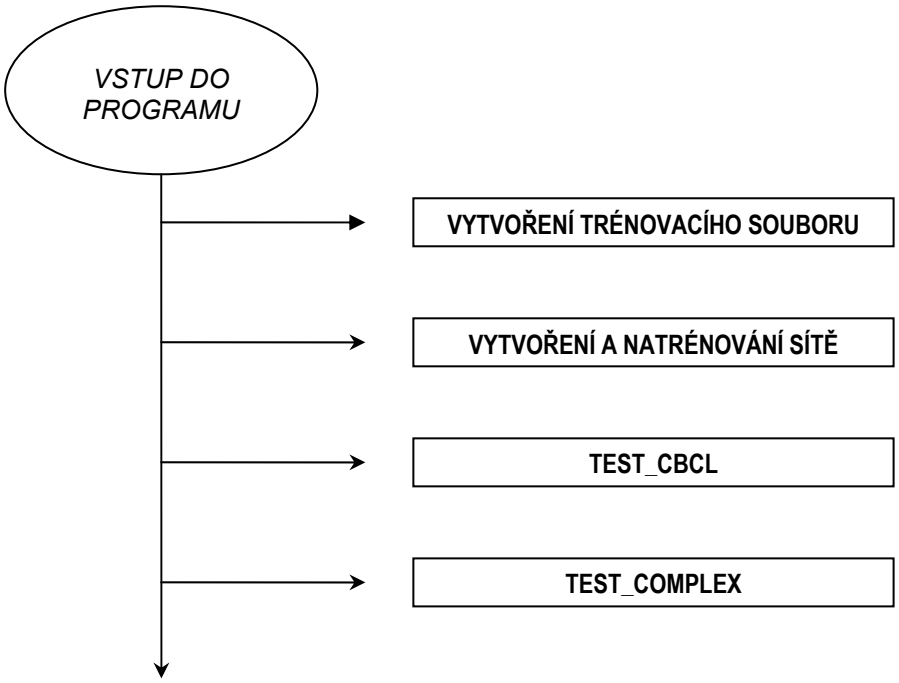


schéma 4.1 - znázornění celku složeného z podprogramů

4.3.1 Vytvoření trénovacího souboru

Tento podprogram je v podstatě velmi jednoduchý a má za úkol pouze vytvořit trénovací soubor pro síť. Jeho úkolem je načíst trénovací data, provést předzpracovaní kroky a data následně uložit do souboru, ze kterého budou čtena sítí při trénování.

Z hlediska uživatele není přístupný.

4.3.2 Vytvoření a natrénování sítě

Tento podprogram také není z hlediska uživatele přístupný, protože má za úkol trénování a vytváření sítě. Uživateli je předložena hotová síť pojmenovaná *ann.net*, kterou musí využívat.

Podprogram je schopen buď vytvořit standardní síť typu vícevrstvý perceptron, nebo síť s upravenou architekturou, jak už ale bylo řečeno, tato síť zatím není využita a bude jí věnována náležitá pozornost při pokračování v práci.

Dalším krokem je trénování sítě. K tomuto využívá podprogram standardní funkce, které poskytuje knihovna fann. Z funkcí lze zmínit například funkce na úpravu parametrů sítě, či samotné trénování.

4.3.3 TEST_CBCL

Toto je první podprogram přístupný uživateli. Je to podprogram, který je schopen projít zadaný adresář, který obsahuje obrázky v rozměrech 19x19.

Je napsán hlavně kvůli testování úspěšnosti sítě a nebude pravděpodobně moc využit, protože je mu potřeba zadat, jaká data jsou ve složce(obličejová / neobličejová). Pokud je mu zadána tato informace, je chopen spočítat procentuální úspěšnost sítě v detekování zadaných dat. Veškeré statistické informace nakonec vypíše na standardní výstup.

4.3.4 TEST_COMPLEX

Asi nejzajímavější podprogram z pohledu uživatele je *TEST_COMPLEX*. Je to podprogram, který je schopen projít zadanou složku libovolně velkých obrázků s tím, že otestuje veškeré obrázky ve složce a upravené obrázky(s vykreslenými detekcemi) uloží do druhé zadané složky.

4.4 Parametry

Parametry v programu slouží k ovlivnění jeho běhu. Jak už bylo řečeno, přístup je procedurální a jednotlivé podprogramy se spouští v průběhu funkce *main*. Parametry lze řídit tok programu a zároveň si vybírat, která část(nebo části) programu se má spustit v rámci jednoho běhu programu. Z hlediska uživatele jsou přístupné pouze funkce testovací. První z nich je nazvána *TEST_CBCL*(viz podkapitola 4.3.3), což je funkce, která umožňuje testování celého adresáře obsahujícího obrázky o rozměrech 19x19. Druhou funkcí je *TEST_COMPLEX*(viz podkapitola 4.3.4), která je schopná otestovat adresář s komplexními scénami. Podrobně bude popsáno použití parametrů v příloze A.

5 Výsledky

5.1 Testy

Síť byla testována na obou výše zmiňovaných databázích(viz podkapitoly 3.7.1 a 3.7.2), kde dosahovala úspěšnost rozdílných výsledků. Zároveň byla otestována i na několika komplexních obrázcích.

Součástí systému není žádný analytický nástroj, který by umožňoval automatické zpracování výsledků na jiných datech, než jsou testovací data databáze CBCL. V závěrečných fázích testování systému na komplexnějších příkladech se ukázalo, že takový nástroj by velice usnadnil práci a zároveň umožnil sběr mnohem většího množství dat a statistik při testování. Protože už nezbyl čas na implementaci tohoto nástroje(viz [6]), bude implementován až v další verzi systému, kde se předpokládá výrazné vylepšení výsledků právě díky sběru komplexnějších a ucelenějších statistik a to nejen při testování úspěšnosti, ale i při trénování sítě.

Tato kapitola shrnuje dosažené výsledky a diskutuje generalizační schopnosti sítě a zároveň neúspěchy sítě při falešných detekcích a nerozpoznaných obličejích. Výsledky jsou rozděleny do několika logických celků, které jsou obsahem následujících podkapitol.

5.1.1 Test na databázi CBCL

Tento test byl proveden jako první, aby se zjistila, základní úroveň natrénování sítě a byl tak umožněn další vývoj.

Metodika testování této databáze je trochu odlišná a je potřeba na to upozornit. Při testování neuronové sítě se totiž ukázalo, že nejmenší chybovosti síť dosahuje při automatickém nastavení prahu v aktivační funkci výstupního neuronu. Výstupní neuron potom obsahuje pouze dvě hodnoty a to buď 0(neobličej) nebo 1(obličej).

Jak už bylo řečeno výše, toto je jediný test, který je schopen akumulovat některé statistické údaje. Následuje ukázka úspěšnosti na testovacích datech.

	úspěšnost
obličejová data	72,4%
neobličejová data	84,9%

Pro úplnost je přiložena ukázka úspěšností na trénovacích datech.

	úspěšnost
obličejová data	100%
neobličejová data	82%

Jak je vidět na úspěšnosti rozpoznání trénovacích dat, je poměrně vysoká.

Procentuální úspěšnost při testování na testovacích datech sice není zcela uspokojivá, ale je potřeba také vzít v úvahu charakter testovaných dat. Pro názornost je přiloženo několik příkladů obličejových dat z testovací sekce databáze CBCL.

obr. 5.1 ukázka obličejových dat z testovací sekce databáze CBCL

Jak je vidět z příkladů, data jsou celkově menší kvality, obsahují obličeje pootočené o poměrně vysoký úhel, či obličeje téměř nerozeznatelné, takže dosažené výsledky se dají shrnout jako uspokojivé.

5.1.2 Test na databázi BioID

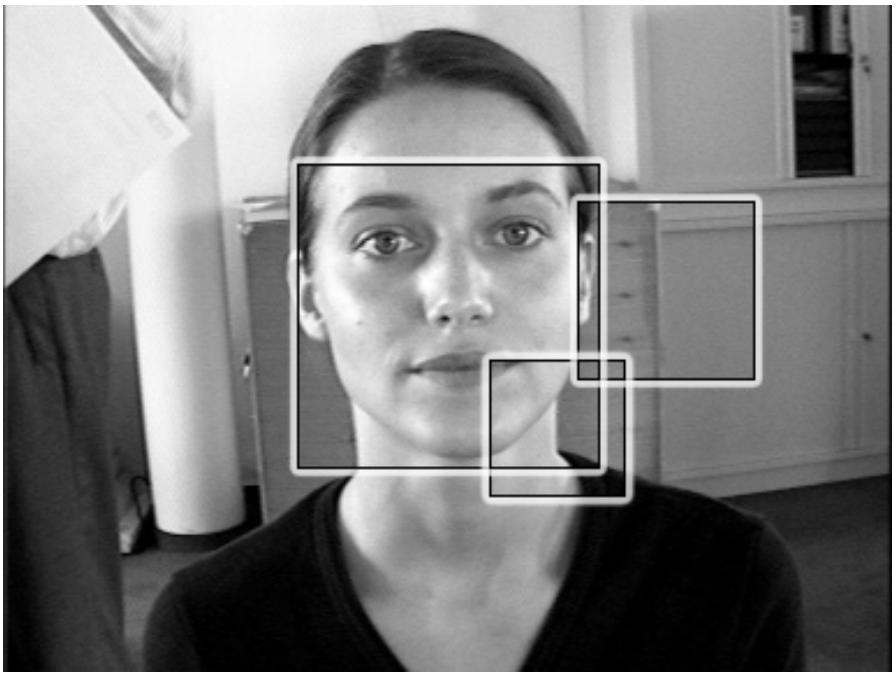
Testování na této databázi probíhalo až v závěrečné fázi projektu a podtrhuje dosažené výsledky, které shrnuje tato kapitola.

Při testování této databáze byl experimentálně stanoven práh, který určuje, které detekce jsou brány jako obličejové a které jsou brány jako neobličejové. Konečná hodnota prahu byla stanovena na 0,996.

Zpracování výsledků této databáze nelze zautomatizovat, protože databáze obsahuje komplexní příklady obrázků, kde nemůžeme hodnotu výsledku zjednodušit na pouhé ano, či ne. Z poměrně rozsáhlé databáze(viz podkapitola 3.7.2) bylo náhodně vybráno 100 příkladů zanalyzovaných obrázků. Na této množině byl potom proveden test úspěšnosti sítě při rozpoznávání obličejů. Následují statistické údaje vyjadřující úspěšnost sítě na databázových datech.

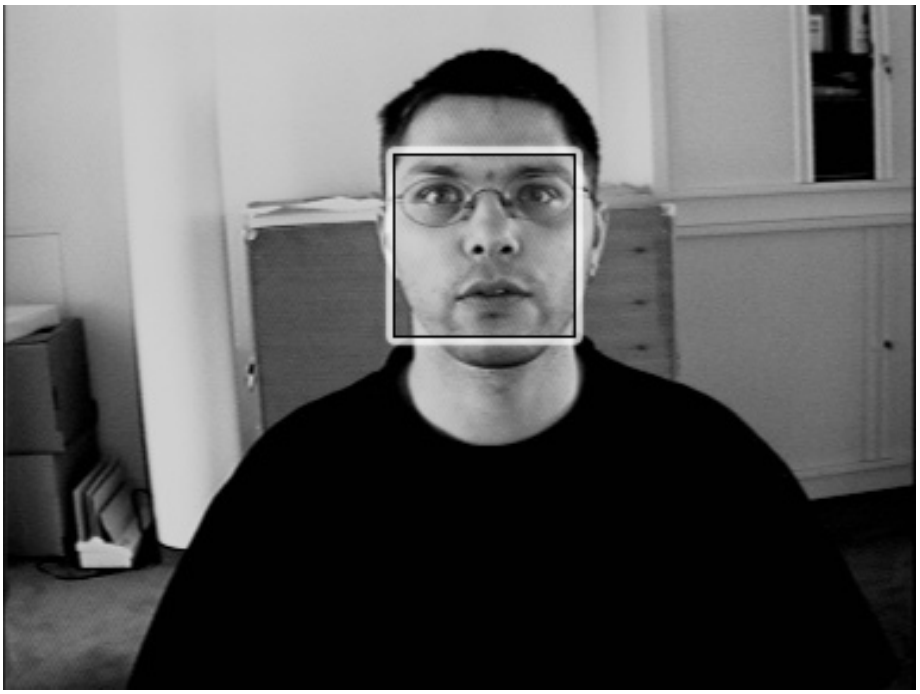
odhalené obličeje	falešné detekce
86%	79%

Co se týče detekování obličejů, systém dosahuje relativně vysoké úspěšnosti, ale jak je vidět z procentuálního ohodnocení falešných detekcí, systém jich detekuje velice mnoho. Následuje několik ukázek detekcí, pro lepší pochopení problému falešných detekcí.



obr. 5.2 ukázka falešných detekcí

obr. 5.3 ukázka nedetekovaného obličeje a jedné falešné detekce



obr. 5.4 ukázka správné detekce

Obrázek 5.2 je typickým příkladem analyzovaného obrázku s poměrně velkým počtem falešných detekcí. Zároveň je ovšem detekován obličej. Dalším příkladem obrázku s falešnou detekcí je obrázek 3.6(viz podkapitola 3.6).

Obrázek 5.3 je příkladem, kdy není obličej vůbec detekován. Zároveň ukazuje jeden s typických problémů se kterými se detektor potýká a to je označení spojitých ploch jako obličeje. Tento problém bude diskutován dále(viz podkapitola 5.1.4).

Posledním ukázkovým obrázkem z databáze BioID je obrázek 5.4. Tento obrázek ukazuje úspěšnou detekci.

5.1.3 Test skupinové fotky

Úplně na závěr je přiložen test skupinové fotky.

obr 5.5 ukázka testu skupinové fotky

Na fotce je 25 aktérů, z čehož systém detekoval 15, což je 59,9%. Falešných detekcí je zaznamenáno 25.

5.1.4 Diskuze úspěšnosti

Jak je vidět z výše uvedených hodnot, výsledky jsou zatíženy určitým procentem falešných detekcí, což je vidět už z náznaků předchozích testů. Detektor má zřetelný problém s oblastmi spojité barvy a s oblastmi, kde je v barvách zlomový přechod. Množství falešných detekcí, které jsou na spojitých plochách lze umírnit druhotným testem, který by z daného úseku obrázku spočítal některé statistické údaje. Tyto údaje by nám potom umožnily detekovat spojitou plochu, čímž bychom odhalili falešnou detekci. I toto může být jeden ze směrů budoucího vývoje.

Další možností je analýza propojenosti sítě. Síť lze totiž vytvořit pouze částečně propojenou a tato potom vykazuje jiné výsledky, než síť plně propojená. Toto je další z případů, který může zanášet chyby do výsledných detekcí.

Odhalení falešných detekcí na zlomových plochách je problém, který má základ v trénování sítě na příliš velkou přesnost. Tato síť má potom sice velice dobrou odezvu na trénovacích datech, ale v komplexnějších příkladech selhává. Kdybychom dokázali přesněji trefit moment nutný k ukončení trénování, tak by se nám s velkou pravděpodobností povedlo natrénovat síť, která by dosahovala při detekcích daleko větších úspěchů.

6 Závěr

Závěr shrnuje výsledky celé práce a zároveň je jejím ukončením. Naznačeny jsou i budoucí směry, kterými se dále práce bude ubírat.

Práce se zabývala lokalizací lidského obličeje ve statickém obraze a lze zkonstatovat, že implementovaný systém do jisté míry splňuje zadání, protože dokáže v zadaném obrázku nalézt určité procento obličejů, které obrázek obsahuje. Také při testování na testovacích datech z databáze CBCL dosahuje detektor poměrně dobrých výsledků. Zároveň je ale nutno dodat, že detekce, které systém provádí na komplexnějších scénách jsou zatíženy jistou chybovostí a je tudíž potřeba systém dále vyvíjet a pracovat na odstranění odhalených problémů, které s sebou onu chybovost přináší.

Co se týče dalšího vývoje, bylo stanoveno několik kroků, kterými bude práce na systému směřovat:

- **architektura sítě:** Vhodné upravení architektury sítě je jeden s hlavních kroků, na který bude kladen nemalý důraz kvůli potenciálnímu urychlení a zlepšení detekčních schopností systému.
- **trénovací data:** Budoucí modifikace systému by měly umožňovat dělit trénovací data na menší jednotky, díky kterým bude řízení trénování snazší a efektivnější.
- **trénování sítě:** Toto je další velice důležitý směr, kterým by se měla práce na systému dále ubírat. Bude nutné vytvořit sofistikovanější algoritmus trénování, který bude umožňovat zpětnou vazbu, což přímo souvisí s potřebou vytvoření analyzátoru statistik(viz podkapitola 5.1) produkovaných při trénování i testování sítě.
- **druhotný test:** Součástí budoucího systému by mohl být i druhotný test na obsah „rozpoznané“ oblasti. Tento test by nám pomohl odhalit oblasti spojité barvy, se kterými má stávající síť problémy. Nutno ovšem dodat, že pokud by se podařilo natrénovat síť na uspokojivou úspěšnost, daný test by pravděpodobně nebyl potřeba.

Z uvedených kroků je zřejmé, že práce na projektu bude pokračovat dále aby bylo dosaženo větší procentuální úspěšnosti a aby výsledný systém dosahoval co možná nejlepších výsledků.

Literatura

[1] *Neural Networks Introduction*. [online] [cit.18.02.2007]
URL <http://uhavax.hartford.edu/compsci/neural-networks-tutorial.html>

[2] DRÁBEK O., SEIDL P., TAUFER I. Umělé neuronové sítě – Základy teorie a aplikace(3). *CHEMagazín*, 2006, roč. 16, č.1, s. 12-14.

[3] YANG M.-H.: *Recent Advances in Face Detection*. California: Honda Research Institute, 2004, 93 p. IEEE ICPR Tutorial

[4] ROWLEY A. H. *Neural Network-Based Face Detection* ,CMU-CS-99-117, May 1999, 149 p., thesis on Computer Science Department at Carnegie Mellon University

[5] *Bicubic interpolation - Wikipedia, the free encyclopedia*. [online] [last rev. 05.05.2007] [cit. 06.05.2007]

[6] ŠVUB M. *Ročníkový projekt*, Brno, 2005, 33 s., Ročníkový projekt na Fakultě informačních technologií na Vysokém učení technickém

Příloha A

Tato příloha obsahuje seznam parametrů, které podporuje program vypracovaný v rámci tohoto projektu. Podporovány jsou dva módy testování. První otestuje adresář obrázků velikosti 19x19 a druhý otestuje adresář obrázku libovolné velikosti.

test_cbcl- `-cbcl cesta typ_dat`

-**cbcl** - přepínač
-**cesta** - vyjadřuje cestu k adresáři s obrázky
-**typ_dat** - 0- neobličeje, 1- obličeje

-všechny parametry jsou povinné

-**příklad použití** `-cbcl test\cbcl_faces\ 1`
`-cbcl c:\testovani\ 0`

test_complex- `-complex cesta_test cesta_vysl`

-**complex** -přepínač
-**cesta_test** -vyjadřuje cestu k adresáři s testovanými obrázky
-**cesta_vysl** -vyjadřuje cestu k adresáři, kde budou uloženy výsledné obrázky

-všechny parametry jsou povinné

-**příklad použití** `-complex test\ test_vysledek\`

jiné parametry program nepřijímá