UNIVERZITA HRADEC KRÁLOVÉ

FAKULTA INFORMATIKY A MANAGEMENTU

KATEDRA INFORMATIKY A KVANTITATIVNÍCH METOD

DISERTAČNÍ PRÁCE

Detekce objektů ve videosekvenci

Autor: Ing. Karel Petránek

Školitel: prof. RNDr. Eva Milková, Ph.D.

Konzultant: Mgr. Jan Vaněk, Ph.D.

Hradec Králové říjen 2017

Prohlášení:

Prohlašuji, že jsem disertační práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 4. 10. 2017 Karel Petránek

Poděkování:

Rád bych poděkoval vedoucí práce prof. RNDr. Evě Milkové, Ph.D. a konzultantovi Mgr. Janu Vaňkovi, Ph.D. za podnětné rady, pomoc s publikační činností a spolupráci na výzkumu.

Anotace

Disertační práce se zabývá oblastí detekce objektů v obraze a videosekvenci, která je spojením dílčích oblastí zpracování obrazu a strojového učení. Práce shrnuje základní algoritmy používané v obou oblastech, které vedou k současným state-of-the-art detekčním metodám. Podrobně jsou diskutovány hluboké neuronové sítě, které v oblasti detekce objektů v současnosti dosahují nejlepších výsledků.

Základními stavebními prvky systému pro detekci objektů v obraze jsou kvalitní datová sada, vhodné předzpracování a přesný klasifikátor. Práce přináší výsledky ve všech těchto oblastech – je vytvořena vlastní datová sada, na které jsou otestovány existující metody, navržen detektor umístění objektu ve videosekvenci založený na kombinaci hlubokých neuronových sítí a částicových filtrů, vytvořena neuronová síť pro segmentaci objektu v obraze a navrženo rozšíření operátoru Local Binary Patterns.

Navržený detektor umístění objektu ve videosekvenci překonává 5 existujících detektorů a dosahuje dostatečné přesnosti pro praktické nasazení. Vytvořená segmentační síť vytváří masku přítomných objektů ve vstupním obraze v reálném čase při použití akcelerace pomocí grafické karty při zachování dobré generalizační schopnosti i na neznámé objekty.

Klíčová slova: analýza obrazu, detekce objektů, segmentace obrazu, Local Binary Patterns

Abstract

The doctoral thesis “Object Detection in Videos” deals with the area of object detection in images and videos. The subject area consists of two main sub-areas – image processing and machine learning. The thesis summarizes common algorithms used in image processing and machine learning that lead to the state-of-the art object detection methods. Deep neural networks are the main focus of the research, as they are currently the best performing object detection method.

A high quality dataset, preprocessing and a precise classifier are the basic building blocks of an object detection system. The thesis achieves results in all these areas – a custom dataset is forged which is used to test existing methods, an object detector for videos is designed based on the synergy of deep neural networks and particle filters, a neural network is trained for object segmentation and finally, an extension for Local Binary Patterns is presented.

The proposed video object detector surpasses the performance of 5 existing detectors and its precision is sufficient for real-world usage. The segmentation network creates an object mask for the given input image in real time when GPU acceleration is used while generalizing well to new object types.

Keywords: image analysis, object detection, image segmentation, Local Binary Patterns

Obsah

Seznam zkratek

BN	Batch Normalization
CGAN	Conditional Generative Adversarial Networks
GAN	Generative Adversarial Networks
GLOH	Gradient Location and Orientation Histogram
GPU	Graphics Processing Unit, grafická karta
LBP	Local Binary Patterns
MS COCO	Microsoft Common Objects in Context
ReLU	Rectified Linear Unit
RGB	Red, Green, Blue (barevné spektrum)
SGD	Stochastic Gradient Descent, stochastická metoda největšího spádu
SIFT	Scale-Invariant Feature Transform
SURF	Speeded-Up Robust Features

1 Úvod

Detekce objektů v obraze patří k nejčastějším úlohám počítačového vidění především proto, že do této kategorie spadá i detekce lidských tváří. Pro člověka téměř nevědomý proces analýzy světa kolem sebe a rozpoznávání objektů v zorném poli je pro počítač i přes rostoucí výkon stále velmi náročnou úlohou [Beran et al., 2011].

Největším problémem při rozpoznávání objektu jsou zkreslení, která vznikají průmětem trojrozměrné podoby objektu na senzor snímacího zařízení. Robustní algoritmus pro rozpoznávání obrazu se musí vyrovnat s měnící se pozicí, natočením, měřítkem, perspektivním zkreslením, různými úrovněmi jasu a kontrastu, změnou barevnosti, šumem snímače, rozostřením a s mnoha dalšími strukturálními či náhodnými zkresleními.

Pro vytvoření robustního řešení je proto potřeba rozsáhlých datových sad, které zachycují co nejvíce možných zkreslení a podob objektu. Cílem je, aby výsledný rozpoznávací algoritmus našel reprezentaci objektu nezávislou na uvedených zkresleních a byl schopen o daném obrazovém vstupu rozhodnout, zda odpovídá dané reprezentaci či nikoliv. Vytvoření takové reprezentace je datově a výpočetně náročný proces, ve kterém je nutné několikrát analyzovat všechna data v trénovací sadě.

Všechny moderní systémy se maximálně snaží využít výpočetní prostředky současných počítačů. Při návrhu algoritmu je kladen důraz především na snadnou paralelizaci řešení, která usnadňuje implementaci na programovatelných grafických kartách (GPU) a počítačových clusterech.

Přirozeným rozšířením detekce objektů ve statickém obraze je detekce a sledování objektů ve videosekvenci. Základní přístup spočívá v analýze každého snímku samostatně algoritmem pro detekci objektu ve statickém obraze. Tento přístup však nevyužívá dodatečné informace obsažené ve videu, jako například vzájemnou podobnost po sobě jdoucích snímků, které lze využít pro zpřesnění detekce a úsporu výpočetních prostředků.

V práci je nejprve analyzována teoretická architektura algoritmů pro detekci objektů v obraze a je porovnána s existujícími přístupy. Dále jsou představeny existující techniky pro práci s videosekvencemi a možnosti jejich kombinace s algoritmy pro detekci objektů v obraze. Na základě provedené analýzy je prezentován vlastní výzkum v oblasti.

2 Definice cílů disertační práce

Hlavním cílem disertační práce je navrhnout systém či systémy pro detekci objektů ve videosekvenci, které budou mít dostatečnou přesnost na to, aby výrazně usnadnily pro člověka únavné úlohy či úplně nahradily lidský dozor. Práce se proto bude soustředit na otestování existujících přístupů na modelovém projektu, který bude vycházet z praktického problému, a na návrh vlastního řešení, budou-li stávající řešení nedostatečná.

Cíl disertační práce je soustředěn do čtyř dílčích, navzájem provázaných oblastí:

1) **Testování stávajících state-of-the-art metod na nestandardní datové sadě**
   Většina algoritmů pro rozpoznání objektů v obraze a ve videosekvenci je optimalizována na několik standardních datových sad, jako je ImageNet [Russakovsky et al., 2014a], CIFAR [Krizhevsky et al., 2009], PASCAL [Everingham et al., 2010] či Microsoft COCO [Lin et al., 2014]. Dílčím cílem práce je otestování existujících metod na vlastní datové sadě, především na detekci objektu za obtížných zkreslení a při okluzi.
2) **Analýza a zlepšení extrakce atributů**
   Základem každého systému pro rozpoznání obrazu je vhodný výběr atributů a jejich extrakce ze vstupního obrazu. Cílem práce je navrhnout rozšíření či vylepšení stávajících metod extrakce atributů a tato rozšíření porovnat s výchozím stavem. Základní zaměření bude na odolnost atributů vůči posunutí, rotaci a změně měřítka, protože tato zkreslení jsou běžná při sledování objektu ve videosekvenci.
3) **Zlepšení detekce umístění objektu v obraze**
   Pro sledování objektu ve videosekvenci je nutné přesně zaznamenat i pozici objektu, práce se proto zaměří na zlepšení lokalizace objektu v rámci jednoho snímku.
4) **Propojení detekce objektu s jeho sledováním ve videosekvenci**
   Stávající metody pro sledování objektů ve videosekvenci vyžadují uživatelem vybrat objekt, který se má sledovat. Oproti tomu detekční algoritmy aplikované na každý snímek nevyužívají vzájemné korelace po sobě jdoucích snímků videosekvence a spojitost pohybu objektu v čase. Cílem dalšího výzkumu je proto propojit tyto dva přístupy tak, aby detekce i sledování mohly probíhat bez lidského zásahu.

3 Analýza současného stavu v oblasti detekce objektů

Snímek z kamer či fotoaparátů operujících ve viditelném spektru je v počítači reprezentován jako obraz, tj. dvourozměrná mřížka vektorů, které obsahují barevnou informaci. Cílem systému pro rozpoznání objektu v obraze je zjistit, které (pokud nějaké) buňky v této mřížce odpovídají hledanému objektu, případně nalézt střed či ohraničující obdélník hledaného objektu.

Aby bylo možné objekt robustním způsobem rozpoznávat, je nutné vytvořit jeho počítačový model. Model by měl splňovat následující podmínky [Tarr et al., 2002]:

- Unikátnost
    - Daný model odpovídá pouze jednomu objektu, případně úzce vymezené množině objektů, jiné objekty mají odlišnou reprezentaci.
- Opakovatelnost
    - Pomocí modelu je možné rozpoznat stejný objekt transformovaný různými afinními transformacemi – rotacemi, posunutími, změnami měřítka a především při perspektivním zkreslení.
- Robustnost
    - Model je odolný proti náhodným změnám v podobě objektu, jako je barevný šum kamer či změna osvětlení scény.

Při praktické implementaci jsou často určujícími podmínkami i paměťová náročnost reprezentace modelu a výpočetní složitost jeho vybudování ze vstupního snímku.

Historicky probíhal vývoj od nejjednodušších typů objektů, jako jsou čáry či kruhy na obrazech s nízkým šumem a zkreslením. Již v roce 1962 byla představena po autorovi dnes pojmenovaná Houghova transformace, která na základě bodů v obraze umožňuje určit parametry přímek, které tyto body tvoří [Hough, 1962]. Později byl algoritmus rozšířen i na složitější objekty, ale nutnost parametrického vyjádření objektu, exponenciální paměťová složitost vzhledem k počtu parametrů objektu a nízká odolnost vůči šumu neumožňuje Houghovu transformaci použít pro detekci složitých objektů, jako jsou například lidské obličeje.

Další základní metodou je použití šablony objektu a její přikládání na analyzovaný obraz [Lewis, 1995]. Místa analyzovaného obrazu, která jsou dostatečně podobná šabloně, jsou označena jako objekt. Ačkoliv je možné vyjádřit složitější objekty než v případě Houghovy transformace, je složité dosáhnout robustnosti vůči rotaci, změně velikosti, jasu apod. Částečným řešením je porovnávat transformace výřezů a šablony. Často používanou transformací je histogram, protože je robustnější vůči šumu, neuchovává informaci o rotaci a lze jej porovnávat nezávisle na jasu. Tyto

výhody jsou však kompenzovány vyšší výpočetní náročností a ztrátou informace způsobenou převedením na histogram.

Robustnějším řešením než porovnávání pomocí šablon je rozklad (segmentace) obrazu na regiony se stejnou barvou či texturou a následná analýza těchto regionů. Výhodou tohoto přístupu je odolnost vůči náhodnému šumu v obraze. Normalizací regionů lze dosáhnout invariance vůči rotaci a změně měřítka či rozlišení. Problémem algoritmů založených na segmentaci je především překryv objektů jinými objekty ve scéně, který vede ke změně tvaru, rozdělení či úplnému skrytí některých částí objektu. Důležitým aspektem je také odolnost výsledné segmentace vůči zkreslením, přičemž algoritmy produkující stabilní segmentace bývají výpočetně náročné a obtížně paralelizovatelné [Petránek et al., 2011]. Mezi běžně používané segmentační algoritmy patří K-Means [Hartigan et al., 1979], MeanShift [Comaniciu et al., 2002] a hledání minimálního řezu pixelovou mřížkou [Shi et al., 2000].

Některé části obrazu jsou pro detekci objektů důležitější než jiné – například hrany či rohy obvykle přinášejí více informací o objektu než jednobarevné plochy. Tohoto faktu využívají algoritmy založené na detekci významných oblastí a bodů v obraze, jejichž kombinací modelují výslednou podobu objektu. Vhodnou volbou kritérií pro výběr významných oblastí lze docílit invariance vůči většině běžných zkreslení – otočení, posunutí, změna měřítka, perspektivní zkreslení a zkreslení jasu a kontrastu. Mezi algoritmy, které detekují významné oblasti s těmito charakteristikami, patří SIFT, SURF, GLOH, LBP, které jsou zmíněny podrobněji v následujících kapitolách. Výstupy z těchto algoritmů lze poté kombinovat do popisu objektů např. pomocí techniky Bag of Visual Words [Leung et al., 2001]. Nevýhoda detektorů významných oblastí spočívá v jejich přílišné náročnosti na kritéria – ne všechny objekty obsahují dostatek významných bodů či oblastí pro úspěšnou detekci.

Současný přístup k detekci objektů se snaží o zjednodušení celého procesu a odstranění ručně navrhovaných částí algoritmu, jako je například detekce významných bodů. Cílem je objekty rozpoznávat přímo na bitmapovém obraze a odstranit tak předzpracování, které obvykle přináší množství parametrů, které je potřeba stanovit. Nejlepší úspěšnost detekce mají v současnosti umělé neuronové sítě, které drží rekordy ve všech tradičně používaných srovnávacích testech [Russakovsky et al., 2014a; Krizhevsky et al., 2009; Everingham et al., 2010].

V následujících kapitolách je nejprve analyzována obecná architektura algoritmů pro detekci objektů, a následně jsou popsány její jednotlivé součásti – předzpracování, extrakce atributů a klasifikace. Zvláštní kapitola je věnována umělým neuronovým sítím, které lze použít ve všech částech detekční architektury. V předposlední podkapitole jsou rozebrány metody pro analýzu videosekvencí a jejich propojení s detekčními algoritmy. Nakonec jsou diskutovány běžně používané datové sady.

3.1 Obecná architektura detekčních algoritmů

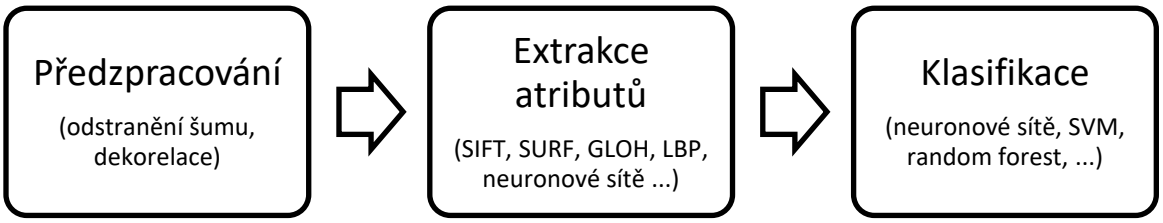
Algoritmus, který detekuje přítomnost objektu v obraze lze vyjádřit jako funkci nad vstupním obrazem:

$$f(I) = \begin{cases} 0, & objekt\ není\ přítomen \\ 1, & objekt\ je\ přítomen \end{cases}$$

kde I je vstupní obraz a $f(I)$ detekční algoritmus.

V případě detekce n objektů je výstupem n-rozměrný binární vektor. Uvedenou definici je dále možné rozšířit na pravděpodobnostní verzi, kdy výstupem algoritmu je reálné číslo v intervalu $\langle 0, 1 \rangle$. Je-li detekováno více vzájemně se vylučujících objektů, pak musí platit, že součet všech hodnot výstupního vektoru je roven 1, aby jej bylo možné interpretovat jako hustotu pravděpodobnosti.

Vstupní obraz ve formě dvourozměrné mřížky s RGB hodnotami není příliš vhodný pro přímou detekci objektů. První částí detekčního algoritmu je proto předzpracování obrazu a extrakce důležitých atributů analyzovaného obrazu. Tyto atributy jsou použity jako vstup pro klasifikační algoritmus, který rozhoduje o vstupním vektoru atributů, zda odpovídají hledanému objektu. Obrázek 1 zobrazuje diagram architektury obecného algoritmu pro detekci objektů.



Obrázek 1: Obecná architektura algoritmu pro detekci objektů

V následujících kapitolách jsou podrobně diskutovány jednotlivé algoritmy používané ve výše zmíněných fázích.

3.2 Algoritmy předzpracování obrazu

3.2.1 Redukce šumu

Nejčastěji se vyskytujícím šumem při snímání scény kamerou je barevný šum, který je způsoben CCD či CMOS snímači běžně používanými v kamerách a fotoaparátech. Tento efekt je často umocněn následnou kompresí pomocí JPEG či MPEG. Pro odstranění tohoto typu šumu je možné použít Gaussovské vyhlazení, což je konvoluce obrazu z kamery diskretizovaným Gaussovským konvolučním jádrem, například pro čtvercové okolí 5×5:

$$G = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

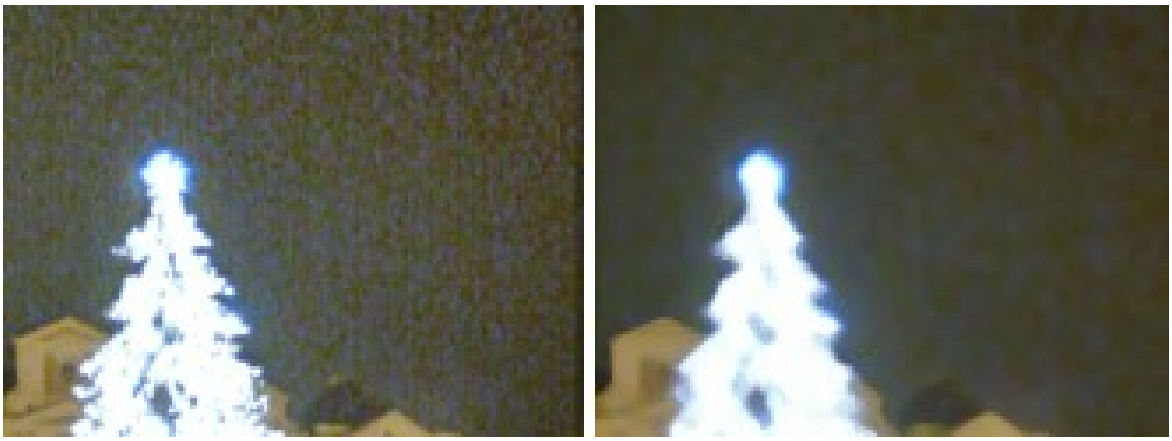
$$\mathrm{p}(x, y) = G * \mathrm{N}(x, y)$$

kde x a y jsou souřadnice obrazového bodu, $\mathrm{p}(x, y)$ je barva bodu na souřadnicích x, y a $\mathrm{N}(x, y)$ je okolí bodu $\mathrm{p}(x, y)$. Operátor * značí konvoluci.

Výsledkem vyhlazení je rozostřený obraz, ve kterém původní vysokofrekvenční šum kamery zaniká. Nevýhodou tohoto způsobu redukce šumu je, že je současně redukována i vysokofrekvenční informace, což vede především k redukci ostrosti obrazu na hranách objektů (viz Obrázek 2).

Obrázek 2: Eliminace šumu pomocí Gaussovského vyhlazení. Je patrné vyhlazení CCD šumu především v oblasti oblohy, dochází však k rozostření hran.

Alternativou k redukci šumu pomocí Gaussovského vyhlazení je použití mediánového filtru. Mediánový filtr nahrazuje bod obrazu mediánem z bodů v jeho okolí. Pro odstranění lokálního barevného šumu postačuje čtvercové okolí o velikosti 3×3 či 5×5 bodů. Výhodou tohoto filtru oproti Gaussovskému vyhlazení je zachování významných hran v obraze a odstranění pouze drobných lokálních výkyvů v barevné informaci, které obvykle odpovídají šumu (Obrázek 3). Nevýhodou je větší výpočetní náročnost při větších okolích, protože pro nalezení mediánu je třeba body v okolí nejprve setřídit či použít rekurzivní algoritmus Quickselect [Hoare, 1961].



Obrázek 3: Vyhlazení obrazu pomocí mediánového filtru. Dochází k odstranění šumu v oblasti oblohy při zachování důležitých hran.

3.2.2 Dekorelace

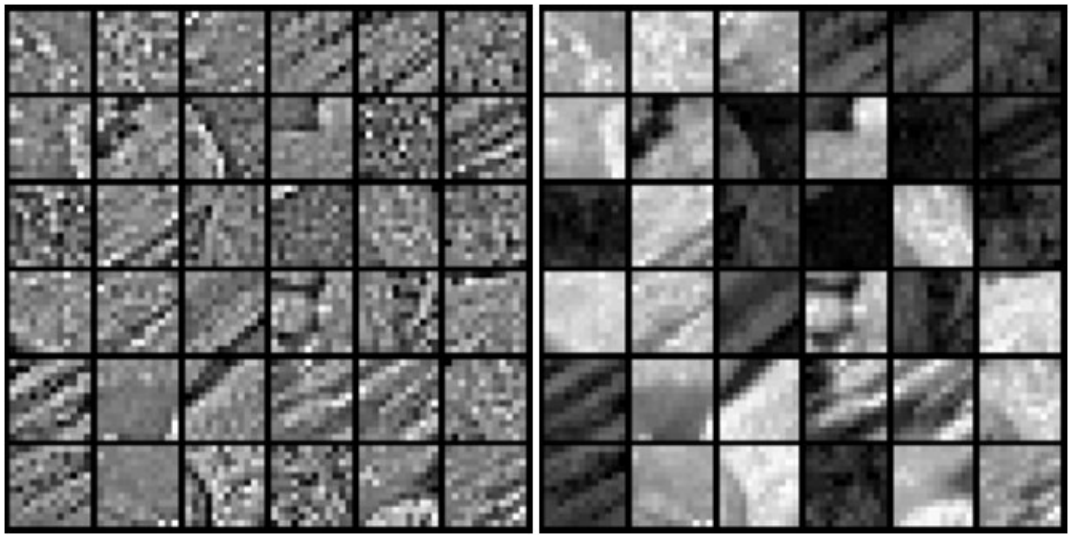
Na obraz lze nahlížet jako na pozorování náhodného jevu popsané dvourozměrnou mřížkou intenzit, kde každá intenzita představuje konkrétní hodnotu jedné náhodné proměnné. Protože obrazy zachycují strukturu scény, sousedící intenzity jsou obvykle mezi sebou silně korelované. Silná korelace může způsobit pomalejší výpočet některých atributů nebo způsobit, že algoritmy pro výpočet atributů zanedbají důležitou informaci obsaženou v obraze.

Dekorelace pracuje s malými výřezy vstupního obrazu (např. 16×16 bodů) a snaží se minimalizovat korelaci mezi jednotlivými složkami vzniklých 256rozměrných vektorů intenzit [Ng et al., 2015b]. Výřezy lze náhodně vybrat z trénovací sady dat, jejich množství závisí na velikosti trénovací sady – mělo by se jednat o reprezentativní vzorek všech možných výřezů. Korelace mezi složkami vzniklých vektorů je minimální, má-li kovarianční matice nulové hodnoty všude kromě diagonály. Toho lze dosáhnout výpočtem vlastních vektorů kovarianční matice a vyjádřením výřezů v soustavě souřadnic dané těmito vlastními vektory.

Algoritmy pro tvorbu atributů a klasifikační algoritmy obvykle lépe pracují se vstupními daty, která mají nulovou střední hodnotu a jednotkový rozptyl. Před výpočtem kovarianční matice

a vlastních vektorů je proto vhodné odečíst od všech výřezů odhad střední hodnoty, obvykle průměr či medián. Jednotkového rozptylu lze dosáhnout vydělením jednotlivých složek vektorů odpovídajícími vlastními čísly kovarianční matice. V případě nulových či skoro nulových vlastních čísel je možné redukovat počet složek vektoru nebo přidat k vlastnímu číslu malou regularizační konstantu (např. 10^{-4}).

Nevýhodou dekorelace je zdůrazňování šumu, který může způsobovat snížení přesnosti u navazujících algoritmů, viz Obrázek 4. Je proto vhodné ze vstupních výřezů předem odstranit šum pomocí technik zmíněných v předchozí kapitole.



Obrázek 4: Ukázka dekorelovaných výřezů se zvýrazněným šumem (vlevo). Pravá část ukazuje výřezy před dekorelací. Obrázek převzat z [Ng et al., 2015b].

3.3 Algoritmy pro extrakci atributů

Při vytváření modelu objektu jsou atributy modelu stanovovány na základě pozorování vstupních dat, buď manuálně, nebo automaticky. Z podmínek modelu uvedených na začátku kapitoly 3.1 (unikátnost, opakovatelnost, robustnost) je možné odvodit podmínky pro atributy modelu:

- Nezávislost
  - Atributy by měly být vzájemně nekorelované a nezávislé náhodné veličiny.
- Popisnost
  - Atributy vyjadřují význačné vlastnosti daného objektu. Význačné vlastnosti jsou takové, které objekt s dostatečnou spolehlivostí odliší od ostatních objektů a od pozadí scény.
- Vyčíslitelnost
  - Hodnoty atributů jsou vektory či skaláry diskrétních nebo spojitých veličin.

Nezávislost zajišťuje, že jednotlivé atributy nebudou rozlišovat stejné či velmi podobné vlastnosti. Cílem při identifikaci objektů je nalézt co nejvíce atributů, jež se vzájemně co nejlépe doplňují. Atributy, které jsou triviální kombinací jiných atributů, nepřinášejí do rozhodování žádné dodatečné informace. Jejich zahrnutí může snížit robustnost a efektivitu detekce. Naopak, použití nekorelovaných atributů zvyšuje robustnost, protože pravděpodobnost, že náhodný šum v datech ovlivní nekorelované atributy stejným způsobem, s počtem atributů klesá.

Popisnost je založena na extrakci významných charakteristik objektu, které jsou invariantní vůči běžně pozorovatelným deformacím a zkreslením objektu. Splnění této podmínky zajišťuje opakovatelnost detekce objektu i při změněných podmínkách pozorování.

Vyčíslitelnost poskytuje možnost hodnoty atributů zpracovat pomocí matematických a statistických metod. Tato vlastnost je důležitá pro splnění všech tří podmínek – unikátnosti, robustnosti i opakovatelnosti – které jinak není možné algoritmicky zajistit.

Hierarchičnost atributů je vlastností, která není nutná pro splnění podmínek modelu, umožňuje však snazší vytváření a kompozici atributů. Cílem je atributy vytvářet postupně kombinováním od základní úrovně – barvy a intenzity přes hrany, rohy až po složité atributy, které odpovídají např. lidskému obličeji. Atributy jsou vždy vyjádřeny pomocí atributů úrovně o jedna nižší, což omezuje prostor hledání.

Mezi základní atributy objektu patří barva, kterou je možné transformovat a kombinovat do složitějších atributů. Barva je vyjádřena jako vektor v prostoru barev, který je určen barevným modelem. Transformací barvy do odpovídajících barevných modelů je možné výpočetně jednoduše

vytvořit kvalitní atributy. Obohacením snímků z kamer dodatečnou prostorovou informací je možné vytvořit další atributy, které umožňují zachytit i prostorovou podobu objektu [Petránek et al., 2010].

Následující podkapitoly uvádějí přehled nejčastěji používaných atributů a algoritmů pro jejich extrakci.

3.3.1 Gradient

Gradient obrázku se používá jako základní atribut v mnoha existujících metodách pro detekci objektů ([Kalal et al., 2010b; Bradski et al., 2008; Lowe, 1999]) a význačných bodů (např. SIFT, SURF či GLOH, viz následující podkapitoly). Lze jej vypočítat jako rozdíl mezi sousedními body jednotlivých barevných kanálů v obraze. Nejčastěji se používá centrální diference pro horizontální a vertikální směr v podobě Sobelova operátoru [Jain et al., 1995]:

$$\mathrm{G_x}(x,y) = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * \mathrm{N}(x,y)$$

$$\mathrm{G_y}(x,y) = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * \mathrm{N}(x,y)$$

$$\mathrm{p}(x,y) = \sqrt{\mathrm{G_x^2}(x,y) + \mathrm{G_y^2}(x,y)}$$

kde $\mathrm{G_x}(x,y)$ je hodnota gradientu ve svislém směru, $\mathrm{G_y}(x,y)$ hodnota gradientu ve vodorovném směru, $\mathrm{N}(x,y)$ je 3×3 okolí bodu, který má v obraze souřadnice $[x,y]$ a $\mathrm{p}(x,y)$ je velikost gradientu v bodě $[x,y]$. Symbol * značí konvoluci.

Hodnoty velikosti gradientu jsou vyšší v místech, kde se v obraze nacházejí hrany či rohy. Tato místa často představují významné části obrazu, protože zde dochází k významné změně barvy či intenzity.

Výhodou gradientu je jeho jednoduchost, výpočetní nenáročnost a invariance vůči konstantním změnám v jasu ve scéně. Při výpočtu gradientu je též možné zachytit jeho orientaci, která poskytuje další atribut, který je možné využít při detekci objektů:

$$\Theta(\mathrm{x},\mathrm{y}) = \frac{\mathrm{G_y}(x,y)}{\mathrm{G_x}(x,y)}$$

kde $\Theta(\mathrm{x},\mathrm{y})$ je tangens úhlu gradientu a $\mathrm{G_x}(x,y)$a $\mathrm{G_y}(x,y)$ hodnoty gradientu ve vertikálním, resp. horizontálním směru.

Výhodou gradientu je možnost jeho skládání do komplexnějších atributů, protože výstupní body jsou ve stejném barevném prostoru jako body vstupního obraz. Výpočet gradientu lze snadno paralelizovat a převést tak jeho výpočet na grafickou kartu.

3.3.2 Scale Invariant Feature Transform (SIFT)

Algoritmus SIFT [Lowe, 1999] extrahuje ze vstupního snímku množinu význačných. Význačný bod je takový, jenž lze jednoznačně odlišit od ostatních detekovaných bodů i v případě, že je scéna zabírána z jiného úhlu. Vlastnosti jednotlivých bodů jsou invariantní vůči rotaci, posunutí, změně velikosti a dalším afinním transformacím (především perspektivnímu zkreslení) a částečně i změnám světelných podmínek. Algoritmus SIFT lze použít pro detekci objektů, které obsahují dostatek význačných bodů, a pro sledování pohybu význačných bodů ve videosekvenci.

Principem algoritmu je výpočet histogramu z gradientu v okolí význačných obrazových bodů extrahovaných analýzou prostoru měřítek [Lindeberg, 1993]. V histogramu jsou zachyceny orientace a intenzity gradientů. Díky využití histogramů a zachycení orientace je zajištěna invariance vůči posunutí a rotaci. Invariance vůči změně velikosti a afinním transformacím je dosažena výpočtem histogramů nad obrazovou pyramidou [Lowe, 1999], kde každá vrstva pyramidy je 2× zmenšenou verzí předcházející vrstvy. Částečné invariance vůči změnám osvětlení je dosaženo prahováním intenzit gradientů a zahozením gradientů nižších než 10 % maximální hodnoty intenzity gradientu. Některé části tohoto algoritmu, především obrazová pyramida a výpočty gradientů, lze snadno paralelizovat a urychlit tak výpočet pomocí grafické karty.

Uvedené vlastnosti SIFT splňují podmínky stanovené v kapitole 3.3 a díky těmto vlastnostem je SIFT využíván v oblasti počítačového vidění, především pro zjišťování pohybu či hledání podobností mezi obrazy [Se et al., 2001].

Nevýhodou tohoto algoritmu je jeho vysoká výpočetní náročnost, která znesnadňuje získávání význačných bodů v reálném čase. Původní algoritmus SIFT je v USA patentován [Lowe, 2004], což ztěžuje jeho použití v praxi. SIFT zároveň netvoří hierarchickou strukturu, neboť nelze aplikovat stejný postup na obraz již transformovaný pomocí SIFT, což znesnadňuje kombinování do složitějších atributů.

3.3.3 Speeded-up Robust Features (SURF)

SURF je algoritmus vyvinutý jako reakce na SIFT [Bay et al., 2006]. Výstupem algoritmu jsou, podobně jako v případě SIFT, význačné body v obraze. SURF atributy jsou invariantní vůči rotaci, posunutí, změně velikosti a dalším afinním transformacím. Oproti algoritmu SIFT je rychlejší na výpočet, což usnadňuje jeho použití v interaktivních aplikacích.

Algoritmus je založen na aproximaci lokálního gradientu pomocí kombinace součtů čtvercových ploch v okolí význačných obrazových bodů detekovaných podobným způsobem jako v případě SIFT. Invariance vůči rotaci je zajištěna sledováním orientace gradientů. Invariance vůči změně velikosti je zajištěna výpočtem gradientu z různě velkých lokálních okolí. Při použití integrálních obrazů je možné vypočítat odhady gradientů v konstantním čase pro libovolně velká okolí, což zajišťuje výpočetní efektivitu algoritmu. Díky datové koherenci a nezávislosti je možné části algoritmu paralelizovat a spustit na grafické kartě.

Stejně jako v případě SIFT netvoří SURF atributy hierarchickou strukturu, což ztěžuje vytváření složitějších reprezentací. Algoritmus SURF je stejně jako SIFT v USA patentován [Funayama et al., 2009].

3.3.4 Census transformace

Census [Zabih et al., 1994] je transformace obrazu, která zachycuje jeho lokální strukturální vlastnosti. Pro každý bod obrazu je vypočítán strukturální vzor, který je odvozen z intenzit bodů ve čtvercovém lokálním okolí. Výsledkem Census transformace je řetězec bitů pro každý bod čtvercového či obdélníkového lokálního okolí, který zachycuje rozložení intenzit. Tento řetězec je sestaven podle následujícího předpisu:

$$x_i = \begin{cases} 0, & i < c \\ 1, & i \geq c \end{cases} \; pro\ všechna\ i\ v\ lokálním\ okolí\ c$$

kde x_i je výstupní bit řetězce a c je bod, pro jehož okolí se transformace počítá.

Nejčastěji se používají census transformace o velikosti okolí 4×4, 8×4 a 8×8, protože počty bitů ve výsledném řetězci odpovídají velikostem celočíselných registrů (16, 32 a 64 bitů), lze s nimi tedy efektivně manipulovat a ukládat je.

Porovnání bodů transformovaných pomocí Census transformace je provedeno pomocí Hammingovy vzdálenosti, která udává počet lišících se bitů u dvou stejně dlouhých bitových řetězců.

Výhodou Census transformace je její nezávislost na změně absolutních hodnot v intenzitě osvětlení. Nevýhodou je nemožnost výsledné atributy hierarchicky skládat, protože vstup a výstup z Census transformace nejsou stejného typu, a omezená invariance vůči geometrickým transformacím.

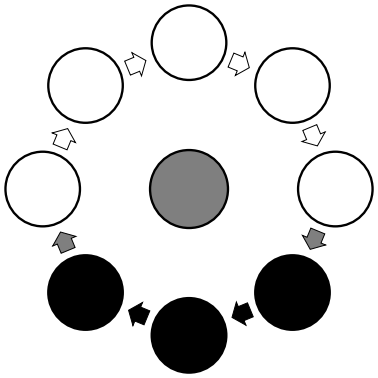
3.3.5 Local Binary Patterns

Local Binary Patterns (LBP) [Liao et al., 2007; Ojala et al., 2002; Petránek et al., 2012] podobně jako Census zachycují strukturu lokálního okolí každého bodu v obraze porovnáváním intenzit a ukládáním binárních hodnot do bitového řetězce. Na rozdíl od Census transformace jsou LBP tvořeny bitovým řetězcem z kruhového okolí o daném poloměru (obvykle 3 nebo 5 bodů). Výhodou tohoto přístupu je vyšší odolnost vůči změnám v rotaci objektu.

Výsledkem aplikace LBP operátoru je celé číslo reprezentující vlastnosti textury v daném bodě obrazu. Samotný výpočet LBP je založen na sestavení bitového řetězce z prahovaných bodů v kruhovém lokálním okolí každého bodu (viz Obrázek 5). Nejprve jsou vypočítány intenzity bodů ležících na prstencovém lokálním okolí. Jednotlivé intenzity jsou následně porovnány s intenzitou středového bodu. Body okolí s nižší intenzitou, než má středový bod, mají přiřazenu hodnotu 0, ostatní body hodnotu 1. Výsledná posloupnost nul a jedniček je interpretována jako celé číslo, které reprezentuje třídu textury v daném bodě. Formálně lze výpočet LBP definovat jako:

$$\mathrm{LBP}_{p,r}(x,y) = \sum_{i=0}^{p-1} 2^i \mathrm{thr}(\mathrm{I}(x_i\, y_i) - \mathrm{I}(x,y))$$

kde $\mathrm{LBP}_{p,r}(x,y)$ je hodnota LBP pro obrazový bod na souřadnicích $[x,y]$, p velikost kruhového okolí (počet bodů), r poloměr kruhového okolí, $\mathrm{I}(a,b)$ intenzita bodu obrazu na souřadnicích a,b a $\mathrm{thr}(v)$ je prahová funkce s nulovou hodnotou prahu. Bod se souřadnicí $[x_i,y_i]$ je bod ležící na kružnici o poloměru r se středem v bodě $[x,y]$.



Obrázek 5: Příklad Local Binary Pattern pro lokální kruhové okolí, černé body označují body s nižší intenzitou než je intenzita prostředního bodu, bílé body mají vyšší intenzitu. Počet bodů vzoru P = 8.

Dojde-li k rotaci vstupního obrazu, například v důsledku náklonu snímacího zařízení, dochází k posunu bitů v LBP vzoru, což vede k výrazně jiné hodnotě třídy textury. V literatuře publikovaným řešením závislosti na rotaci je posunutí každého vzoru do základní polohy. Základní poloha je definována jako cyklická rotace bitů rotačně závislého vzoru, která má maximální počet nulových bitů

na začátku řetězce. Nevýhodou rotačně invariantních LBP je jejich exponenciální počet ($O(2^P)$, kde *P* je počet bodů okolí). Pravděpodobnost výskytu některých vzorů je tak velmi malá a zmenšuje se s růstem počtu bitů vzoru. Pozorováním četností výskytu jednotlivých vzorů a jejich vlastností bylo zjištěno, že více než 80 % vzorů obsahuje 2 a méně přechodů 0/1 či 1/0, byly proto definovány uniformní vzory, které obsahují maximálně dva přechody mezi 0 a 1 bitem. Ostatní vzory jsou seskupeny do jedné kategorie. Celkem je tak definováno *P*+2 uniformních vzorů, což výrazně zvyšuje četnosti výskytu jednotlivých vzorů [Ojala et al., 2002].

Nevýhodou uniformních LBP je ztráta informace z 20 % vzorů, které jsou zařazeny do kategorie ostatních vzorů a náchylnost na šum v oblastech s konstantní světelností. V rámci disertačního výzkumu bylo navrženo rozšíření uniformních LBP, které zmíněné problémy řeší, viz kapitola 4.2 a [Petránek et al., 2012; Petránek et al., 2015; Petranek et al., 2016].

LBP podobně jako Census transformaci nelze hierarchicky skládat, pro složitější atributy je proto nutné je kombinovat pomocí jiného algoritmu.

3.3.6 Lokální histogramy

Metoda lokálních histogramů vypočítává histogram pro definované čtvercové či kruhové okolí každého bodu. Tyto histogramy je následně možné porovnávat pomocí statistických metod (korelace, testy o shodě distribučních funkcí) či pomocí metod teorie informace (entropie a mutual information) [Bradski et al., 2008; Petránek et al., 2013].

Výhodou lokálních histogramů je jejich invariance vůči posunutí a rotaci, což usnadňuje detekci objektů pod různými zornými úhly. Lokální histogramy jsou však citlivé na šum v osvětlení, obraz proto před použitím musí projít fází odstranění šumu a vyrovnáním světelných intenzit.

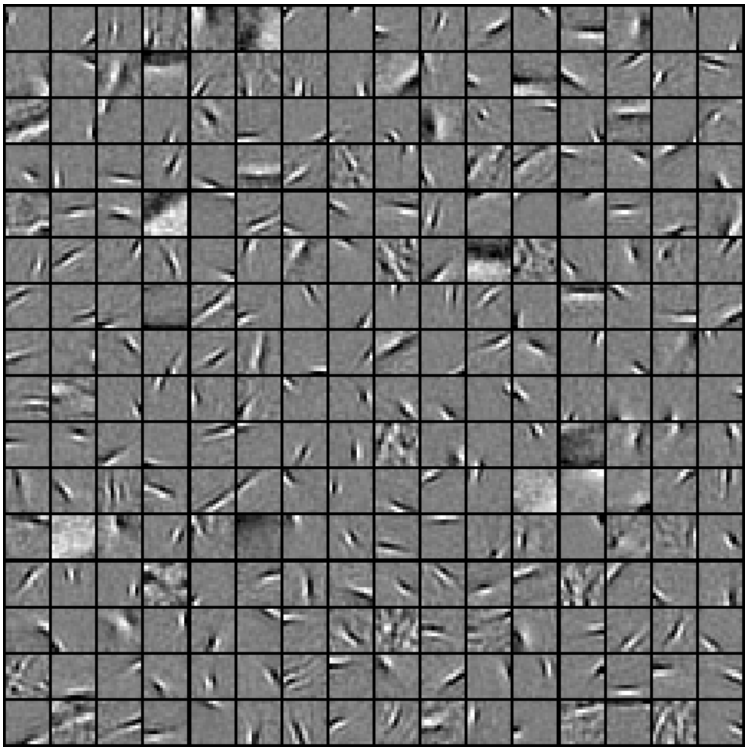
Histogramy lze použít i pro porovnávání strukturální informace na základě jednotlivých atributů, jako je gradient či hodnoty Local Binary Patterns. Výhodou histogramů je možnost jejich spojování, čímž lze docílit i invariance vůči změně měřítka.

Ačkoliv je teoreticky možné hierarchicky skládat histogramy, prakticky bývá problém s velkým množstvím rozměrů. Typický histogram má 16–256 intervalů a tvorba histogramů z těchto vektorů naráží na tzv. curse of dimensionality [Marimont et al., 1979].

3.3.7 Automatická extrakce atributů

Přístupy založené na tvorbě atributů experty mají výhodu v prokazatelnosti žádaných vlastností, jako jsou invariance vůči zkreslením, robustnost a nezávislost na trénovacích datech. Bohužel však díky tomu může docházet ke ztrátě informace, která je důležitá pro správné rozlišování objektů, ale není obsažena v okolí význačných bodů či regionů.

Automatická extrakce atributů se snaží tento problém vyřešit učením atributů přímo z obrazových dat. Základem je rozdělit obrazy z trénovací sady na menší výřezy (cca. 10×10 až 20×20 bodů) a převést je na *n*-rozměrné vektory. Tyto vektory lze chápat jako základní stavební bloky, ze kterých jsou sestaveny všechny obrazy. Cílem automatické extrakce atributů je nalézt vhodnou bázi pro tyto vektory. Vhodná báze je taková, která umožňuje aproximaci běžných vzorů v obraze tak, aby se co nejvíce koeficientů výsledné lineární kombinace blížilo nule. Výsledkem je řídké kódování (sparse coding) a získaná báze obsahuje vektory, které odpovídají často se vyskytujícím vzorům v obraze, jako jsou rohy či hrany [Olshausen et al., 1997]. Jako báze se často používá přeurčená báze, ve které je počet bázových vektorů vyšší, než je dimenze vektorového prostoru. Díky tomu lze dosáhnout stability výsledných kódování – malé změny ve vstupním vektoru vedou k malým změnám ve výsledném kódování. Obrázek 6 ukazuje aplikaci řídkého kódování na výřezy šedotónových obrazů.



Obrázek 6: Ukázka atributů nalezených pomocí řídkého kódování nad výřezy šedotónových obrazů. Obrázek převzat z [Ng et al., 2015a].

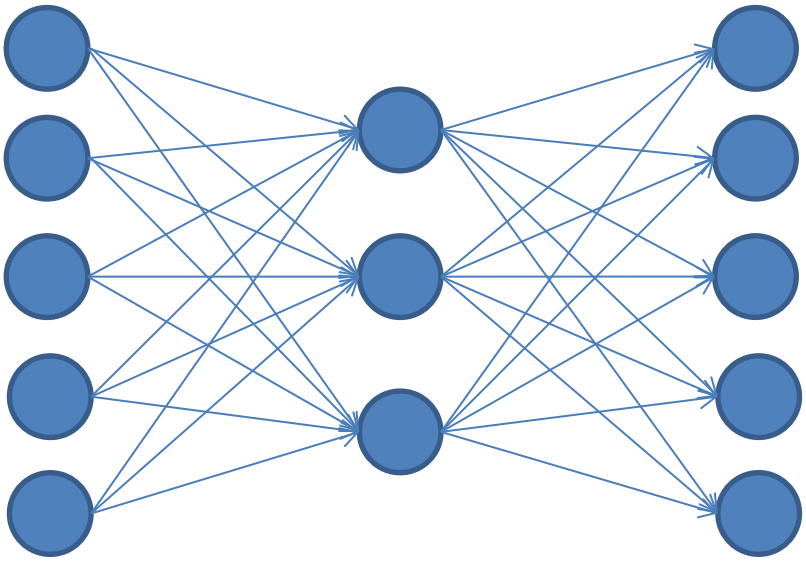
Formálně lze řídké kódování zapsat jako:

$$\vec{x} \approx \sum_{i=1}^{k} a_i \vec{b}_i$$

kde $\vec{x}$ je aproximovaný výřez, a_i skalární koeficienty, $\vec{b}_i$ bázové vektory a k počet bázových vektorů (musí být větší nebo roven dimenzi $\vec{x}$). Pro typické výřezy bude většina skalárních koeficientů a_i blízká nule.

Bázové vektory pro řídké kódování lze efektivně nalézt pomocí speciálně upravených dopředných neuronových sítí – autoencoderů. Vstupem do těchto sítí jsou jednotlivé výřezy z trénovací sady a cílem trénování je získat na výstupu stejný výřez, který je na vstupu. Autoencoder se tak trénuje na funkci identity. Architektura autoencoderu je reprezentována symetrickou dopřednou neuronovou sítí, v níž je omezena průměrná aktivace neuronů ve skrytých vrstvách, čímž je docíleno řídké aktivace. Pokud má autoencoder pouze jednu skrytou vrstvu, její neurony s odpovídajícími vahami odpovídají řídkému kódování trénovací sady. Váhy a neurony výstupní vrstvy pak tvoří odpovídající dekodér.

Autoencodery lze rovněž využít pro kompresi vstupu, kdy je prostřední vrstva autoencoderu omezena na nižší počet neuronů (viz Obrázek 7), a neuronová síť je tak během trénování nucena hledat úspornou reprezentaci. Výsledná reprezentace díky kompresi obsahuje výrazně méně šumu, autoencodery tak lze použít i ve fázi předzpracování obrazu.



Obrázek 7: Ukázka architektury autoencoderu s jednou skrytou vrstvou.

Autoencodery lze vzájemně skládat za sebe, tzn. kódování z prvního autoencoderu použít jako vstup pro další autoencoder. Výsledkem je hierarchie atributů, která zachycuje složitější struktury v obraze. Při dostatečném množství vrstev lze tímto způsobem získat atributy odpovídající např. lidské tváři [Le et al., 2013].

Je-li k dispozici dostatečné množství trénovacích dat (řádově statisíce obrazů), je možné atributy naučit přímo jako součást trénování rozsáhlé klasifikační sítě, bez nutnosti předtrénovat jednotlivé vrstvy pomocí autoencoderů [He et al., 2015].

3.4 Klasifikační algoritmy

3.4.1 Logistická regrese

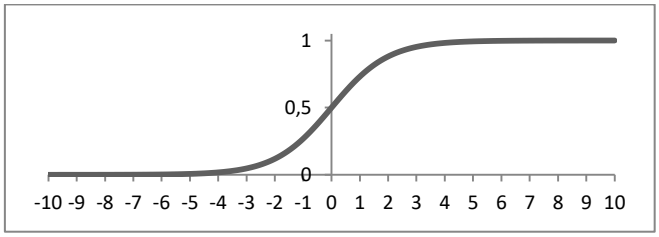
Logistická regrese je zobecněním lineární regrese s více proměnnými. Predikce hodnot je určena následující rovnicí:

$$y = \mathrm{L}(\vec{\theta} \cdot \vec{x})$$

kde y je predikovaná hodnota pro vstupní vektor dat $\vec{x}$, $\vec{\theta}$ vektor parametrů a $\mathrm{L}(z)$ logistická funkce:

$$\mathrm{L}(z) = \frac{1}{1 + e^{-z}}$$

Oborem hodnot logistické funkce je interval (0, 1), k jehož hranicím se funkce v $-\infty$, resp. $+\infty$, asymptoticky přibližuje. Díky těmto vlastnostem je možné logistickou regresi využít pro klasifikaci vstupních dat. Určením prahu v intervalu (0, 1) lze výstup interpretovat binárně, tedy např. jako „je objekt/není objekt“. Nejčastější hodnota pro tento práh je 0,5, tedy místo, kde logistická křivka protíná osu y (viz Obrázek 8).



Obrázek 8: Logistická křivka (sigmoida)

Vstupní vektor dat $\vec{x}$ je možné rozšířit o mocniny, kombinace a další nelineární transformace hodnot z tohoto vektoru a zajistit tak možnost nelineární klasifikační hranice.

Cílem při trénování je nalezení vektoru parametrů $\vec{\theta}$ takového, který minimalizuje funkci cena, což je čtverec rozdílu mezi predikovanými a očekávanými hodnotami y^*:

$$\mathrm{cena}(\vec{x}, \vec{\theta}) = (\mathrm{L}(\vec{\theta} \cdot \vec{x}) - y^*_{\vec{x}})^2$$

kde $\vec{x}$ je jeden vstup z trénovací sady a $y^*_{\vec{x}}$ jemu odpovídající správná predikce.

Tato rovnice však vede při minimalizaci k problémům s mizející derivací díky postupné saturaci logistické funkce. To způsobuje výpočetní komplikace při hledání minima, v praxi se proto používá alternativní forma minimalizované funkce:

$$\text{cena}(\vec{x}, \vec{\Theta}) = y^* \log\left(\text{L}(\vec{\Theta} \cdot \vec{x})\right) - (1 - y_{\vec{x}}^*) \log\left(1 - \text{L}(\vec{\Theta} \cdot \vec{x})\right)$$

Pro současnou minimalizaci cenové funkce pro všechna data v trénovací sadě je provedena minimalizace souhrnné funkce:

$$f(\vec{\Theta}) = \sum_i \text{cena}(\vec{x}_i, \vec{\Theta})$$

$$\arg\min_{\vec{\Theta}} f(\vec{\Theta})$$

Během trénování je trénovací sada rozdělena na dvě části – trénovací data a testovací data. Na trénovacích datech je provedena optimalizace a nalezení vektoru parametrů, na testovacích datech je poté ověřena schopnost predikce logistické regrese.

Výhodou logistické regrese je její matematická jednoduchost a možnost aproximace i velmi složitých funkcí, tedy i velmi složitých objektů v obraze. Nevýhodou je vysoká výpočetní náročnost při větších objemech dat a kombinacích atributů, kdy pro přesné řešení minimalizační rovnice je potřeba provádět náročné maticové operace a pro numerická řešení je nutné v každém kroku optimalizace počítat derivaci souhrnné funkce, a tedy projít celou sadu dat.

Logistická regrese a přístupy k dělení trénovací sady jsou základem pokročilejších algoritmů, jako jsou neuronové sítě či boosting.

3.4.2 Rozhodovací stromy

Rozhodovací stromy (decision trees) jsou stromové datové struktury, které je možné sestavit na základě trénovacích dat. Každý vrchol stromu obsahuje podmínku, podle které je při rozhodování vybrán dceřiný vrchol. Listy stromu obsahují hodnotu, která představuje výslednou klasifikaci. Více listů může reprezentovat jednu třídu, ke stejné klasifikaci je tedy možné dojít více cestami.

Pro klasifikaci objektů jsou výhodné binární rozhodovací stromy, kde každý vrchol (kromě listů) má právě dva potomky. Pro sestavování rozhodovacího stromu je použit rekurzivní algoritmus, který vybírá atribut, jenž nejlépe odděluje objekt a pozadí a z něho vytvoří nový vrchol stromu.

Výhodou rozhodovacích stromů je jejich snadná pochopitelnost pro člověka a rychlost klasifikace. Naopak, častým problémem rozhodovacích stromů je tendence k přetrénování, kdy strom správně klasifikuje trénovací data, není však schopen zobecnit rozhodování na dosud neznámá data.

3.4.3 Random forests

Random forests [Breiman, 2001] jsou zobecněním rozhodovacích stromů. Klasifikátor sestává z množiny rozhodovacích stromů, z nichž každý je vytvořen pomocí náhodného výběru několika atributů. Rozhodování probíhá na základě většinového hlasování jednotlivých rozhodovacích stromů o daném vstupu.

Oproti rozhodovacím stromům mají random forests menší problémy s přetrénováním [Breiman, 2001] a rozhodování na základě rozhodnutí více stromů je méně zatíženo šumem. Nevýhodou je delší trénovací doba (je nutné sestavit několik rozhodovacích stromů) i doba klasifikace, protože je nutné vypočítat rozhodnutí pro každý strom zvlášť a tato rozhodnutí poté agregovat.

Random forests byly s úspěchem použity pro sledování objektu ve videosekvenci, kde případné přetrénování je paradoxně spíše žádoucím efektem, protože umožňuje jednoznačněji identifikovat objekt v konkrétní sekvenci a odlišit jej od podobných objektů [Kalal et al., 2010b].

3.4.4 Boosting

Boosting je metoda strojového učení, která umožňuje spojení více dílčích rozhodovacích algoritmů do jednoho, který dává lepší výsledky než kterýkoliv z dílčích algoritmů samostatně. Předpokladem pro správnou funkčnost boostingu je, že každý z dílčích algoritmů je schopen odlišit nadpoloviční většinu příkladů objektů od příkladů, které objekt neobsahují (jeho rozhodování není zcela náhodné).

Nejznámějším příkladem boostingu je AdaBoost [Freund, 2001], který je lineární kombinací dílčích „slabých“ klasifikátorů:

$$\mathrm{H}(x) = \mathrm{sgn}(\sum_{t=1}^{T} \propto_t \mathrm{h_t}(x))$$

kde $\mathrm{H}(x)$ je výsledný klasifikátor, $\propto_t$ váhy dílčích klasifikátorů a $\mathrm{h}_t(x)$ dílčí klasifikátory, které vracejí +1 pro pozitivní a –1 pro negativní příklady objektu, a T počet dílčích klasifikátorů.

Postup trénování lze shrnout do následujících kroků:

- Nalezení dílčího klasifikátoru s nejmenší váženou chybou v predikci na trénovací sadě dat („nejlepšího klasifikátoru“).
- Výpočet váhy nalezeného klasifikátoru na základě počtu správných detekcí.
- Přiřazení vah příkladům v trénovací sadě, špatně detekovaným příkladům je váha zvýšena, správně detekovaným snížena.

Výsledkem je lineární kombinace klasifikátorů, která spojuje dílčí klasifikátory tak, aby výsledná klasifikační chyba byla co nejmenší. Detailní popis trénování je popsán v [Matas et al., 2012].

Modifikovaná verze AdaBoostu je využita pro detekci objektů v knihovně OpenCV [Bradski, 2000].

AdaBoost funguje jako meta algoritmus, který umožňuje spojit jiné typy klasifikátorů (např. neuronové sítě, rozhodovací stromy) a vytvořit tak silnější klasifikátor z několika slabších.

3.5 Umělé neuronové sítě

Dopředné neuronové sítě přinášejí v současnosti nejlepší výsledky v detekci a lokalizaci objektů ve statickém obraze, především proto, že jsou schopné v rámci trénování automaticky extrahovat významné atributy z trénovacích dat. Lze tak v rámci jednoho procesu trénování spojit dvě části detekčního řetězce – extrakci atributů a klasifikaci, což vede k výpočetním úsporám a zvýšení přesnosti detekce.

Vzhledem k širokým možnostem použití neuronových sítí je možné je zařadit jak mezi algoritmy pro extrakci atributů, tak mezi klasifikační algoritmy a algoritmy pro analýzu videosekvencí.

3.5.1 Neurony

Umělé neuronové sítě jsou složeny z neuronů a jejich vzájemného propojení. Neuron je definován jako funkce, která kombinuje vektor vstupů ze vstupních neuronů do jednoho skalárního výstupu. Neuron má obvykle ke každému vstupu přiřazenu skalární váhu, která určuje sílu propojení se vstupním neuronem. Výstup neuronu je pak vypočítán jako skalární součin vektoru vstupů a vektoru vah, na který je aplikována nelineární aktivační funkce:

$$n = a(\vec{w}_n \cdot \vec{z})$$

kde n je výstup neuronu, a je aktivační funkce, $\vec{w}_n$ vektor vah a $\vec{z}$ vektor vstupů.

Neuronová síť může obsahovat neurony s různými aktivačními funkcemi. Aby výsledná neuronová síť mohla reprezentovat nelineární závislosti mezi vstupem a výstupem, musí obsahovat alespoň jednu nelineární aktivační funkci. Mezi běžně používané nelineární aktivační funkce patří logistická funkce (sigmoida), hyperbolický tangens a rektifikovaná lineární jednotka.

Sigmoida a hyperbolický tangens jsou funkce, jejichž průběh je přibližně lineární okolo nuly, v nekonečnu se limitně blíží 1 a v minus nekonečnu se limitně blíží 0 v případě sigmoidy, resp. –1 v případě hyperbolického tangentu. Neurony s těmito aktivačními funkcemi mají tendenci utlumovat silné signály a nutit aktivace neuronů do oblasti okolo nuly během trénování. To může být výhodné pro některé typy problémů, ale v případech, kdy jsou pro správné učení nutné velké rozdíly mezi aktivacemi jednotlivých neuronů, není tento typ aktivačních funkcí vhodný.

Funkce $\mathrm{softplus}(x) = \log(1 + e^x)$ se v minus nekonečnu limitně blíží 0 a v plus nekonečnu funkci $f(x) = x$. Na rozdíl od sigmoidy a hyperbolického tangens tak nedochází k saturaci aktivací neuronů. Tuto funkci lze zároveň aproximovat pomocí rektifikované lineární funkce $\mathrm{ReLU}(x) = \max(0, x)$, jejíž výpočet je velmi rychlý [Glorot et al., 2011]. Nevýhodou použití aproximace pomocí

maxima je nulová derivace pro $x < 0$, čímž v průběhu učení může dojít k deaktivaci neuronu, ale ne k jeho opětovné aktivaci. Jako alternativu je možné použít funkci, která má v záporné části malou konstantní derivaci:

$$\mathrm{LReLU}(x) = \begin{cases} 0{,}01x, & x < 0 \\ x, & x \geq 0 \end{cases}$$

Díky výpočetní nenáročnosti a předcházení problémů se saturováním neuronů používají moderní neuronové sítě pro rozpoznávání obrazu právě rektifikované lineární neurony [Szegedy et al., 2014].

Výstup aktivační funkce je výstupem neuronu, tento výstup je možné napojit jako vstup do dalších neuronů a vybudovat tak celou neuronovou síť.

3.5.2 Architektura dopředných neuronových sítí

U dopředných neuronových sítí jsou neurony sdruženy do po sobě jdoucích vrstev, kde neurony v každé vrstvě mají propojení pouze na neurony z předchozí vrstvy. Dle hustoty propojení neuronů s předchozí vrstvou lze vrstvy rozdělit na plně propojené a částečně propojené, z nichž jsou pro rozpoznání objektů v obraze používány především konvoluční neurony [LeCun et al., 1998]. Moderní neuronové sítě rovněž obsahují pomocné vrstvy, které zjednodušují či normalizují vstup – jedná se především o redukční (pooling) vrstvy, normalizace vstupní dávky (batch normalization) a dropout, které jsou popsány níže.

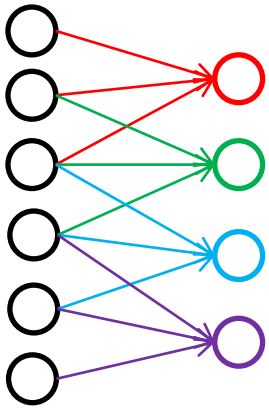
3.5.2.1 Plně propojené vrstvy

Každý neuron v plně propojené vrstvě obsahuje spojení na všechny neurony z předcházející vrstvy. Plně propojené vrstvy jsou vhodné pro analýzu širších souvislostí, kde informace nutná pro správnou detekci může být přítomna ve všech neuronech předcházející vrstvy. Při detekci objektů v obraze se plně propojené neurony obvykle používají jako součást finálních vrstev rozsáhlejší neuronové sítě, kde je nutné shrnout informaci z celého obrazu do několika málo výstupů.

3.5.2.2 Konvoluční vrstvy

Neurony v částečně propojených vrstvách obsahují propojení pouze na některé neurony z předcházející vrstvy. Neurony konvoluční vrstvy obsahují propojení jen na ty neurony v předchozí vrstvě, které leží v určitém regionu vizuálního pole, jak znázorňuje Obrázek 9. Díky tomu je uspořeno velké množství výpočtů oproti plně propojeným vrstvám, aktivace neuronů konvoluční vrstvy lze efektivně vypočítat pomocí konvolucí nad vstupními neurony, kde konvolučními jádry jsou matice vah [LeCun et al., 1995]. Konvoluční vrstvy jsou používány především jako první vrstvy neuronové sítě, kde analyzují vstupní obraz a extrahují z něj vhodné atributy pro plně propojené vrstvy. Pro snazší a rychlejší trénování lze váhy konvolučních vrstev inicializovat pomocí vah z řídkých

autoencoderů popsaných v kapitole 3.3.7, pomocí omezených Bolzmannových strojů (Restricted Bolzmann Machine, RBM) [Salakhutdinov et al., 2009; Hinton et al., 2012b] či z vrstev neuronových sítí již natrénovaných na příbuzný problém.



Obrázek 9: Ukázka 1D konvoluční vrstvy. Každý neuron je propojen pouze s několika okolními neurony z předchozí vrstvy. Všechny neurony zobrazené konvoluční vrstvy mají stejné váhy, detekují tak jeden konkrétní atribut v celém vstupu.

3.5.2.3 Pooling vrstvy

Aby neuronová síť byla schopna detekovat objekt na různých pozicích ve vstupním obraze, jsou do sítí zařazovány redukční vrstvy, které seskupují výstupy předcházejících vrstev. Tím se rozšiřuje oblast obrazu, kterou mohou neurony v následující vrstvě analyzovat. Jako redukční operace se nejčastěji používá maximum, k dalšímu zpracování je tak poslán nejsilnější signál z redukované oblasti. Nevýhodou redukčních vrstev je ztráta informace o výsledném umístění. Neuronová síť je schopna detekovat přítomnost objektu, není však zachována informace o tom, kde v původním obraze se objekt nachází. Informaci o umístění objektu je tak nutné dohledat jiným způsobem, např. zpětným trasováním jednotlivých aktivací neuronů či vytvořením regresní sítě [Simonyan et al., 2013].

3.5.2.4 Batch Normalization

Batch Normalization [Ioffe et al., 2015] je vrstva, která v průběhu trénování postupně vypočítává plovoucí průměr a rozptyl pro každý výstup předchozí vrstvy. Pomocí vypočítaného průměru a rozptylu poté jednotlivé vstupy normalizuje na nulovou střední hodnotu a jednotkový rozptyl:

$$\vec{x} = \frac{\vec{x} - E'[x]}{\sqrt{Var'[x]}}$$

kde $E'[x]$ je plovoucí průměr (odhad střední hodnoty) a $Var'[x]$ plovoucí rozptyl (odhad populačního rozptylu).

Protože příliš agresivní normalizace může ovlivnit schopnost sítě reprezentovat libovolnou funkci, obsahují Batch Normalization vrstvy i dva trénovatelné parametry, měřítko (γ) a posun (β), které upravují sílu normalizace:

$$\vec{x} = \frac{\vec{x} - (E'[x] - \beta)}{\sqrt{Var'[x]}}\gamma$$

Měřítko i posunutí jsou trénovány stejným způsobem, jako váhy v tradičních vrstvách.

Výhodou normalizace je vyšší rychlost trénování, protože následující vrstvy sítě mají během trénování stabilnější vstupní rozdělení hodnot, což umožňuje rychlejší konvergenci gradientních optimalizačních metod [Ioffe et al., 2015].

3.5.3 Trénování neuronových sítí

Neuronovou síť lze vyjádřit jako funkci, která vektoru vstupů přiřazuje vektor výstupů a jejíž chování lze ovlivnit změnou parametrů představovaných vahami jednotlivých neuronů. Na začátku trénování jsou jednotlivé váhy v síti nastaveny náhodně nebo podle vah nalezených pomocí předtrénování [Hinton et al., 2006]. Rozsah vah při náhodné inicializaci je velmi důležitý, při nesprávné inicializaci vah dochází během trénování ke konvergenci do nevhodných lokálních minim [Hinton et al., 2012a].

Během samotného procesu trénování dochází k výpočtu chyby porovnáním hodnot výstupních neuronů a očekávaného výstupu pro jednotlivé obrazy z trénovací sady dat. Následně je vypočítána derivace této chyby podle jednotlivých vah pomocí algoritmu backpropagation [Rumelhart et al., 1988]. Na základě vypočítaných derivací jsou upraveny jednotlivé váhy pomocí optimalizačního algoritmu. Mezi nejpoužívanější optimalizační algoritmy patří stochastická metoda nevětšího spádu (Stochastic Gradient Descent, SGD) [Weisstein, 2015], momentová metoda největšího spádu

(Gradient Descent with Momentum) [Rehman et al., 2011], RMSProp [Hinton, 2012], Nesterov Accelerated Gradient (NAG) [Nesterov, 2007], ADAGRAD [Duchi et al., 2011], ADADELTA [Zeiler, 2012] a ADAM [Kingma et al., 2014].

Pro výpočet chyby lze použít různé chybové funkce. Často používanými chybovými funkcemi je střední kvadratická chyba (Mean Squared Error, MSE) pro rektifikované lineární neurony, binární křížová entropie (Binary Cross Entropy) pro sigmoidové neurony a kategorická křížová entropie (Multi-class/Category Cross Entropy) pro softmax neurony u klasifikačních sítí, kde se jednotlivé třídy vzájemně vylučují a jejichž výstupem je hustota pravděpodobnosti výskytu jednotlivých tříd [Nielsen, 2015].

Celkově lze proces trénování shrnout jako optimalizační proces, který hledá sadu vah takovou, která minimalizuje rozdíl mezi ideálním výstupem a výstupem sítě:

$$f(w) = \sum_{i=1}^{n} error(N(w, x_i), y_i)$$

$$\arg\min_{w} f(w)$$

kde $error()$ je chybová funkce, $N(w, x_i)$ je výstup neuronové sítě pro dané váhy w a vstup x_i, y_i očekávaný výstup a n velikost trénovací sady.

3.5.4 Regularizace neuronových sítí

U rozsáhlých neuronových sítí s miliony parametrů, které jsou běžné při detekci objektů v obraze, často dochází k přetrénování sítě. Při přetrénování má síť velmi malou chybovost na trénovací sadě, není však schopna výsledky zobecnit na příklady mimo trénovací sadu. Cílem regularizace je omezit přetrénování a zlepšit generalizační schopnosti trénovaných sítí.

Základní metodou regularizace je penalizace velikosti vah [Moody et al., 1995]. Toho je dosaženo přidáním kritéria minimalizace součtu druhých mocnin vah k funkci, která je optimalizována během trénování:

$$f(w) = \sum_{i=1}^{n} error(N(w, x_i), y_i) + \lambda \sum_{i} w_i^2$$

$$\arg\min_{w} f(w)$$

kde λ je regularizační parametr, který udává, jak velký důraz je při optimalizaci kladen na minimalizaci vah.

Výsledkem je omezení rozptylu vah a tím i omezení extrémních aktivací některých neuronů, které mohou vést k prudkým změnám výstupu při drobných změnách vstupu. Zároveň však dochází i k omezení vyjadřovacích možností sítě, takže je velmi důležité správně nastavit parametr λ.

Další možností regularizace je průměrování predikce sítí natrénovaných na různých částech trénovací sady. Výsledkem je vyšší odolnost proti přetrénování, protože ačkoliv jednotlivé sítě mohou být přetrénovány na své části datové sady, ve výsledné predikci je zastoupeno trénování na celé sadě. Díky oddělenému trénování spolu sítě nemohou sdílet informace a stát se vzájemně závislými, výsledná průměrná predikce proto lépe zobecňuje. Nevýhodou tohoto přístupu je velká náročnost na výpočetní prostředky, protože musí být natrénováno velké množství neuronových sítí. V případě detekce obrazu a sítí s miliony parametrů trvá trénování jedné sítě řádově dny, což znemožňuje aplikaci této metody v praxi.

Dropout je regularizační metoda, která je založena na průměrování sítí, eliminuje však výše zmíněnou výpočetní náročnost [Hinton et al., 2012a]. Trénována je pouze jedna neuronová síť a v průběhu trénování je síti představena vždy pouze část trénovací sady. Trénování na vybrané části trénovací sady potom probíhá na síti s 50 % náhodně vypnutými neurony. Dropout lze tedy zkráceně popsat pomocí následujícího algoritmu:

1. Vyber M, podmnožinu příkladů z testovací sady
2. Náhodně vyber 50 % neuronů a deaktivuj je
3. Proveď standardní trénování na sadě M

Výsledkem je neuronová síť, která je průměrem dílčích neuronových sítí. Lze dokázat, že pokud jsou po dokončení trénování všechny váhy vyděleny 2, odpovídá predikce výsledné síti geometrickému průměru predikcí všech možných dílčích sítí [Hinton et al., 2012a]. Díky trénování pouze na částech trénovací sady a na síti poloviční velikosti Dropout dokonce urychluje trénovací proces.

Na generalizační schopnost sítě může mít rovněž vliv zvolený optimalizační algoritmus. Minima nalezená pomocí metody největšího spádu jsou často vhodnější než minima nalezená pomocí pokročilejších metod využívajících předchozích iterací, jako je RMSProp či ADAM [Wilson et al., 2017].

3.5.5 Rekurentní neuronové sítě

Standardní dopředné neuronové sítě nejsou schopné modelovat stav a tím i časově závislá data. Pro každý vstup je na základě vah vypočítána aktivace neuronů, nezávisle na vstupech předchozích. Ačkoliv tato vlastnost plně postačuje pro detekci objektů ve statických obrazech, u videosekvencí výsledek rozpoznání často závisí na předchozích snímcích.

Rekurentní neuronové sítě obsahují propojení neuronů, která tvoří cykly ve výsledném grafu sítě. Díky tomu mohou v síti vznikat složité interakce neuronů umožňující zachycení stavu v čase. Aktivace neuronů rekurentní sítě v jednom kroku je vypočítána podobně jako u dopředných sítí skalárním součinem vah a aktivací ze vstupních neuronů v předchozím kroku, oproti dopředným neuronovým sítím je však jejich chování výrazně složitější. Je možné pomocí nich simulovat Turingův stroj [Graves et al., 2014], teoreticky tak mohou provádět libovolné výpočty, což ztěžuje trénování a analýzu.

Pro trénování rekurentních neuronových sítí je možné použít algoritmus backpropagation podobně jako u dopředných sítí. Rekurentní síť je na začátku trénování rozvinuta v čase do standardní dopředné sítě, která je následně natrénována běžným způsobem [Mozer, 1989]. Výsledné váhy natrénované dopředné sítě jsou přeneseny zpět do rekurentní sítě. Pokud jedné váze v rekurentní síti odpovídá více vah v dopředné síti, jsou při přenosu jejich hodnoty zprůměrovány.

Při trénování rekurentních sítí často dochází k problému mizejícího gradientu, kdy se derivace vah během zpětné propagace s každou vrstvou exponenciálně snižují [Hochreiter, 1998]. Tímto problémem trpí především sítě s neurony, jejichž aktivační funkce je sigmoida či hyperbolický tangens. Problém mizejícího gradientu způsobuje, že je velmi těžké zachytit vzdálené časové souvislosti. Řešením tohoto problému je použití speciálních paměťových neuronových bloků Long Short Term Memory (LSTM), které jsou schopny díky speciálnímu návrhu udržet informaci po mnoho časových úseků [Hochreiter et al., 1997]. LSTM bloky jsou složeny z diferencovatelných neuronů, lze je proto trénovat pomocí backpropagation stejně jako zbytek sítě.

3.5.6 Generative Adversarial Networks

Generative Adversarial Networks (GAN) je systém dvou neuronových sítí, generátoru a diskriminátoru, které slouží ke generování dat, jejichž rozdělení odpovídá pravděpodobnostnímu rozdělení dat trénovací sady [Goodfellow et al., 2014]. Vstupem do generátoru je vektor náhodných hodnot, výstupem je vektor hodnot v cílovém pravděpodobnostním rozdělení (např. barevný obraz).

Proces trénování GAN je založen na principu soutěže mezi generátorem a diskriminátorem. Cílem generátoru je produkovat věrohodná výstupní data, cílem diskriminátoru je rozlišit, zda daný vstupní vektor pochází z trénovací sady, nebo byl vytvořen generátorem. Střídavým trénováním generátoru a diskriminátoru tak lze dosáhnout toho, že generátor produkuje výstup podobný datům z trénovací sady. Při aplikaci na obrazová data tak lze např. generovat věrohodné fotografie.

Podmíněné GAN (Conditional GAN, CGAN) rozšiřují GAN o dodatečný vstupní vektor, který slouží jako základ pro výstupní vektor. CGAN lze použít například pro obarvování šedotónových obrazů, generování fotografií na základě náčrtů apod. [Isola et al., 2016]. Během trénování generátor musí

minimalizovat jak chybu z diskriminátoru, tak vzdálenost od ideálního výstupu, čímž je dosaženo podmíněnosti výstupu na daném vstupním vektoru. Diskriminátor u CGAN tak slouží jako dodatečná chybová funkce, která penalizuje strukturální odchylky v pravděpodobnostním rozdělení generovaných výstupních vektorů od trénovací sady. Modifikovaný princip CGAN je rovněž použit v navrženém řešení pro generování realistických segmentací objektů.

Hlavním problémem při použití GAN a CGAN je náročný proces trénování – obě neuronové sítě, generátor a diskriminátor, se během optimalizace mohou zastavit ve špatném lokálním minimu, případně může generátor produkovat pouze jeden či několik málo výstupů nezávislých na vstupním vektoru (tzv. mode collapse). Pro minimalizaci těchto problémů se používá vysoká míra regularizace, střídání epoch, kdy se trénuje buď výhradně generátor, nebo diskriminátor, a pokročilé chybové funkce pro trénování diskriminátoru (např. Earth Mover Distance v případě Wasserstein GAN [Arjovsky et al., 2017]).

3.5.7 Existující architektury pro detekci objektů

Všechny současné detekční architektury jsou založené primárně na konvolučních vrstvách. Detekční síť postupně transformuje vstupní obraz na abstraktnější vektory atributů, které lépe zachycují podobu objektu pod různými zkresleními. U všech zmíněných architektur se počet konvolučních jader, tedy i kardinalita vektorů atributů, postupně zvyšuje s hloubkou sítě.

3.5.7.1 AlexNet

Jednou z prvních architektur navržených pro detekci objektů v obraze je AlexNet [Krizhevsky et al., 2012], která využívá střídavé kombinace konvolučních vrstev a max pooling vrstev pro generování abstraktních atributů. Konvoluční vrstvy postupně mají více konvolučních jader (96 v první vrstvě, 256 ve druhé, 384 ve třetí), což je výpočetně kompenzováno postupným zmenšováním vstupu pomocí max pooling vrstev. Atributy poslední konvoluční vrstvy jsou poté předány do 3 plně propojených vrstev, jejichž výstupem je pravděpodobnostní rozdělení výskytu jednotlivých kategorií objektů – výstupní neurony mají výstup od 0 do 1, pro každý typ detekovaného objektu je jeden výstup.

3.5.7.2 VGG

Architektura VGG je založena na kombinaci třech 3×3 konvolučních vrstev a max pooling vrstev. Základní princip architektury spočívá v rozdělení větších konvolucí (např. 7×7 či 9×9) na několik po sobě jdoucích 3×3 konvolucí. Tím je omezeno množství parametrů, což vede k regularizaci konvolučních filtrů a úspoře výpočetních prostředků. Výstupní neurony poslední konvoluční vrstvy jsou podobně jako u AlexNet napojeny na 3 plně propojené vrstvy, jejichž výstupem je pravděpodobnostní rozdělení přítomnosti jednotlivých objektů.

3.5.7.3 Network in Network

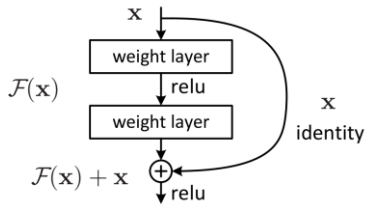
Architektura Network in Network rozšiřuje vrstvy s lineárními konvolucemi na konvoluční vícevrstvé perceptrony – každé lineární konvoluční jádro z tradičních architektur je nahrazeno vícevrstvým perceptronem. Dochází tak k vytvoření nelineárních konvolučních jader, která mohou zachytit složitější atributy. Tyto rozšířené konvoluční vrstvy jsou následně prokládány max pooling vrstvami pro postupnou redukci rozlišení, podobně jako u výše zmíněných architektur.

3.5.7.4 Inception

Inception architektura je založena na kombinaci několika různých konvolučních jader v rámci jedné vrstvy. Místo jednoho konvolučního jádra jsou tak aplikována 1×1, 3×3 a 5×5 konvoluční jádra, jejichž výstup je poté napojen do výsledného vektoru. Novější iterace Inception architektury rovněž využívají lineárně oddělitelných konvolucí, kdy je nejprve aplikována 1×7 konvoluce v horizontálním směru, a následně 7×1 konvoluce ve vertikálním směru, což vede k další úspoře parametrů a výpočetního výkonu.

3.5.7.5 ResNet

Architektura ResNet [He et al., 2016] přidává do konvoluční architektury reziduální spojení, která umožňují snáze reprezentovat identitu. Tradiční neuronové architektury se funkci identity učí pouze obtížně, což limituje možnosti zvyšování hloubky sítě, a tím i její schopnost reprezentace složitějších konceptů. Reziduální spojení propojuje vstup s výstupem (viz Obrázek 10) a mezivrstvy se učí pouze rozdíl mezi vstupem a očekávaným výstupem. Pro aproximaci identity tak stačí, aby výstupem mezivrstev byl nulový vektor, čehož lze dosáhnout naučením vah na 0.



Obrázek 10: Demonstrace reziduálního spojení architektury ResNet. Obrázek převzat z [He et al., 2016].

3.5.8 Existující architektury pro lokalizaci objektů

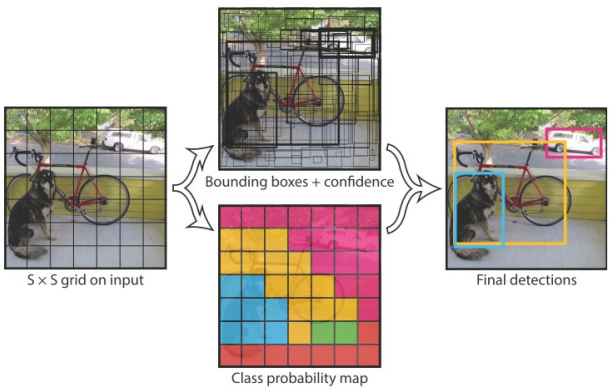
Lokalizace objektů v obraze je rozšířením problému detekce objektů. Cílem lokalizace je kromě určení typu objektu stanovit též jeho umístění v obraze pomocí ohraničujícího obdélníku. Pokud je v obraze přítomno více objektů, neuronová síť by měla lokalizovat všechny.

3.5.8.1 R-CNN, Fast R-CNN a Faster R-CNN

R-CNN, Fast R-CNN a Faster R-CNN [Girshick et al., 2016; Girshick, 2015; Ren et al., 2015] proces lokalizace rozdělují na tři fáze – generování potenciálních ohraničujících obdélníků, filtrování vygenerovaných umístění a určení typu objektu pro zbývající umístění. Pro filtrování a určení typu objektu jsou využívány výše uvedené architektury pro detekci objektů. R-CNN a Fast R-CNN používá pro generování potenciálních umístění Selective Search, který generuje 1000-10000 potenciálních ohraničujících obdélníků. R-CNN poté aplikuje detekční neuronovou síť na každý z těchto obdélníků a rozhodne, zda a kterému objektu odpovídá. Fast R-CNN proces filtrace a určení objektu optimalizuje rozdělením detekční sítě na konvoluční část obsahující pouze konvoluční a pooling vrstvy, která jsou aplikovány pouze jednou na celý obraz, a klasifikátor, který je aplikován na výstup konvoluční části v každém obdélníku. Protože většina výpočtů detekčních sítí je soustředěna v konvolučních vrstvách, dochází k výrazné úspoře výpočetních prostředků, protože tyto vrstvy jsou vyhodnoceny pouze jednou. Faster R-CNN dále rozšiřuje Fast R-CNN a nahrazuje Selective Search regresní neuronovou sítí, která generuje potenciální obdélníky. Celý systém je tak diferencovatelný a je možné jej trénovat najednou jako celek.

3.5.8.2 YOLO

Architektura YOLO (You Only Look Once) [Redmon et al., 2016a] je alternativní architektura pro detekci umístění objektů v obraze. YOLO sestává z jedné neuronové sítě, která predikuje jak ohraničující obdélníky, tak typ objektu. První část sítě je složena ze standardní kombinace konvolučních a pooling vrstev. Druhá část sestává z regresních vrstev, jejichž výstupem je mřížka, kde každá buňka obsahuje informaci o typu detekovaného objektu a fixní počet ohraničujících obdélníků s indikací spolehlivosti pro každý obdélník (viz Obrázek 11). Za výslednou detekci jsou vybrány obdélníky s maximální predikovanou spolehlivostí. Díky jednoduchosti architektury je možné detekci provádět s dostatečně výkonnou grafickou kartou v reálném čase (> 30 FPS).

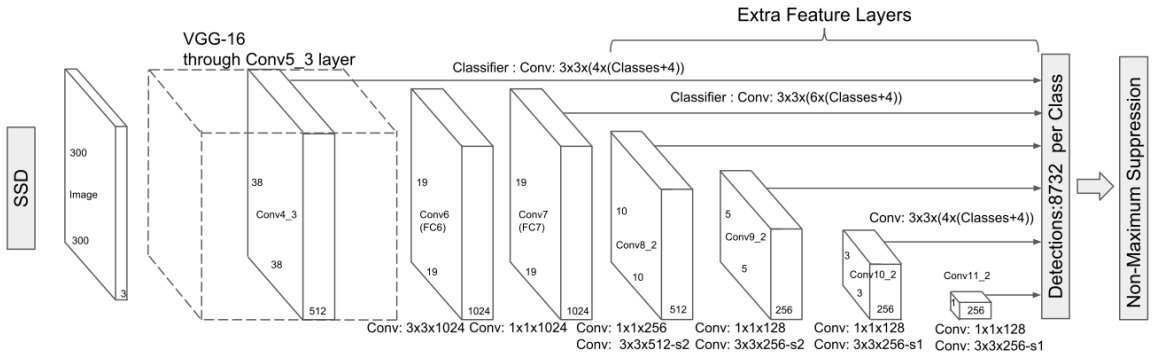


Obrázek 11: Princip lokalizační architektury YOLO. Obrázek převzat z [Redmon et al., 2016a].

Architektura YOLOv2 je založena na stejném principu jako YOLO, avšak umožňuje detekci až 9000 různých druhů objektů a zpřesňuje ohraničující obdélníky [Redmon et al., 2016b].

3.5.8.3 Single Shot Multibox Detector

Architektura Single Shot Multibox Detector (SSD) [Liu et al., 2016] je založena na podobném principu jako YOLO. SSD vychází z detekční architektury VGG, ze které odstraňuje poslední plně propojené vrstvy a nahrazuje je přídavnými konvolučními vrstvami s různými rozlišeními. Výstupem těchto konvolučních vrstev jsou odhady ohraničujících obdélníků, podobně jako u YOLO, avšak díky použití více rozlišení jsou lépe detekovány objekty s různým měřítkem.



Obrázek 12: Architektura SSD a použití konvolučních vrstev s různými rozlišeními pro predikci ohraničujících obdélníků. Obrázek převzat z [Liu et al., 2016].

3.5.9 Existující architektury pro segmentaci objektů

Segmentace objektů v obraze je rozšířením problému lokalizace objektů. Cílem je pro každý pixel obrazu rozhodnout, zda náleží nějakému objektu či nikoliv. Výstupem je tak sada bitmapových masek pro každý objekt.

3.5.9.1 Plně konvoluční sítě

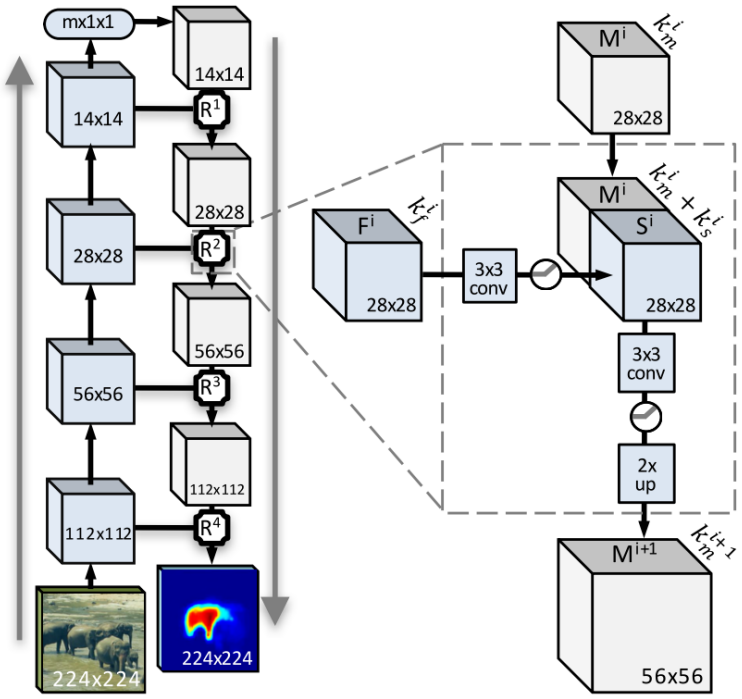
Plně konvoluční sítě (Fully Convolutional Networks, FCN) vycházejí z jednoduché transformace detekčních sítí, kdy jsou finální plně propojené vrstvy transformovány na konvoluční vrstvy [Long et al., 2015]. Výstupem je tak namísto jednoho softmax vektoru 2D mapa softmax vektorů s rozlišením odpovídajícím poslední původní konvoluční vrstvě. Výsledkem je hrubá mapa objektů, kterou lze považovat za segmentační masku. Tato architektura je schopna rozlišovat pouze typy objektů, nikoliv jednotlivé instance. Díky tomu, že plně konvoluční sítě neobsahují plně propojené vrstvy, je lze snadno aplikovat na vstupní obrazy s libovolným rozlišením.

3.5.9.2 *DeepMask*

DeepMask [Pinheiro et al., 2015] je segmentační architektura založená na detekční síti VGG, ze které jsou podobně jako u FCN odstraněny finální plně propojené vrstvy. Místo původních plně propojených vrstev se DeepMask rozděluje na dvě větve – jedna větev predikuje segmentační masku pro daný vstupní výřez obrazu, druhá větev predikuje, zda je v daném místě přítomen objekt či nikoliv. Během trénování je síť DeepMask trénována na jednotlivých výřezech, při predikci segmentace celé scény je hustě aplikována na výřezy scény. Tato architektura je schopna odlišit jednotlivé instance objektů, tedy např. při přítomnosti více osob na snímku má každá svou vlastní segmentační masku.

3.5.9.3 *SharpMask*

SharpMask [Pinheiro et al., 2016] je rozšířením architektury DeepMask, jejímž cílem je produkce ostřejších a přesnějších segmentačních masek. SharpMask toho dosahuje zpětným doplňováním detailů přidáním propojení na dílčí konvoluční vrstvy základní VGG architektury, které mají vyšší rozlišení (viz Obrázek 13). Zatímco původní VGG vrstvy postupně zvyšují abstrakci a snižují rozlišení, SharpMask zpětným propojováním na vrstvy s nižší abstrakcí a vyšším rozlišením obnovuje informace o hranách, které jsou nutné pro vytvoření přesné segmentační masky.



Obrázek 13: Architektura SharpMask ukazující zpětná propojení na původní vrstvy sítě VGG. Obrázek převzat z [Pinheiro et al., 2016].

3.5.9.4 Fully Convolutional Instance-aware Semantic Segmentation

Fully Convolutional Instance-aware Semantic Segmentation (FCIS) [Li et al., 2016] je architektura založená na kombinaci lokalizační sítě a sítě pro predikci segmentační masky. FCIS nejprve detekuje umístění objektů v obraze pomocí ohraničujících obdélníků. Jednotlivé obdélníky jsou poté předány jako výřezy do modulu, který predikuje segmentační masku. Hlavním přínosem této architektury je chytré sdílení konvolučních vrstev mezi moduly pro detekci typu objektu, výpočet ohraničujících obdélníků a výpočet segmentační masky.

3.6 Detekce objektů ve videosekvenci

Z hlediska počítačového vidění odpovídá detekce objektu ve videosekvenci detekci objektu v obraze s časovou dimenzí. Jedná se tedy o zobecnění přístupu popsaného v předchozích kapitolách. Nejjednodušší technikou je projekce do původního podprostoru a ignorování časové dimenze. Objekt je tedy detekován v každém snímku nezávisle na předcházejících snímcích sekvence. Tuto techniku je možné použít, je-li objekt dobře popsán jednoduchými atributy a lze jej efektivně detekovat oproti pozadí.

Při sledování složitějších objektů je vhodné využít dodatečného rozměru a informací pro zpřesnění detekce. Po sobě jdoucí snímky ve videosekvenci jsou často korelované, čehož lze využít pro urychlení výpočtů a omezení množství šumu.

Jedním z možných způsobů, jak využít časovou dimenzi, je natrénování rekurentní neuronové sítě již zmíněné v kapitole 3.5.5, tento přístup je však z pohledu trénování velmi časově i datově náročný. V následujících podkapitolách jsou proto diskutovány především techniky, které nejsou založeny na neuronových sítích. V navrženém řešení je využita kombinace neuronových sítí a níže popsaných technik pro sestavení detektoru objektů ve videu.

3.6.1 Sledování objektů pomocí metod modelování pozadí

Modelování pozadí vychází z předpokladu, že ve videosekvenci se nachází pozadí scény a sledovaný objekt (či objekty), který se pohybuje. Postupným sledováním změněných obrazových bodů lze určit statické body (pozadí) a dynamické body (objekt). Detekce objektů v následujícím snímku je provedena odečtením pozadí a hledáním oblastí s nejvyššími absolutními rozdíly od pozadí.

Je-li známa prostorová informace pro každý bod obrazu, například při zpracování stereoskopického vstupu nebo při použití speciálního hardware, a je-li možné předpokládat výskyt objektu v popředí scény, je modelování pozadí zjednodušeno na vyhledání prahu prostorové souřadnice ([Petránek et al., 2010], [Lee et al., 2007]).

Častým problémem metod modelování pozadí v praxi je porušení výchozích předpokladů – sledovaný objekt může být stacionární, a naopak pozadí může být dynamické (např. rušná ulice). Častým problémem při modelování pozadí jsou též stíny sledovaných objektů, které se pohybují spolu s objektem, ačkoliv nejsou jeho součástí. Existují algoritmy, které omezují dopady porušení těchto předpokladů [Horprasert et al., 1999], avšak úplně je eliminovat není možné.

3.6.2 Sledování pohybu v obraze pomocí optického toku

Alternativou k detekci objektů pomocí metod modelování pozadí je detekovat objekt v prvním snímku pomocí metod pro rozpoznání objektu, a poté sledovat pohyb objektu v obraze. Předpokladem těchto metod je, že objekt zůstává po celou dobu videosekvence v zorném poli kamery a není překryt jiným objektem.

Pro sledování již nalezeného objektu je možné použít algoritmy pro výpočet optického toku. Optický tok určí rozdíly mezi dvěma po sobě jdoucími snímky videosekvence a pro každý bod stanoví vektor jeho pohybu. Tuto informaci lze agregovat pro všechny body detekovaného objektu a určit pohyb celého objektu mezi oběma snímky. Často používaným algoritmem pro výpočet optického toku je algoritmus Lucas-Kanade [Lucas et al., 1981], využít však lze i algoritmy pro detekci disparit v obraze [Scharstein et al., 2002]. Výhodou použití těchto algoritmů je běh v reálném čase a možnost paralelizace. Algoritmy je možné zaměřit pouze na body, které byly součástí objektu v předchozím snímku, a ještě více tak redukovat výpočetní náročnost.

Nevýhodou tohoto přístupu je, že pro modelování optického toku není využita informace o struktuře objektu, algoritmy založené na Lucas-Kanade sledují pouze lokální okolí bodu, pro který je optický tok počítán.

3.6.3 On-line learning

On-line learning je rozšíření metod strojového učení pro detekci objektu ve statickém snímku o časovou dimenzi. Cílem je postupně upravovat parametry klasifikačního algoritmu na základě úspěšných detekcí objektu během zpracovávání videosekvence. Pokud je ve videosnímku detekován objekt, je ze snímku vyříznut a použit zpětně jako část trénovací sady pro inkrementální úpravu parametrů klasifikačního algoritmu. Algoritmus se tak postupně učí novým podobám objektu tak, jak se vyskytují ve videu.

Výhodou on-line learningu je možnost jednoduchého propojení s již existujícím algoritmem a znovupoužití trénovací sady a také to, že sledování je přizpůsobeno přímo danému objektu, je tedy možné výrazně omezit míru falešných detekcí. Na rozdíl od sledování pohybu pomocí optického toku není nutné aplikovat předpoklad, že je objekt vždy v zorném poli kamery – klasifikátor je schopen navíc určit i přítomnost či nepřítomnost objektu. Oproti jednodušším metodám je však on-line learning výpočetně náročnější, protože je s každým novým snímkem nutné provést klasifikaci a aktualizovat parametry.

Tracking-learning detection (TLD) [Kalal et al., 2009] je příkladem on-line learning algoritmu, který je schopen detekce v téměř reálném čase a je robustní vůči změnám v podobě objektu, změně velikosti, perspektivním zkreslením a změnám v globálním osvětlení scény. Pro zpřesnění pozice sledovaného

objektu používá TLD algoritmus doplnění v podobě posuvných šablon a Lucas-Kanade sledování [Kalal et al., 2010a]. Z TLD dále vychází metoda CMT [Nebehay et al., 2015], která v objektu najde význačné body, a ty poté sleduje.

3.6.4 Filtrování

Pojem filtrování shrnuje sadu příbuzných algoritmů, které využívají sérii po sobě jdoucích pozorování pro odhad hodnot náhodných proměnných. Jednotlivá pozorování mohou být zatížena šumem, cílem filtrování je šum eliminovat a odhadnout hodnoty proměnných co nejpřesněji. V případě problému detekce objektu ve videosekvenci jsou hledanými proměnnými souřadnice středu objektu, případně jeho rozměry. Nejčastěji používanými filtry jsou Kalmanův filtr a částicový filtr.

3.6.4.1 Kalmanův filtr

Kalmanův filtr [Kalman et al., 1960] reprezentuje stav proměnných a jejich potenciální hodnoty pomocí vícerozměrného Gaussova rozdělení. Cílem filtrování je co nejvíce snížit rozptyl tohoto rozdělení, a tím soustředit odhad hodnot do co nejmenší oblasti prostoru všech možných hodnot. Kalmanův filtr provádí odhad stavu hodnot sledovaných proměnných ve dvou krocích. V prvním kroku je z aktuálního stavu vypočítán odhad následujícího stavu včetně odhadu horní hranice množství šumu, který provádí externí algoritmus – např. pomocí fyzikální simulace. Výsledek dohadu je poté zkombinován s následujícím měřením stavu, které je rovněž zatíženo šumem. Pokud predikovaný a měřený stav mají podobné hodnoty, pak se snižuje celkový rozptyl (je pravděpodobné, že se objekt v daném místě opravdu nachází), naopak pokud se predikovaný a měřený stav rozcházejí, celkový rozptyl se zvyšuje.

Kalmanův filtr je unimodální, stav proměnných je reprezentován pouze jedním Gaussovským rozdělením. Pro detekci objektů ve videosekvenci to znamená, že Kalmanův filtr je schopen sledovat pozici pouze jednoho objektu. V případě výskytu více objektů dojde k porušení unimodality a Gaussovské rozdělení již nebude dobrým modelem pro pozici objektu.

3.6.4.2 Částicový filtr

Základní princip fungování částicového filtru [Del Moral, 1996] je shodný s fungováním Kalmanova filtru. Cílem je stejně jako u Kalmanova filtru ze sady měření stavu a odhadů následujícího stavu sestavit pravděpodobnostní model reprezentující hodnoty sledovaných proměnných. Namísto Gaussova rozdělení je však odhad hodnot reprezentován množinou částic. V případě detekce objektů se jedná o dvourozměrné částice, kde každá částice reprezentuje potenciální výskyt objektu v daném místě. Jedná se tak o diskretizovaný odhad pravděpodobnostního rozdělení namísto aproximace existujícím rozdělením. Podobně jako u Kalmanova filtru je algoritmus výpočtu následujícího odhadu

stavu rozdělen na dvě části – odhad budoucí pozice jednotlivých částic pomocí externího algoritmu a filtrování málo pravděpodobných částic podle měření nového stavu. Aby nedocházelo k přílišné redukci počtu částic v průběhu sledování, jsou po každém kroku vytvořeny nové částice buď duplikací existujících částic, které mají vysokou pravděpodobnost. Postupně jsou tak nahrazovány málo pravděpodobné částice pravděpodobnějšími a dochází ke koncentraci částic v okolí pravděpodobné pozice.

Vzhledem k tomu, že se částice mohou shromažďovat do více shluků, může oproti Kalmanovu filtru částicový filtr reprezentovat i multimodální rozdělení, kdy je současně sledováno více objektů zároveň.

3.6.5 Sledování objektů pomocí neuronových sítí

Často používaným přístupem pro detekci objektů ve videosekvenci je použití detekční sítě na jednotlivé snímky videosekvence, které díky vysoké přesnosti detekce objektu ve statickém obraze dosahují dobrých výsledků i při takto jednoduchém použití.

3.6.5.1 GOTURN

GOTURN [Held et al., 2016] využívá znalosti ohraničujícího obdélníku objektu v předchozím snímku a předpovídá pozici a velikost ohraničujícího obdélníku v aktuálním snímku. Prediktor ohraničujícího obdélníku je hluboká neuronová síť, jejímž vstupem je výřez z předchozího snímku a výřez aktuálního snímku, ve kterém se objekt pravděpodobně nachází. Systém je tak schopen sledovat jeden objekt, který se plynule pohybuje v zabírané scéně. Výhodou systému je vysoký výkon (> 100 FPS) a generalizace, kdy systém je schopen detekovat objekt i bez toho, aby byl přítomen v trénovací sadě. Nevýhodou GOTURN je obtížné znovunalezení objektu ve chvíli, kdy je chvíli mimo záběr kamery či je během sledování ztracen.

3.6.5.2 OnAVOS

OnAVOS [Voigtlaender et al., 2017] využívají předtrénované sítě pro obecnou segmentaci objektů v obraze, která je v průběhu predikce postupně trénována na konkrétní objekty v dané videosekvenci. Díky tomu dosahuje vysoké přesnosti detekce a je v době psaní práce nejpřesnější metodou pro segmentaci objektů ve videosekvenci podle testů na datové sadě DAVIS [Pont-Tuset et al., 2017; Perazzi et al., 2016].

Obrázek 14: Ukázka segmentovaného snímku z videosekvence pomocí metody OnAVOS. Obrázek převzat z [Voigtlaender et al., 2017].

3.7 Existující datové sady

Pro kvalitní detekci objektů v obraze či videosekvenci je potřeba velké množství příkladů, ze kterých se mohou detekční algoritmy učit podobu objektů. Následující podkapitoly stručně popisují několik nejvýznamnějších existujících datových sad.

3.7.1 ImageNet

ImageNet [Russakovsky et al., 2014b] je velmi rozsáhlá datová sada, která v současnosti obsahuje 14 197 122 obrázků s anotacemi pro 21 841 různých kategorií objektů. Jedná se tak o nejrozsáhlejší veřejně dostupnou datovou sadu pro rozpoznání objektů v obraze. Velká část obsahuje i anotované ohraničující obdélníky okolo přítomných objektů. Od roku 2015 je její součástí i datová sada obsahující videosekvence. Většina existujících neuronových sítí je nejprve předtrénována na této velké datové sadě, a teprve následně dotrénována na menší, cílové sadě.

3.7.2 Microsoft COCO

Microsoft Common Objects in Context (COCO) [Lin et al., 2014] obsahuje 60 000 obrázků a 80 kategorií objektů, pro které jsou k dispozici nejen ohraničující obdélníky, ale i přesné segmentační masky a klíčové body lidských postav. Jedná se proto o datovou sadu používanou primárně pro učení segmentačních algoritmů.

3.7.3 Pascal VOC

Pascal Visual Object Classes (VOC) [Everingham et al., 2010] je již nerozvíjená datová sada obsahující 11 530 obrázků a 20 různých kategorií, pro které jsou k dispozici ohraničující obdélníky. Pro část sady je k dispozici též segmentační maska objektů, díky relativně nízkému počtu kategorií a obrázků oproti MS COCO je obvykle používána pro dotrénování algoritmu natrénovaného na některé z větších datových sad.

3.7.4 DAVIS

Datová sada DAVIS [Pont-Tuset et al., 2017] obsahuje 90 krátkých videí ve Full HD rozlišení, kde každý snímek videa je anotován přesnou segmentační maskou. Oproti výše zmíněným datovým sadám tak jde primárně o datovou sadu zaměřenou na přesné sledování objektů v čase.

3.7.5 YouTube Bounding Boxes

YouTube Bounding Boxes (BB) [Real et al., 2017] je rozsáhlá datová sada společnosti Google obsahující 240 000 videí z YouTube anotovaných do 23 kategorií. Pro každý snímek videa je k dispozici ohraničující obdélník okolo objektu. Nevýhodou této datové sady je, že ke stažení jsou pouze odkazy na videa a anotace, díky čemuž mnoho videí může být v budoucnosti mnoho videí neodstupných.

4 Dosažené výsledky

Problém detekce objektů ve videosekvenci lze rozdělit na několik dílčích podproblémů – sběr dat a vytvoření odpovídající datové sady, extrakci atributů, detekci a klasifikaci objektů a jejich sledování ve videosekvenci (viz kap. 3.1). V rámci výzkumu bylo dosaženo výsledků ve všech zmiňovaných oblastech.

V oblasti sběru dat byla vytvořena vlastní datová sada zaměřující se na detekci hlav laboratorních potkanů pohybujících se ve volném poli. Cílem této datové sady je otestovat robustnost algoritmu vůči nevhodným světelným podmínkám a různému umístění kamer, což jsou jedny z nejčastějších zkreslení v praktických nasazeních.

V návaznosti na vytvořenou datovou sadu byl navržen systém pro detekci objektů ve videosekvenci založený na kombinaci konvolučních neuronových sítí, optického toku a modifikovaného částicového filtru.

V oblasti extrakce atributů byly analyzovány binární vzory Local Binary Patterns založené na lokálních rozdílech v kontrastu a navrženo jejich rozšíření, které umožňuje snazší propojení s algoritmy strojového učení [Petránek et al., 2012]. Zároveň byly zkoumány možnosti segmentace obrazu a její akcelerace za účelem redukce množství vstupních dat do učících algoritmů. Jedním ze zkoumaných směrů bylo i rozšíření vstupních dat o hloubkovou informaci na základě snímání ze dvou kamer [Petránek et al., 2010].

V kapitole 4.4 je popsán systém pro segmentaci objektů v obraze a možnosti jeho použití na videosekvence.

4.1 Návrh datové sady

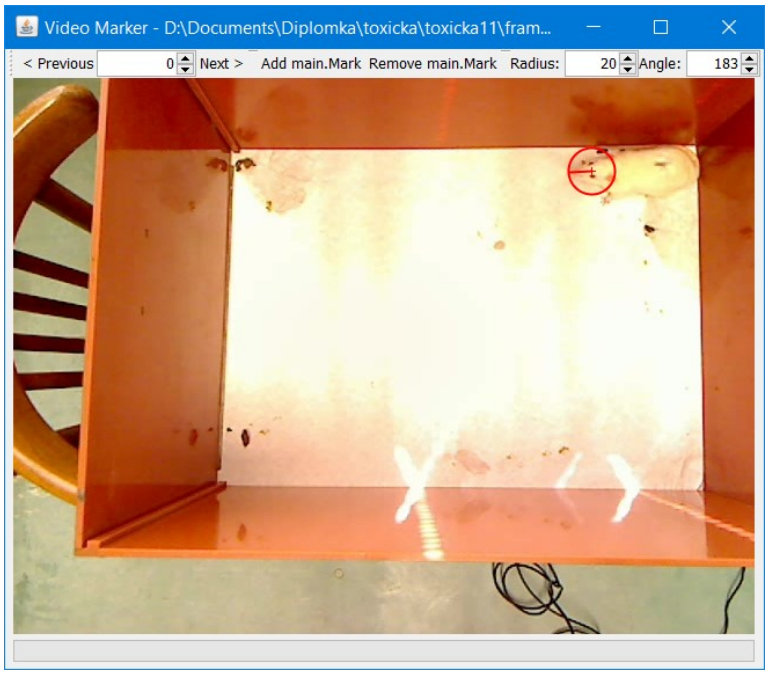
Ve spolupráci s Katedrou toxikologie Fakulty vojenského zdravotnictví Univerzity obrany byla v rámci práce vytvořena datová sada pro detekci pohybu potkana ve volném poli. V současnosti je pohyb potkana monitorován laborantem bez asistence výpočetní techniky, což je velmi časově náročná činnost.

V rámci spolupráce bylo vytvořeno celkem 12 přibližně sedmiminutových videí s rozlišením 640×480 pixelů. Videa monitorují pohyb potkana v tzv. volném poli – prázdném boxu se savým papírem na dně, ve kterém má potkan možnost volného pohybu. Cílem zpracování videí je detekovat pozici hlavy potkana v průběhu celé videosekvence, aby bylo možné sestavit trajektorii pohybu zvířete pro další analýzu.

Stanovený problém a zvolená snímací technika vedou ke vzniku celé řady zkreslení, která dělají detekci hlavy potkana netriviální:

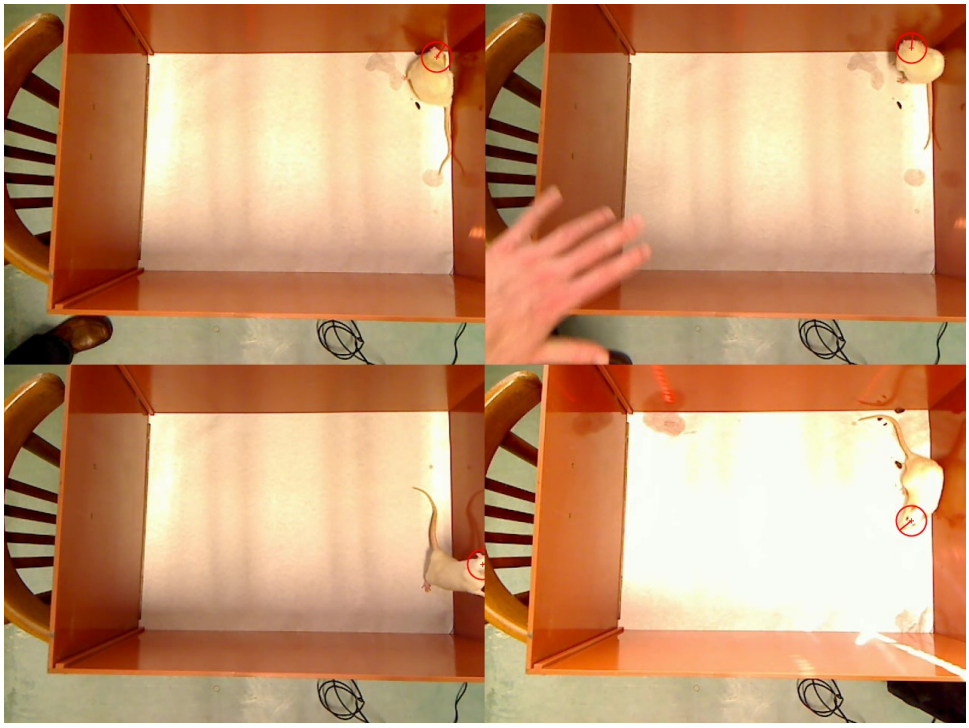
- Potkan má možnost volného pohybu, hlava se tak může vyskytovat v libovolném místě obrazu a být libovolně natočena či skryta.
- Videa jsou zabírána z různých úhlů, simulujících reálné nasazení.
- Při prudkém pohybu potkana dochází k rozmazání hlavy vlivem délky expozice.
- V záběru může probíhat nesouvisející pohyb osob či vržených stínů, který by neměl mít vliv na detekci.
- Během snímání může docházet ke změně intenzity okolního osvětlení, a tím např. přesvětlení či přílišnému ztmavení jednotlivých snímků.
- Savý papír na dně volného pole má bílou barvu, mezi hlavou potkana a pozadím je tak velmi nízký kontrast.
- Rozlišení videa je pouze 640×480 pixelů, hlava potkana má na snímcích velikost přibližně 15×15 pixelů. Výsledkem je velmi řídká datová sada, kdy pouze malé procento pixelů obsahuje hledaný objekt.
- Sledovaný objekt je často stacionární, je proto problém s použitím technik pro modelování pozadí (viz kapitola 3.6.1).

Pro účely anotování byl vyvinut speciální anotační program, který umožňuje rychlé označení objektů na snímku a jednoduchý pohyb mezi snímky videa (viz Obrázek 15). Kromě označení pozice umožňuje anotační program specifikovat i natočení objektu, což může být využito pro zarovnání objektu do výchozí polohy pro trénování či prohlížení datové sady.



Obrázek 15: Ukázka anotačního programu pro vytvoření datové sady s příkladem snímku z anotovaného videa.

Celkem bylo zpracováno a označeno 10 000 snímků. U videí byl označován každý pátý snímek, aby byl rozdíl pohybu dostatečně znatelný a nevzniklo příliš mnoho skoro stejných anotací. Obrázek 16 ukazuje příklady anotovaných snímků z vytvořené sady.



Obrázek 16: Ukázka anotovaných snímků z vytvořené datové sady. Jednotlivé snímky demonstrují obtížnost detekce díky různému natočení, okluzi a obtížným světelným podmínkám.

4.2 Extrakce atributů pomocí Local Binary Patterns

V návaznosti na analýzu obrazu z vlastní datové sady byl zkoumán binární atribut Local Binary Patterns (LBP), viz kapitola 3.3.5. Výhodou LBP je nezávislost výsledných atributů na otočení, posunutí, měřítku, jasu a kontrastu analyzovaného obrazu [Ojala et al., 2002], což jsou žádoucí vlastnosti především pro analýzu datové sady navržené v předchozí kapitole.

Nevýhodou uniformních LBP vzorů je zařazení zhruba 20 % všech vzorů do společné kategorie. Díky tomu dochází ke ztrátě informace, která může být důležitá pro navazující algoritmy v analýze obrazu. Ačkoliv existují řešení tohoto problému, tato řešení jsou obvykle zaměřena na jednu konkrétní aplikační oblast a jsou implementačně komplikovaná.

Cílem výzkumu v této oblasti bylo ověřit možnost využití LBP jako vstupních atributů pro neuronové sítě. Protože LBP je binární operátor produkující sadu bitů, je spojení s neuronovými sítěmi, jejichž vstupy jsou vektory reálných čísel, netriviální úlohou. Jednoduchý přístup – převést jednotlivé 0 a 1 na desetinné 0 a 1 a použít je jako vstupy – je datově velmi neefektivní. Například již základní 8bitové LBP znamená 8 vstupních kanálů do neuronové sítě namísto původního jednoho kanálu šedotónového obrazu. Uniformní LBP vzory datové náročnosti také nepomáhají, neboť se jedná o kategorickou veličinu s $p + 2$ kategoriemi (kde p je počet bitů LBP), je tedy potřeba ji reprezentovat pomocí $p + 2$ vstupních vrstev neuronů.

4.2.1 Návrh řešení

Aby bylo možné LBP efektivně použít spolu s neuronovými sítěmi, je nutné opustit reprezentaci LBP jako bitového řetězce či kategorické hodnoty.

Navržené řešení namísto přímé reprezentace vyjadřuje rotačně invariantní vzory pomocí počtu přechodů mezi 0 a 1 bity (T) a počtu nulových bitů ve vzoru (Z). Nastavením prahu pro tyto počty je možné volit mezi množstvím zachycené informace a počtem vzorů. Omezením T na hodnotu 2 je dosažena ekvivalence s uniformními vzory dle [Ojala et al., 2002].

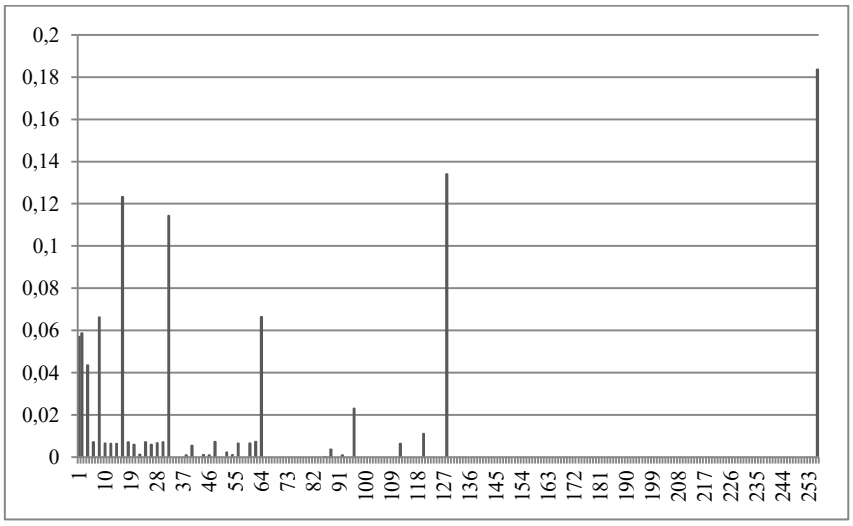
Protože počty Z a T jsou poměrové proměnné, lze je snáze použít v algoritmech strojového učení – namísto jednoho binárního vstupního atributu pro každý vzor postačují dva celočíselné atributy pro všechny vzory. Vydělením Z a T jejich maximálními hodnotami lze získat reálné proměnné v intervalu [0, 1]. Touto jednoduchou transformací je dosaženo výrazné redukce počtu vstupů, přičemž je překonána informační hodnota uniformních LBP.

Pouhé použití veličin Z a T pro reprezentaci LBP vzoru vede ke ztrátě informace – počet p-bitových LBP vzorů je $O(2^p)$, zatímco hodnoty Z a T nabývají hodnot od 0 do P, respektive $\frac{P}{2}$, jsou tak schopny

reprezentovat pouze $O(P^2)$ vzorů. Přestože část informace je ztracena, transformovaný vzor i nadále udržuje informaci o relativní intenzitě bodů v okolí (počet nulových bitů) a o stabilitě gradientu (počet přechodů). Z a T jsou též přirozeně rotačně invariantní, což zrychluje výpočet, protože není nutné rotačně invariantní variantu vzoru dohledávat v tabulce či opakovaně počítat bitový posun.

4.2.2 Analýza řešení

Navržené řešení bylo testováno na obrázcích z testovací sady ILSVRC 2014, která obsahuje 100 000 obrázků z datové sady ImageNet [Russakovsky et al., 2014b]. Nejprve bylo ověřeno, že tvrzení z [Ojala et al., 2002], že uniformní vzory tvoří 80 % všech vzorů, je platné i pro vybranou datovou sadu. Obrázek 17 ukazuje rozložení relativních četností 8bitových LBP vzorů o poloměru 1 v testovací sadě. Uniformní vzory tvoří celkem 84,6 % všech vzorů, což odpovídá tvrzení z [Ojala et al., 2002]. Podobných výsledků bylo dosaženo pro 16bitové vzory s poloměrem 4 pixely, kde uniformní vzory tvoří 54,6 %, což odpovídá 57,6 % uvedeným v [Ojala et al., 2002].

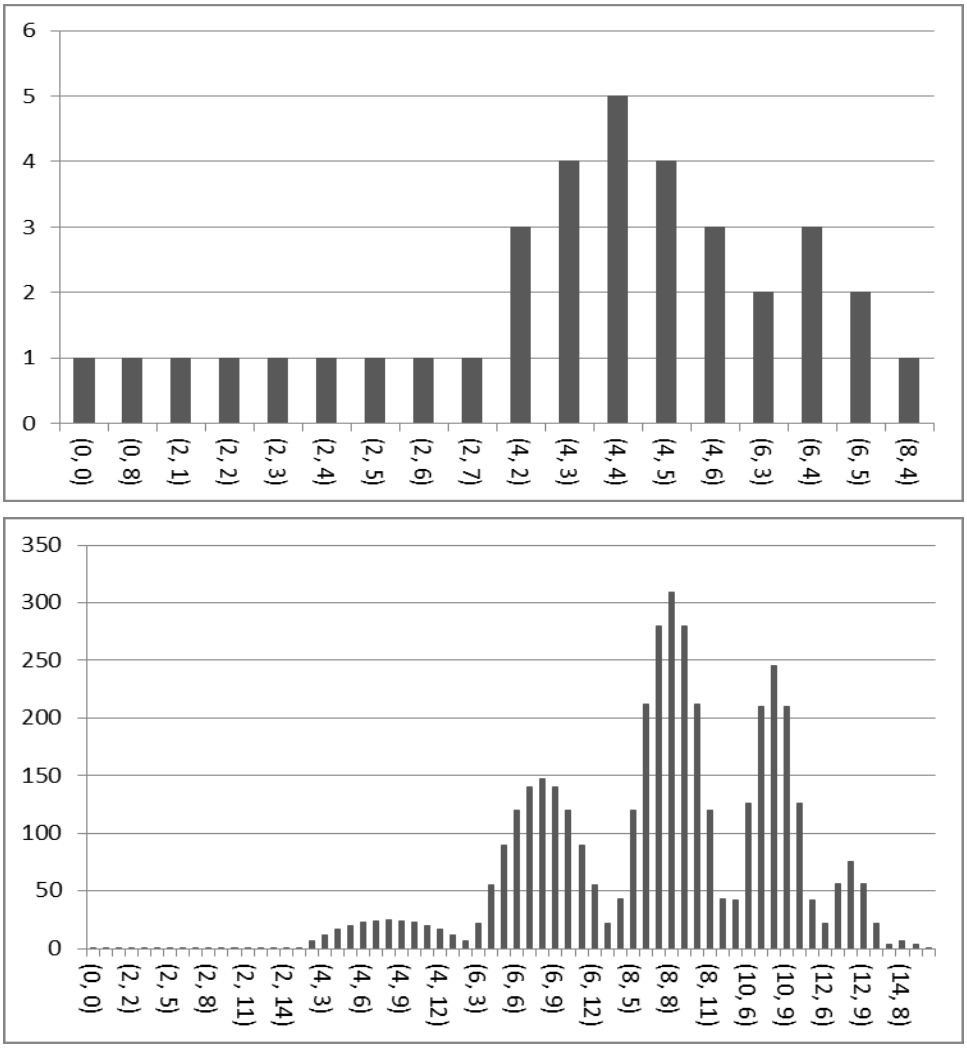


Obrázek 17: Relativní četnosti 8bitových rotačně invariantních LBP v testovací sadě ILSVRC 2014, která obsahuje 100 000 fotografií a obrázků.

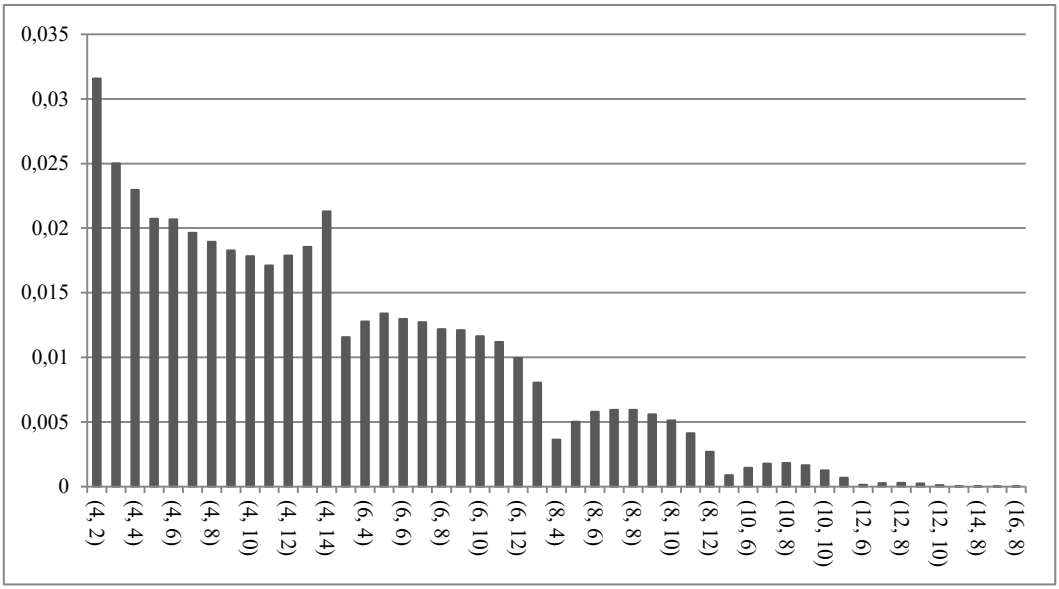
Protože navržená Z, T transformace LBP je ztrátová, byla analyzována ztráta informace v důsledku této transformace a porovnána se ztrátou informace v případě uniformních LBP vzorů. Obrázek 18 ukazuje aliasování jednotlivých vzorů. Čím více Z, T kombinací odpovídá danému vzoru, tím vyšší je sloupec daného LBP vzoru. Pokud by transformace byla bezztrátová, všechny sloupce by obsahovaly právě jeden element.

Sloupce odpovídající uniformním LBP mají pouze jednu odpovídající Z, T transformaci, tyto vzory jsou tak efektivně zakódovány bez ztráty informace. Navržené řešení rovněž efektivně reprezentuje neuniformní vzory, kdy vzory s menším počtem přechodů (T), jež se v obraze vyskytují častěji, jsou

aliasovány výrazně méně než šumové vzory s mnoha přechody, obzvláště pak u LBP s větším počtem bitů (viz Obrázek 18 dole a Obrázek 19).



Obrázek 18: Počty rotačně invariantních vzorů pro dané T, Z kombinace pro 8bitové (nahoře) a 16bitové (dole) vzory. Ke ztrátě informace dochází především u vzorů s mnoha přechody, které se v obraze vyskytují zřídka. Popisy některých sloupců u spodního grafu byly pro přehlednost vynechány.



Obrázek 19: Četnosti výskytu T, Z kombinací v testovací sadě ILSVRC 2014 pro 16bitové vzory. Vzory s méně přechody (T) se vyskytují výrazně méně často. Popisy některých sloupců grafu byly pro přehlednost vynechány.

4.2.3 Propojení s neuronovými sítěmi

V rámci testování navrženého řešení byly aplikovány LBP, rotačně invariantní LBP, uniformní LBP a navržená metoda na dvě standardní datové sady, MNIST [Lecun et al., 2017] a CIFAR-10 [Krizhevsky et al., 2009] převedené do šedotónu. Tyto datové sady byly vybrány pro datovou nenáročnost, obrazy v těchto sadách jsou dostatečně malé na to, aby bylo možné vstupy do neuronové sítě reprezentovat jako rozložené vektory nul a jedniček. Datové sady transformované jednotlivými metodami byly použity jako vstupy do state-of-the-art neuronových sítí pro klasifikaci MNIST a CIFAR-10. Přesnost klasifikace a doba trénování je uvedena v tabulkách. Pro experimenty byl vybrán framework Keras s backendem TensorFlow díky existujícím definicím architektur neuronových sítí pro MNIST i CIFAR-10.

Celkem byly provedeny 4 experimenty se 3 různými architekturami neuronových sítí. Nejprve byla na obě datové sady použita jednoduchá dopředná neuronová síť s dvěma skrytými vrstvami, následně byla použita konvoluční síť se 4 skrytými vrstvami pro MNIST a hlubší konvoluční síť se 6 skrytými vrstvami pro CIFAR-10. Konvoluční sítě jsou založeny na Keras modelech dostupných v rámci frameworku [Chollet, 2016b; Chollet, 2016d; Chollet, 2016c; Chollet, 2016a]. Architektury sítí a parametry trénování jsou shrnuty v Tabulka 1.

Tabulka 1: Architektury použitých sítí a parametry trénování

	Dopředná	CNN-MNIST	CNN-CIFAR-10
Architektura[1]	VV, PP(512), DO(20 %), PP(512), DO(20 %), PP(10)	VV, KV(3×3), KV(3×3), MP(2×2), DO(25 %), PP(128), DO(50 %), PP(10)	VV, KV(3×3), MP(2×2), DO(25 %), KV(3×3), KV(3×3), MP(2×2), DO(25 %), PP(512), DO(50 %), PP(10)
Aktivační funkce	ReLU	ReLU	ReLU
Aktivační funkce výstupů + chybová funkce	Softmax + kategorická křížová entropie	Softmax + kategorická křížová entropie	Softmax + kategorická křížová entropie
Optimalizační algoritmus	RMSProp	AdaDelta	Nesterov SGD[2] + momentum, koeficient učení 0.01, momentum 0.9 a koeficient L2 normalizace vah 10^{-6}
Epochy	15	15	15

Sítě byly trénovány na přenosném počítači s GPU nVidia GTX 660M (2 GB VRAM) a i7-3610QM procesorem, bylo využito GPU akcelerované trénování.

Tabulka 2 porovnává jednotlivé metody na datové sadě MNIST v kombinaci s jednoduchou dopřednou neuronovou sítí. Všechny metody mají podobnou přesnost, avšak navržená metoda je natrénována za výrazně kratší dobu díky nízké díky tomu, že vstup je pouze dvourozměrný.

Tabulka 3 porovnává dobu trénování a přesnost s použitím konvoluční neuronové sítě na datové sadě MNIST. Navržená metoda má o něco vyšší přesnost než uniformní LBP, zaostává však za jednoduchými LBP variantami, které obsahují více informací, které je schopna konvoluční síť oproti

[1] Model tvoří sada vrstev zleva doprava (VV – vstupní vrstva, KV – konvoluční vrstva, PP – plně propojená vrstva, MP – max pooling vrstva, DO – drop-out vrstva

[2] Stochastic Gradient Descent

jednoduché dopředné síti efektivně zpracovat. Dobu trénování ovlivňuje především složitost vnitřních konvolučních vrstev, i tak je však navržená metoda v trénování nejrychlejší.

Tabulka 4 porovnává testované metody na datové sadě CIFAR-10 v kombinaci s jednoduchou dopřednou neuronovou sítí. Ačkoliv uniformní LBP mají nejvyšší přesnost, jedná se o artefakt inicializace počátečních parametrů sítě – v průběhu trénování tato síť divergovala z úvodních 33,3 % po první epoše na konečných 31,54 %. Navržená metoda byla jediná, která vykazovala učení (z 9,98 % na 18,67 %).

Tabulka 5 porovnává jednotlivé metody na datové sadě CIFAR-10 v kombinaci s konvoluční neuronovou sítí. Navržená metoda má vyšší přesnost než ostatní LBP varianty, protože jednoduchost výstupu navrhované metody usnadňuje trénování a v porovnání s ostatními metodami tím vyváží ztrátu informace. Doba trénování je opět ovlivněna především složitými vnitřními vrstvami, ale navržená metoda se přesto trénuje o něco rychleji.

4.2.4 Shrnutí

Přestože výsledky potvrzují hypotézu, že trénování s navrženou metodou bude přesnější a zabere méně času, výsledky ukazují, že použití LBP v kombinaci s neuronovými sítěmi pro detekci složitějších objektů je problematické. State-of-the-art metody pro klasifikaci CIFAR-10 pomocí konvolučních neuronových sítí, jež mají na vstupu přímo bitmapová data obrazu, dosahují přesnosti až 95 % [Springenberg et al., 2015]. Navrženou metodu lze proto snadno využít tam, kde se v současnosti využívají LBP, především pro klasifikaci a segmentaci textur, přičemž lze počítat s datovou i výpočetní úsporou. V kombinaci se složitějšími konvolučními neuronovými sítěmi je vhodnější využít jako vstup přímo bitmapový obraz, jelikož nedochází ke ztrátě informace a neuronová síť je schopna se vhodné atributy naučit sama.

Tabulka 2: Porovnání doby trénování a přesnosti s použitím jednoduché dopředné sítě a datové sady MNIST. Navržená metoda má podobnou přesnost jako ostatní varianty LBP, ale trénuje se mnohem rychleji.

MNIST + dopředná	Přesnost (test. sada)	Doba trénování (s)
LBP	98.27 %	238
LBP (rot. invariantní)	98.07 %	236
LBP (uniformní)	97.25 %	256
Navržená metoda	97.92 %	148

Tabulka 3: Porovnání přesnosti a doby trénování s CNN-MNIST sítí a MNIST datovou sadou.

MNIST + CNN-MNIST	Přesnost (test. sada)	Doba trénování (s)
LBP	98.17 %	1570
LBP (rot. invariantní)	98.07 %	1569
LBP (uniformní)	96.91 %	1583
Navržená metoda	97.06 %	1524

Tabulka 4: Porovnání přesnosti a doby trénování s použitím jednoduché dopředné sítě a datové sady CIFAR-10.

CIFAR-10 + dopředná	Přesnost (test. sada)	Doba trénování (s)
LBP	10.00 %	229
LBP (rot. invariantní)	10.00 %	231
LBP (uniformní)	31.54 %	274
Navržená metoda	18.67 %	129

Tabulka 5: Porovnání přesnosti a doby trénování s použitím konvoluční sítě a datové sady CIFAR-10.

CIFAR-10 + CNN-CIFAR-10	Přesnost (test. sada)	Doba trénování (s)
LBP	30.4 %	2850
LBP (rot. invariantní)	28.62 %	2850
LBP (uniformní)	29.68 %	2866
Navržená metoda	38.38 %	2805

4.3 Sledování objektů ve videosekvenci pomocí konvolučních sítí a částicových filtrů

Detekci objektu ve videosekvenci je možné rozdělit na detekci objektu ve statickém snímku a následně sledování objektu v postupující videosekvenci. Mezi řešené problémy patří především měnící se podoba objektu, střihy ve videu, vystoupení objektu ze záběru či změny zabírané scény (pohyby nesouvisejících objektů, změna osvětlení, okluze apod.).

Z testování existujících detektorů na navržené datové sadě (viz kap. 4.1) bylo zjištěno, že žádný z běžně dostupných detektorů nedosahuje dostatečné přesnosti detekce. Byl proto navržen vlastní systém, který je schopen přesnější detekce na navržené datové sadě. Ačkoliv byl systém primárně vyvinut pro detekci na vlastní datové sadě, cílem návrhu byla i dostatečná obecnost – pro detekci jiných objektů stačí vyměnit datovou sadu pro trénování.

4.3.1 Návrh řešení

Systém byl rozdělen na tři dílčí moduly – modul pro detekci objektu ve statickém snímku, modul predikující pohyb objektu mezi snímky a modul zajišťující plynulé sledování objektu. Cílem detekce objektu ve statickém obraze je sestavení pravděpodobnostní mapy výskytu objektu (či více instancí stejného typu objektu), která je dále použita spolu s odhadem pohybu jednotlivých bodů obrazu jako vstup do modifikovaného částicového filtru.

4.3.1.1 Detekce objektu ve statickém obraze

Cílem detekce objektu ve statickém obraze je vytvořit pravděpodobnostní mapu výskytu pro každý typ detekovaného objektu. Vstupem je tedy tříkanálový RGB obraz a výstupem je k-kanálová mapa se stejným rozměrem, jako vstupní obraz, kde k je počet různých typů detekovaných objektů.

State-of-the-art metody pro detekci objektů ve statickém obraze jsou založeny na hlubokých neuronových sítích, jejich využití se proto nabízí i na výpočet pravděpodobnostní mapy. Oproti běžným detekčním architekturám přináší výpočet pravděpodobnostní mapy několik rozdílů:

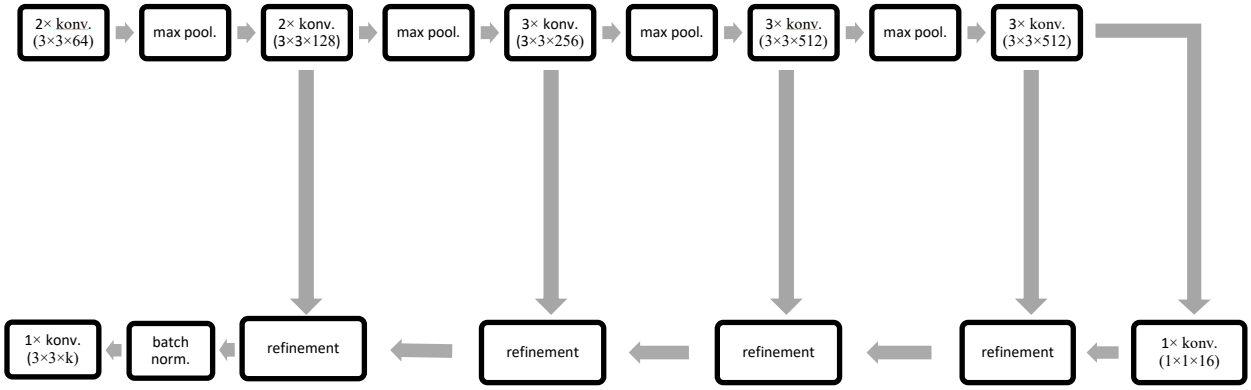
1. Běžné detekční architektury predikují buď pouze typ objektu (k výstupů), nebo ohraničující obdélníky. Výstupem tak není per-pixel mapa pravděpodobností, ale datově mnohem méně náročné vektory. Je proto nutné vyřešit problém s postupným snižováním rozlišení u hlubších vrstev sítě, ke kterému dochází vlivem pooling vrstev.

2. Výsledná pravděpodobnostní mapa je velmi řídká – většina pixelů obrazu obvykle nenáleží sledovanému objektu. V ideálním případě je pouze jeden pixel v místě středu objektu roven 1, ostatní jsou 0. To přináší problém při trénování, protože jsou počty pozitivních a negativních příkladů velmi nevyvážené.

Problém postupného snižování rozlišení byl adresován využitím modifikované architektury SharpMask (viz kapitola 3.5.9.3, [Pinheiro et al., 2016]), která byla primárně navržena pro segmentaci objektů v obraze, jejím výstupem je proto maska se stejným rozlišením, jako má vstupní obraz. Jako základ sítě je použita detekční síť VGG-16 předtrénovaná na datové sadě ImageNet. Problém postupného snižování rozlišení je řešen přidáním dodatečných konvolučních vrstev, které postupně opět zvyšují rozlišení a mají dostatečně nízký počet konvolučních filtrů, aby byla zachována paměťová a výpočetní úspornost. Výsledkem je architektura sítě zobrazená na Obrázek 20, kde původní VGG-16 vrstvy postupně zvyšují abstrakci a snižují rozlišení, zatímco tzv. refinement moduly postupně abstraktní vrstvy transformují zpět na výslednou pravděpodobnostní mapu s plným rozlišením.

Refinement modul je sada několika konvolučních vrstev zakončených fixní vrstvou, jež zdvojnásobí rozlišení pomocí bilinární interpolace. Do refinement modulu vstupuje výstup z předchozího refinement modulu a odpovídající vrstva z VGG architektury, výstupem je vylepšený kombinovaný vstup s dvojnásobným rozlišením. Architektura tohoto modulu byla převzata z architektury SharpMask, pouze byl snížen počet konvolučních filtrů z 32 na 16. Nákres architektury refinement modulu znázorňuje Obrázek 13.

Oproti původní SharpMask architektuře je výstup posledního refinement modulu normalizován pomocí batch normalization vrstvy, což usnadňuje učení poslední konvoluční vrstvy, jejímž výstupem jsou hodnoty od 0 do 1. Všechny konvoluční vrstvy v celé architektuře kromě výstupní využívají rektifikovanou lineární aktivační funkci, výstupní vrstva používá sigmoid aktivaci, aby byl zaručen výstup v intervalu od 0 do 1.



Obrázek 20: Ukázka celkové architektury sítě pro výpočet pravděpodobnostní mapy. Každá max pooling vrstva snižuje rozlišení na polovinu a každý refinement modul rozlišení zvyšuje na dvojnásobek. Vrchní část představuje původní VGG architekturu, spodní část s refinement moduly poté rozšíření založené na SharpMask.

Problém příliš řídké pravděpodobnostní mapy byl vyřešen kombinací třech dílčích přístupů. Nejprve byly pro VGG část sítě použity předtrénované váhy z ImageNetu, což usnadňuje propagaci relevantní informace o objektech do zbylých částí sítě. Dále byla během trénování upravena cílová pravděpodobnostní mapa tak, aby pravděpodobnost klesala lineárně od středu ke kraji objektu. Tím je zvýšen počet nenulových bodů v obraze, avšak stále je snadné nalézt pozici objektu nalezením souřadnic bodu s nejvyšší pravděpodobností. Nakonec byla síť po první epochu[3] natrénována se softmax aktivační funkcí ve výstupní vrstvě namísto sigmoid. Softmax zaručuje, že součet všech výstupů sítě bude roven 1. Síť se tak nemůže naučit předpovídat pouze nulový výstup. Po první epoše byla aktivační funkce změněna zpět na sigmoid, která se již natrénuje správně díky vhodně předtrénovaným vahám z první epochy.

4.3.1.2 Detekce pohybu pomocí částicového filtru a optického toku

Díky tomu, že výstup detekční sítě lze interpretovat jako pravděpodobnost výskytu objektu v daném bodě, nabízí se použití Kalmanova či částicového filtru pro redukci šumu v odhadovaných pozicích objektu. Protože jedna pravděpodobnostní mapa může predikovat výskyt více objektů stejného typu, byl zvolen modifikovaný částicový filtr, jenž je schopen lépe reprezentovat multimodální rozdělení.

[3] Epocha znamená, že síť při učení prošla celou trénovací sadu.

Algoritmus detekce pohybu vychází z algoritmu pro částicový filtr:

1. **Inicializace:** Na základě pravděpodobnostní mapy prvního snímku jsou vygenerovány úvodní částice. Částice jsou vybírány ze souřadnic pravděpodobnostní mapy s opakováním, poměrově podle přiřazené pravděpodobnosti.
2. **Odhad pohybu:** V následujících snímcích je vždy vypočítán optický tok mezi předchozím a následujícím snímkem pro každou částici. K výpočtu optického toku byla použita metoda Lukas-Kanade implementovaná jako calcOpticalFlowPyrLK v knihovně OpenCV [Bradski, 2000]. Optický tok slouží jako odhad pro nové pozice částic v následujícím snímku.
3. **Resamplování:** Každé částici je přiřazena pravděpodobnost na základě pravděpodobnostní mapy aktuálního snímku. Pro každou částici je vygenerováno náhodné číslo v intervalu [0, 1] a pokud je toto číslo větší než přiřazená pravděpodobnost, je částice vyřazena. Aby počet částic zůstal konstantní, jsou namísto vyřazených částic vygenerovány nové stejným způsobem, jako v kroku 1.
4. **Výpočet nových pozic objektů:** Po resamplování je nutné analyzovat shluky částic a sloučit je do odhadu pozic objektů, které tyto shluky reprezentují. Pro tento účel byl zvolen algoritmus Mean Shift [Comaniciu et al., 2002], který metodou největšího spádu pro každou částici hledá nejbližší střed hustoty částic. Částice konvergující ke stejnému středu jsou přiřazeny do stejného shluku. Algoritmus má tak pouze jeden parametr – poloměr posuvného okna používaného pro odhad hustoty částic v daném bodě. Tento parametr ovlivňuje nejmenší možnou detekovatelnou velikost a jeho výchozí hodnota byla experimentálně stanovena na 1/20 šířky vstupního obrazu. Protože Mean Shift vždy vytvoří alespoň jeden shluk, a to i v případě, že se objekt v obraze nenachází, jsou výsledné shluky dodatečně filtrovány. Pro každý shluk je vybrána částice s maximální pravděpodobností a zjištěn celkový počet částic ve shluku – velmi málo pravděpodobné shluky s maximální pravděpodobností menší než 0,01 či počtem částic menším než 25 % z celkového počtu částic jsou odebrány. Středy zbývajících shluků představují výsledné pozice objektů.

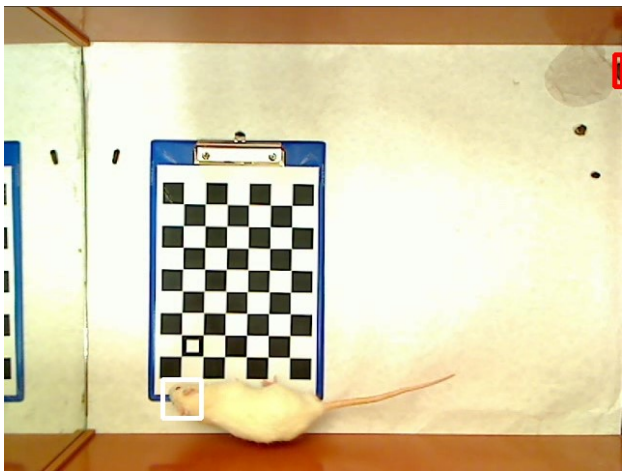
Výše uvedený algoritmus se od základního částicového filtru liší především ve třetím kroku, kdy nové částice nejsou vytvářeny výhradně z existujících, ale i na základě pravděpodobnostní mapy. Díky tomu je možné zachytit výskyt nových objektů i ve chvíli, kdy jsou všechny stávající částice soustředěny v okolí existujícího objektu.

Výhodou navrženého algoritmu je implementační jednoduchost a snadná paralelizace – všechny jednotlivé kroky lze snadno paralelizovat a v případě vysokého důrazu na výkon je lze implementovat na grafické kartě [Petránek et al., 2011].

4.3.2 Testování a porovnání s existujícími řešeními

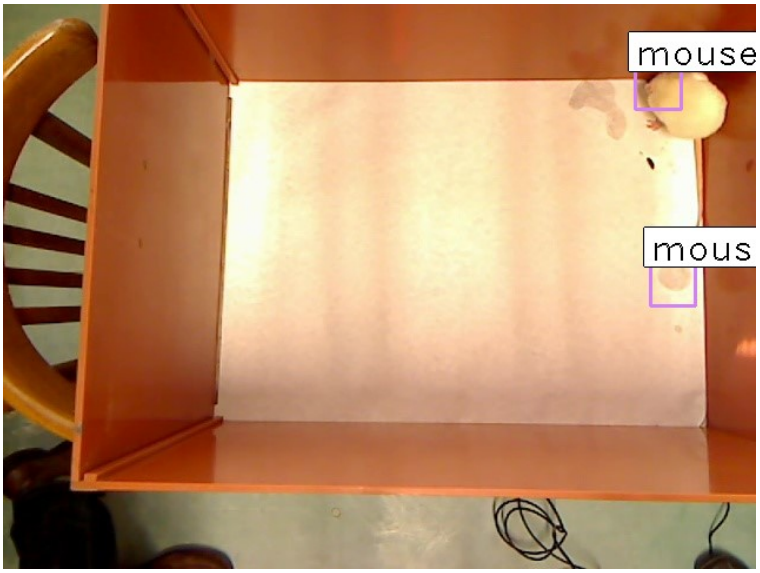
Navržený detektor byl testován na vytvořené datové sadě (viz kapitola 4.1) a výsledky detekce byly porovnány s několika existujícími řešeními – detekčními sítěmi Faster R-CNN [Ren et al., 2015] a YOLOv2 [Redmon et al., 2016b] natrénovanými na vytvořenou sadu, GOTURN [Held et al., 2016] s výchozími natrénovanými vahami a sledovací algoritmy OpenTLD [Kalal et al., 2009] a CMT [Nebehay et al., 2015]. Jednodušší řešení založená na sledování význačných bodů či regionů, jako jsou SIFT, SURF a MSER, selhávají na nízkém kontrastu mezi detekovaným objektem a pozadím (nenalézají vhodné význačné body), a proto nebyly po pilotním ověření do porovnání na vytvořené datové sadě zařazeny. Pro testování bylo použito 3061 snímků z anotovaného videa, které nebylo použito pro trénování žádné z porovnávaných metod. Zbylých 6939 anotovaných snímků z ostatních videí bylo rozděleno v poměru 80:20 na trénovací a validační sadu a použito pro trénování. Vybrané testovací video obsahuje velké množství obtížných situací – hlava potkana se dostává mimo záběr kamery, v průběhu videa se mění světelné podmínky, dochází k pohybu osob v záběru a mění se pozadí.

Během testování jednotlivých metod byla potvrzena obtížnost navržené datové sady. GOTURN bohužel hlavu potkana ztratil během několika prvních snímků a nebyl schopen do skončení videosekvence objekt znovu zachytit (viz Obrázek 21), podobně fungovaly i sledovací algoritmy OpenTLD a CMT. Výhodou je, že algoritmy GOTURN, OpenTLD a CMT nevyžadují žádné trénování, je však nutné objekt označit pomocí ohraničujícího obdélníku v prvním snímku videosekvence.



Obrázek 21: Ukázka chybného sledování objektu metodou GOTURN. Bílý rámeček představuje ruční anotaci, červený výstup GOTURN.

Detekční síť Faster-RCNN v Keras implementaci [Henon, 2017] byla trénována po dobu 80 epoch, dokud se snižovala validační chyba. Natrénovaná síť funguje výrazně lépe než přístupy založené na sledování obecných objektů, ale produkuje velké množství falešných detekcí, jak ukazuje Obrázek 22 (přesnost pouze 48,6 %, viz Tabulka 6).



Obrázek 22: Ukázka falešné detekce skvrny na podložce jako objektu pomocí metody Faster R-CNN.

Architektura YOLOv2 byla natrénována dle oficiálního návodu [Alexey, 2017] po délku 2000 iterací, dokud se snižovala chyba na validační sadě. Objekt byl označen jako detekovaný, byla-li predikovaná spolehlivost detekce větší než 50 % (0,5).

Pro navržené řešení byla trénována detekční síť pomocí optimalizačního algoritmu ADAM nejprve 1 epochu s aktivační funkcí softmax, poté po dobu 16 epoch s rychlostí učení (learning rate) 0,001 s neměnnými vahami původní VGG architektury, a poté dalších 8 epoch s rychlostí učení 0,0001 včetně trénování vah z původní VGG (tzv. finetuning). Pro sledování objektu bylo použito 1000 částic v částicovém filtru.

4.3.2.1 Vyhodnocení detekčních algoritmů

Jako míra přesnosti detekce byla zvolena metrika F-skóre, která je harmonickým průměrem veličin precision a recall, jenž se vypočítávají dle následujících vzorců:

$$precision = \frac{tp}{tp + fp}$$

$$recall = \frac{tp}{tp + fn}$$

$$F = 2\frac{precision \cdot recall}{precision + recall}$$

kde tp je počet detektorem správně detekovaných pozic (true positives), fp počet nesprávně určených umístění, kde se objekt ve skutečnosti nenachází (false positives) a fn počet detektorem neoznačených pozic, kde se objekt nachází (false negatives).

Jako true positive byla označena detekce, jejíž střed ležel do vzdálenosti 5 pixelů od manuálně anotovaného středu objektu. V případě, že detektor detekoval objekt mimo tento poloměr, byla chybná detekce započítána jak mezi false positive, tak mezi false negative.

Tabulka 6 shrnuje vyhodnocení jednotlivých testovaných metod. YOLOv2 jako jediná existující metoda dosahuje velmi kvalitních výsledků, které jsou použitelné v praxi. Jak YOLOv2, tak navržená metoda ukazují, že trénování na konkrétní datovou sadu přináší extrémní vylepšení výsledků a obecné sledování objektu bez dodatečného dotrénování ještě na obtížnější datové sady není dostatečné pro reálné nasazení.

Nevýhodou navrženého řešení je, že detekuje pouze umístění a nikoliv celý ohraničující obdélník. Tento problém lze vyřešit rozšířením pravděpodobnostní mapy o další dva kanály pro každý typ objektu, které budou předpovídat výšku a šířku objektu v daném místě a je předmětem budoucího pokračování výzkumu.

Tabulka 6: Vyhodnocení testovaných metod.

Metoda	Správná detekce[4]	Precision	Recall	F skóre
GOTURN	1	0,00032	0,00032	0,00032
OpenTLD	3	0,00191	0,00099	0,0013
CMT	35	0,011	0,012	0,012
Faster R-CNN	2651	0,486	0,882	0,626
YOLOv2	2443	0,993	0,813	0,894
Navržená metoda	2708	0,937	0,901	0,919

[4] Počet snímků, na kterých byla pozice objektu správně učena (z celkového počtu 3061 snímků)

4.4 Segmentace objektů ve videosekvenci

Segmentace objektů je nejpřesnější a zároveň nejnáročnější způsob detekce objektů. Cílem je pro každý bod obrazu určit, zda náleží nějakému objektu (1) či nikoliv (0). Výstupem tak již není pouze seznam ohraničujících obdélníků, ale bimapa ve stejném rozlišení, jako vstupní obraz.

Architektura založená na SharpMask, která byla použita v kapitole 4.3.1.1 pro generování pravděpodobnostní mapy, byla v rámci výzkumu natrénována rovněž na vytváření segmentační masky objektů ve vstupním obraze. Segmentační maska je binární mapa, kde každý bod udává, zda se v daném místě nachází objekt či nikoliv.

Cílem výzkumu bylo otestovat schopnost generalizace segmentační sítě a ověřit, zda modifikovaná segmentační síť SharpMask natrénovaná na jedné datové sadě může být použita na jinou datovou sadu bez dodatečného trénování. Hlavním důvodem prováděného výzkumu a důležitou výhodou oproti existujícímu výzkumu využívajícímu předtrénování na jiné datové sadě (např. [Voigtlaender et al., 2017]) je rychlost vyhodnocení modifikované SharpMask sítě, která je použitelná pro zpracování videosekvencí v reálném čase.

Přestože základní architektura vychází ze SharpMask, liší se v několika důležitých charakteristikách:

- Trénování segmentační masky probíhá pouze v rámci navržené architektury. Původní SharpMask vyžaduje ještě natrénovanou DeepMask architekturu, ze které vychází.
- Výstupem je jedna segmentační maska pro celý obraz. Vyskytuje-li se v obraze více objektů, všechny budou v masce přítomny. Původní SharpMask je trénován pouze na výřezech obrazu, které obsahují jeden objekt.
- Původní SharpMask obsahuje 32 kanálové konvoluční vrstvy v refinement modulu, použitá architektura využívá pouze 16, čímž dochází k výrazné úspoře parametrů a výpočetních prostředků při zachování schopnosti generovat ostré a přesné segmentační masky.
- Před výstupní vrstvu byla přidána normalizace pomocí batch normalization vrstvy, která usnadňuje trénování výstupní vrstvy, jejíž výstup je v intervalu od 0 do 1 (viz kapitola 4.3.1.1) a umožňuje tím i redukci počtu parametrů refinenement modulu zmíněnou v předchozím bodě.

4.4.1 Postup trénování

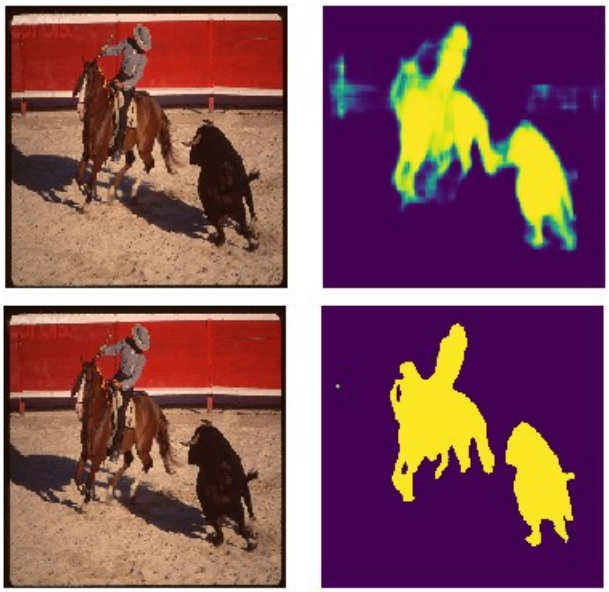
Jako datová sada pro trénování segmentačních masek byla použita sada Microsoft COCO (Common Objects in Context) [Lin et al., 2014], která obsahuje celkem 60 000 obrázků s ručně anotovanými segmentačními maskami 80 běžných objektů, jako jsou osoby, automobily, vybavení domácnosti, jídlo apod.

Datová sada byla předzpracována pro snadné zpracování neuronovou sítí v průběhu trénování. Všechny obrázky byly zmenšeny na velikost 224×224 pixelů, což je vstupní velikost obrazu předtrénované VGG-16 sítě použité jako součást architektury (viz kapitola 4.3.1.1). Výstupní maska pro trénování byla normalizována na hodnotu 1 pro body, které obsahují libovolný objekt a 0 pro body reprezentující pozadí (viz Obrázek 23). Nakonec byla sada náhodně promíchána a rozdělena v poměru 80:20 na trénovací a validační sadu. Zvláštní testovací sada nebyla vytvářena, protože pro testování byla použita odlišná datová sada (viz následující podkapitola).

Obrázek 23: Ukázka obrázků z předzpracované trénovací sady MS COCO

Síť byla nejprve trénována po 32 epoch s chybovou funkcí binární křížové entropie (binary cross entropy) pomocí optimalizačního algoritmu ADAM [Kingma et al., 2014] s rychlostí učení (learning rate) 0.001. Následně byla rychlost učení snížena na 0.0001 a síť byla trénována dalších 8 epoch. Vrstvy původní VGG-16 architektury s předtrénovanými vahami byly v této fázi nastaveny jako neměnné, aby nedocházelo k jejich poškození náhodnými gradienty z dosud nenatrénovaných vrstev. Následně byla síť trénována dalších 14 epoch s rychlostí učení 0.00001, kdy byly trénovány i původní VGG-16 vrstvy (tzv. fine-tuning).

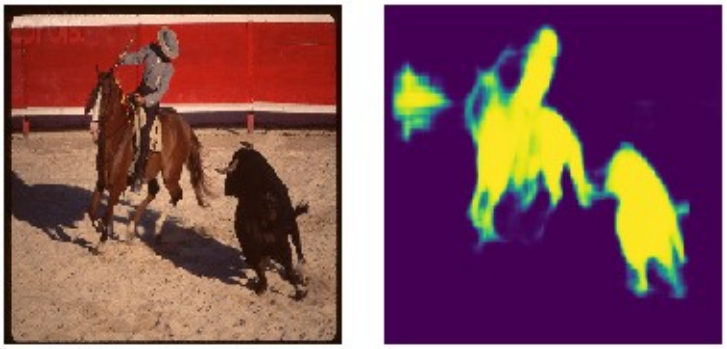
Při použití chybové funkce binární křížové entropie dochází ke vzniku neostrých hran (viz Obrázek 24), což způsobuje nepříjemné vizuální artefakty při přímé vizualizaci či prahování výsledné masky. Stejný problém se vyskytuje i u původní SharpMask architektury (viz Obrázek 13). Natrénovaná síť byla proto ještě po 8 epoch dotrénována pomocí chybové funkce střední absolutní odchylky, která je založena na absolutní hodnotě rozdílu mezi očekávanou a predikovanou hodnotou. Derivace této funkce je rovna –1 v intervalu $[-\infty, 0)$, 0 v bodě 0 a +1 v intervalu $(0, \infty]$, což během trénování nutí síť k ostré predikci 0 nebo 1, protože malá i velká odchylka během trénování do sítě propagují stejně velký gradient. Použití této chybové funkce produkuje téměř binární mapu, není proto ani nutné hledat vhodnou prahovou hodnotu jako v případě použití binární křížové entropie.



Obrázek 24: Ukázka neostrých hran okolo segmentovaných objektů při použití binární křížové entropie jako chybové funkce pro trénování a zlepšení výsledné segmentace při dotrénování s použitím střední absolutní odchylky.

Pro další zlepšení podoby segmentační masky byla testována kombinace střední absolutní odchylky a diskriminační neuronové sítě z frameworku Pix2Pix [Isola et al., 2016]. Cílem diskriminační sítě je o dané segmentační masce rozhodnout, zda pochází z trénovací sady, nebo zda ji vygenerovala segmentační neuronová síť. Protože se jedná o neuronovou síť, jde o diferencovatelnou funkci a lze ji tedy zároveň použít jako chybovou funkci. Diskriminační síť byla nejprve 5 epoch trénována bez propagace chyby do segmentační sítě. Následně byla diskriminační síť použita i jako chybová funkce v kombinaci se střední absolutní odchylkou, podobně jako v [Isola et al., 2016]. Během trénování následujících 8 epoch byla střídavě trénována diskriminační a segmentační síť (jedna měla vždy neměnné váhy) – díky tomu jsou obě sítě nuceny soutěžit a zlepšovat celkový výstup. Byl tak využit stejný princip a postup trénování jako u Generative Adversarial Networks (GAN) (viz kapitola 3.5.6).

Ačkoliv výsledná segmentační maska ze sítě trénované pomocí GAN přístupu je mírně ostřejší než maska výchozí sítě trénované pomocí binární křížové entropie, neprodukovala výrazně lepší výsledky než síť využívající pouze střední absolutní odchylku a výsledná maska byla méně ostrá, viz Obrázek 25. Pro testování byla proto použita pouze síť natrénovaná pomocí střední absolutní odchylky.



Obrázek 25: Ukázka segmentace objektů pomocí sítě trénované GAN přístupem.

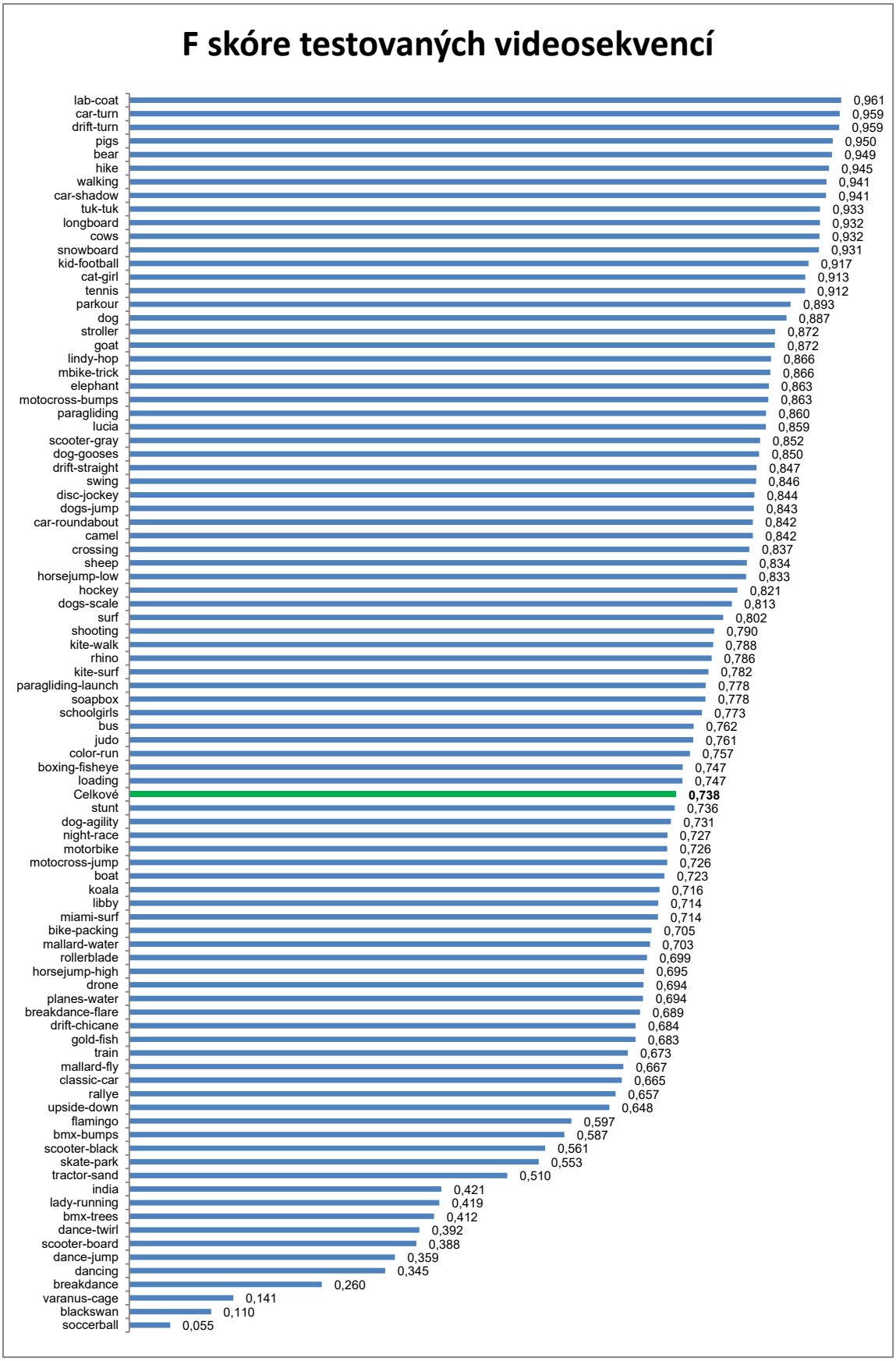
4.4.2 Testování

Cílem trénování segmentační sítě bylo otestovat možnosti použití SharpMask sítě natrénované na datové sadě MS COCO pro segmentaci objektů ve videosekvenci. Jako testovací sada byla použita datová sada DAVIS [Pont-Tuset et al., 2017], která obsahuje 90 krátkých videí, v nichž je každý snímek anotován přesnou segmentační maskou. Síť nebyla na tuto datovou sadu žádným způsobem trénována.

Protože v datové sadě DAVIS jsou anotované jen některé z přítomných objektů, byla při testování použita anotace prvního snímku jako výchozí segmentační maska. Pro každý následující snímek videosekvence byla provedena binární dilatace předchozí segmentační masky o 10 pixelů, aby byl zachycen možný pohyb objektu všemi směry, a tato dilatovaná maska byla vynásobena aktuální segmentační maskou z natrénované neuronové sítě. Výsledkem je průnik dilatované předchozí masky a aktuální masky, čímž se zvyšuje pravděpodobnost, že dojde ke sledování pouze těch objektů, které byly součástí výchozí segmentační masky prvního snímku.

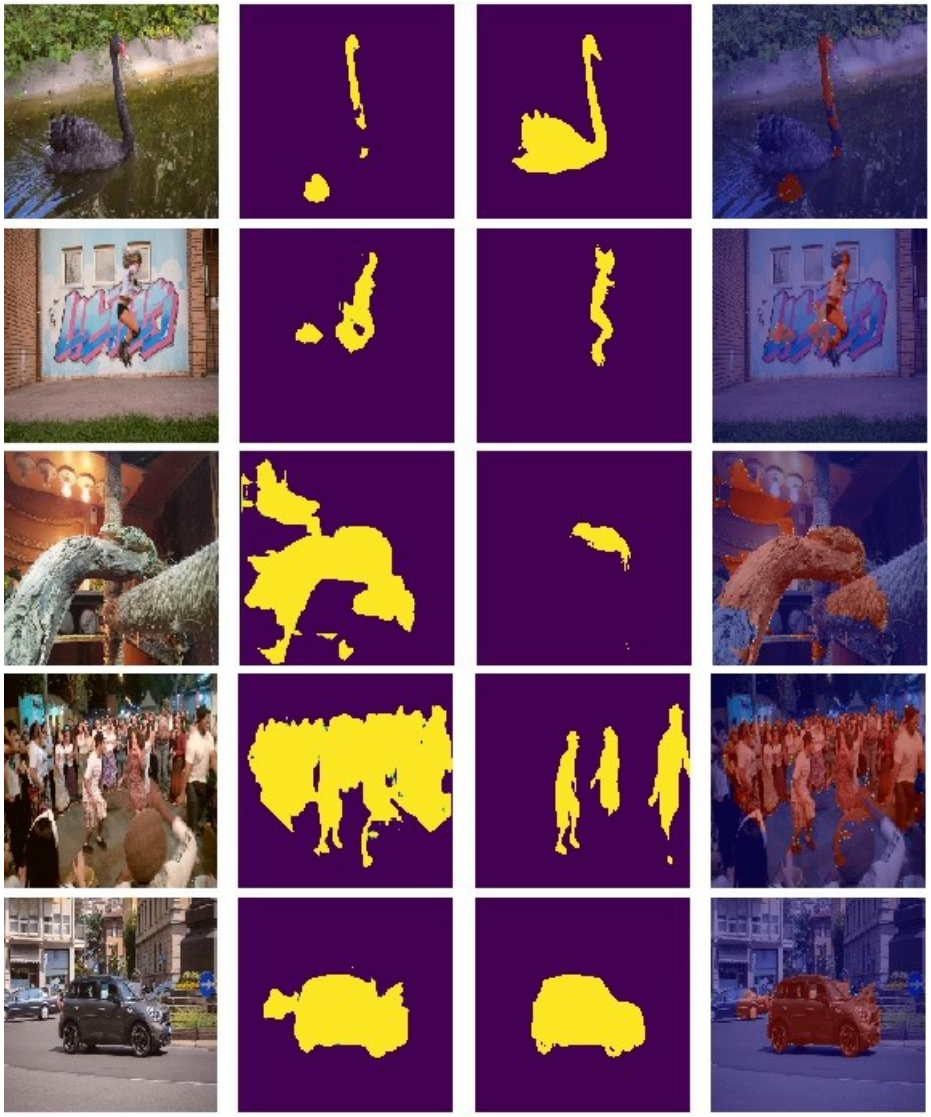
Protože oficiální kritéria pro vyhodnocení DAVIS zahrnují i typ objektu, nelze tato kritéria přímo použít pro vyhodnocení přesnosti navržené sítě, která předpovídá pouze binární segmentační masku. Jako alternativní způsob vyhodnocení bylo proto použito F skóre, podobně jako v kapitole 4.3.2. Hodnoty true positive, false positive a false negative byly v tomto případě počítány pro každý bod segmentační masky a agregovány přes celou videosekvenci.

Celkové F skóre agregované přes všech 90 videí z datové sady je 0,738, výsledky pro jednotlivá videa jsou shrnuty v grafu na následující straně.


F skóre testovaných videosekvencí
lab-coat 0,961
car-turn 0,959
drift-turn 0,959
pigs 0,950
bear 0,949
hike 0,945
walking 0,941
car-shadow 0,941
tuk-tuk 0,933
longboard 0,932
cows 0,932
snowboard 0,931
kid-football 0,917
cat-girl 0,913
tennis 0,912
parkour 0,893
dog 0,887
stroller 0,872
goat 0,872
lindy-hop 0,866
mbike-trick 0,866
elephant 0,863
motocross-bumps 0,863
paragliding 0,860
lucia 0,859
scooter-gray 0,852
dog-gooses 0,850
drift-straight 0,847
swing 0,846
disc-jockey 0,844
dogs-jump 0,843
car-roundabout 0,842
camel 0,842
crossing 0,837
sheep 0,834
horsejump-low 0,833
hockey 0,821
dogs-scale 0,813
surf 0,802
shooting 0,790
kite-walk 0,788
rhino 0,786
kite-surf 0,782
paragliding-launch 0,778
soapbox 0,778
schoolgirls 0,773
bus 0,762
judo 0,761
color-run 0,757
boxing-fisheye 0,747
loading 0,747
Celkové 0,738
stunt 0,736
dog-agility 0,731
night-race 0,727
motorbike 0,726
motocross-jump 0,726
boat 0,723
koala 0,716
libby 0,714
miami-surf 0,714
bike-packing 0,705
mallard-water 0,703
rollerblade 0,699
horsejump-high 0,695
drone 0,694
planes-water 0,694
breakdance-flare 0,689
drift-chicane 0,684
gold-fish 0,683
train 0,673
mallard-fly 0,667
classic-car 0,665
rallye 0,657
upside-down 0,648
flamingo 0,597
bmx-bumps 0,587
scooter-black 0,561
skate-park 0,553
tractor-sand 0,510
india 0,421
lady-running 0,419
bmx-trees 0,412
dance-twirl 0,392
scooter-board 0,388
dance-jump 0,359
dancing 0,345
breakdance 0,260
varanus-cage 0,141
blackswan 0,110
soccerball 0,055

Přestože u některých videí nebyla síť schopna objekt nalézt (např. soccerball), celkově je segmentace pomocí navržené sítě v praxi využitelná na jednodušší videosekvence neobsahující příliš mnoho překrývajících se objektů. Na grafické kartě nVidia GTX 1070 trvá vyhodnocení jednoho snímku o rozlišení 224×224 pixelů průměrně 13,5 ms, lze jej proto využít pro segmentaci videa v reálném čase. Dle očekávání je segmentace nejlepší u objektů, které se vyskytují v trénovací sadě – především osoby, avšak segmentace funguje dobře i například u segmentace videa s divokými prasaty, která se v MS COCO nevyskytují (viz Obrázek 26). Naopak u videí, v nichž se vyskytuje více překrývajících se objektů či jsou v nich objekty málo podobné těm z trénovací sady, dochází k chybám. V případě překrývajících se objektů dochází díky použití jednoduché dilatační techniky k zachycení okolních objektů, v případě neznámých objektů nedojde k žádné či naprosto chybné detekci. Ukázky chybných detekcí znázorňuje Obrázek 27.

Obrázek 26: Ukázka úspěšných segmentací objektů z DAVIS sady. Zleva: vstupní snímek, vypočítaná segmentace, ručně anotovaná segmentace, vypočítaná segmentace překrytá přes vstupní obraz.



Obrázek 27: Ukázka neúspěšných segmentací z DAVIS sady. Zleva: vstupní snímek, vypočítaná segmentace, ručně anotovaná segmentace, vypočítaná segmentace překrytá přes vstupní obraz.

5 Diskuze výsledků

Disertační práce se zabývá oblastí detekce objektů v obraze a ve videosekvenci, která je propojením oblastí analýzy obrazu a strojového učení. V rešeršní části (kapitola 3) byly představeny základní používané techniky pro extrakci užitečných atributů z obrazu, popsány používané algoritmy strojového učení, rozebráno fungování moderních hlubokých neuronových sítí pro detekci obrazu a představeny přístupy pro detekci objektů ve videosekvenci.

Počítačové vidění, především pak detekce objektů v obraze, je oblastí, která v posledních 5 letech zaznamenala prudký vývoj s příchodem dostupného vysoce výkonného paralelního hardwaru v podobě grafických karet, rozsáhlých datových sad a průlomů v trénování rozsáhlých neuronových sítí [Krizhevsky et al., 2012]. Díky velkým investicím do výzkumu v oblasti dochází u standardních datových sad k inkrementálnímu zlepšení výsledků každých několik měsíců. Namísto snahy o dosažení lepších výsledků na standardních datových sadách se proto výzkum v rámci disertační práce zaměřuje na hledání limitů současných přístupů návrhem vlastní datové sady, která se zaměřuje na praktický problém sledování laboratorních potkanů ve vizuálně obtížném prostředí. Během testování bylo zjištěno, že přístupy, které nevyžadují trénování a jsou založeny na sledování obecného objektu, který je označen v prvním snímku videosekvence, selhávají v případě navržené datové sady sledovat hlavu potkana. I pro metodu Faster R-CNN založenou na neuronových sítích představuje navržená datová sada problém především díky relativně malé velikosti detekovaného objektu, kterou není Faster R-CNN schopna během trénování dostatečně rozlišit a vzniká tak mnoho falešných detekcí objektu v místech, kde se nenachází. Jedinou existující architekturou, která byla schopna objekt detekovat na v praxi využitelné úrovni, je YOLOv2, jejíž F skóre při detekci bylo 0,894 a přesnost (precision) dokonce 99,3 %, viz Tabulka 6.

V reakci na provedené testování byla navržena vlastní metoda pro sledování pozice objektu v obraze. Navržená metoda kombinuje hlubokou neuronovou síť, částicový filtr, optický tok a shlukovou analýzu pro přesnější detekci pozice objektu v čase. Navržený detektor dosahuje F skóre 0.921 na testovací sekvenci. Nevýhodou navrženého detektoru oproti existujícím řešením je pouze detekce umístění objektu v obraze, nikoliv celého ohraničujícího obdélníku. Kapitola 4.3.2.1 zmiňuje možné řešení tohoto problému pomocí rozšíření výstupu detekční neuronové sítě.

Nejpokročilejší formou detekce objektu v obraze a videosekvenci je segmentace objektu, kdy každý pixel vstupního obrazu je klasifikován, zda náleží objektu či nikoliv. Pro segmentaci byla použita stejná architektura neuronové sítě jako v případě detekční architektury, založená na zjednodušené segmentační architektuře SharpMask (viz kapitola 4.3.1.1). Navržená segmentační síť se od původní

architektury liší v několika základních aspektech – výstupem je segmentační maska pro celý obraz, nikoliv pouze výřez, síť obsahuje extra normalizační vrstvu, díky které bylo možné výrazně snížit počty konvolučních filtrů, a alternativní způsob trénování, který produkuje velmi ostré segmentační masky. Díky zjednodušení architektury je síť možné používat pro segmentaci obrazu či videosekvence v reálném čase. Byla rovněž otestována generalizační schopnost segmentační sítě, kdy pro testování byla použita zcela jiná datová sada s jinými typy objektů než pro trénování. Na testovací sadě dosahuje F skóre segmentační sítě velmi dobré hodnoty 0,738, u 15 z celkových 90 testovaných videosekvencí dokonce přesahuje hodnotu 0,9.

V rámci výzkumu byla zkoumána i možnost použití alternativních vstupních atributů do neuronových sítí. Primárně byl zkoumán operátor Local Binary Patterns (LBP), který se vyznačuje invariancí vůči velké řadě geometrických i jasových zkreslení. Byla navržena transformace binárního vzoru na dvě reálné hodnoty, díky čemuž je snazší jej využít v kombinaci s neuronovými sítěmi. Testování však ukázalo, že využití LBP místo samotných pixelů jako vstupu do neuronových sítí nepřináší zlepšení přesnosti detekce (viz kapitola 4.2.4).

Protože zpracování velkého množství dat a trénování neuronových sítí závisí na využití výpočetního výkonu grafických karet a vhodných optimalizací, část výzkumné činnosti byla zaměřena i na vývoj programů pro grafické karty [Janečka et al., 2013; Janečka et al., 2011; Petránek et al., 2011; Ježek et al., 2013; Ježek et al., 2014; Tobola et al., 2014] a řešení optimalizačních úloh [Milková et al., 2013; Petránek et al., 2014; Milková et al., 2016].

5.1 Naplnění cílů práce

Hlavním cílem práce stanoveným v kapitole 2 bylo vytvoření systému či systémů pro detekci umístění objektu v obraze, který dosahuje dostatečně vysoké přesnosti na to, aby mohl výrazně usnadnit či zcela nahradit dozor člověka. Detekční a segmentační algoritmy navržené v kapitolách 4.3 a 4.4 dosahují F skóre vyššího než 0,9 na datech, na něž byly natrénovány. Zásah člověka je tak nutný v méně než 10 % případů, což je výrazná úspora oproti stavu, kdy systém pro detekci objektů v obraze není použit vůbec.

Následující podkapitoly diskutují dosažené výsledky v kontextu dílčích cílů práce definovaných v kapitole 2.

5.1.1 Testování stávajících state-of-the-art metod na nestandardní datové sadě

V rámci kapitoly 4.3 bylo otestováno 5 existujících metod, k nimž je k dispozici referenční implementace – GOTURN, OpenTLD, CMT, Faster R-CNN a YOLOv2. Metody byly testovány na vlastní datové sadě navržené v kapitole 4.1, která obsahuje velkou řadu v praxi reálně se vyskytujících zkreslení.

5.1.2 Analýza a zlepšení extrakce atributů

V rámci výzkumu (viz kapitola 4.2 a [Petranek et al., 2016]) byla zkoumána extrakce binárních atributů z obrazu pomocí operátoru Local Binary Patterns, který je invariantní vůči rotaci, posunutí, změně měřítka, jasu a kontrastu. Díky binárnímu výstupu je obtížné jej přímo propojit s moderními detektory založenými na neuronových sítích, byla proto navržena transformace z binární formy na dvě hodnoty v intervalu [0, 1], které jsou navíc schopny reprezentovat větší množství vzorů než běžně používaná varianta uniformních LBP.

5.1.3 Zlepšení detekce umístění objektu v obraze

Na základě testování existujících metod byla navržena nová metoda pro sledování umístění objektu v obraze založená na predikci pravděpodobnostní mapy výskytu objektu, kde každý pixel výstupní mapy představuje pravděpodobnost výskytu objektu v daném bodě (viz kapitola 4.3.1.1). Pro výpočet pravděpodobnostní mapy byla použita modifikovaná hluboká neuronová síť původně navržená pro segmentaci obrazu.

Stejná architektura poté byla použita v kapitole 4.4 i pro vytvoření kompletní segmentační masky všech objektů v obraze, kdy je přítomnost objektu predikována pro každý bod vstupního obrazu. Testování ukazuje, že natrénovaná síť má dobré generalizační schopnosti i přesto, že byla oproti výchozí architektuře SharpMask výrazně zjednodušena.

5.1.4 Propojení detekce objektu s jeho sledováním ve videosekvenci

Protože výstupem detekční sítě z kapitoly 4.3.1.1 je pravděpodobnostní mapa, byl využit částicový filtr, v kombinaci s výpočtem optického toku a shlukové analýzy pro zpřesnění detekce objektu v čase v průběhu videosekvence. Navržená metoda dosahuje nejlepší schopnosti predikce pozice objektu v navržené datové sadě z testovaných metod a jako jediná dosahuje F skóre většího než 0,9, viz kapitola 4.3.2.

Kapitola 4.4.2 ukazuje, že navržená segmentační neuronová síť v kombinaci s jednoduchým operátorem binární dilatace umožňuje velmi přesně segmentovat objekty i ve videosekvenci. Dokonce je možné velmi přesně segmentovat i objekty, které se nevyskytují v původní trénovací sadě.

6 Závěr

V rámci výzkumu bylo dosaženo výsledků ve čtyřech dílčích oblastech – návrhu vlastní datové sady a testování existujících řešení na této datové sadě, vylepšení reprezentace atributů založených na Local Binary Patterns (LBP), vytvoření vlastního detektoru umístění objektu ve videosekvenci a natrénování velmi rychlé neuronové sítě pro segmentaci objektů ve statickém obraze a její následné použití na videosekvenci.

V rámci spolupráce s Katedrou toxikologie Fakulty vojenského zdravotnictví Univerzity obrany byla vytvořena datová sada, jejímž cílem je sledovat pohyb hlavy laboratorního potkana pomocí běžně dostupných kamer. Celkem bylo vytvořeno 12 videí, z nichž bylo manuálně anotováno 10 000 snímků, které lze použít pro učení. Snímky v datové sadě obsahují mnoho zkreslení – okluze objektu, různá natočení objektu, nízké rozlišení, rozmazání pohybu, měnící se světelné podmínky a nízký kontrast mezi sledovaným objektem a pozadím. Tato zkreslení se ukázala být výzvou i pro moderní detektory založené na hlubokých neuronových sítích (viz kapitola 4.3.2).

Kvůli výše uvedeným zkreslením byla zkoumána možnost využití operátoru LBP, který byl navržen tak, aby byl vůči většině geometrických a jasových zkreslení invariantní. Bylo navrženo rozšíření LBP, které umožňuje kompaktní reprezentaci jednotlivých vzorů, a tím snadné propojení s algoritmy strojového učení. V rámci testování LBP bylo zjištěno, že neuronové sítě využívající přímo obrazové pixely jako vstup dosahují lepších výsledků, LBP proto nebyly použity v návrhu vlastního detektoru.

V reakci na výsledky testování existujících detektorů na navržené datové sadě byl vyvinut vlastní detektor, jehož výstupem je pozice objektu v obraze. Navržený detektor propojuje neuronovou síť, jež predikuje pravděpodobnost pozice v rámci jednoho snímku, s modifikovaným částicovým filtrem, který objekt sleduje v čase. Detektor na navržené datové sadě předčí existující detektory a ukazuje, že běžně používané detektory často nedosahují dostatečné přesnosti v případně náročných zkreslení.

Protože výše uvedený detektor navržený v kapitole 4.3 detekuje pouze pozici objektu v obraze, nikoliv jeho velikost, byla v souladu s cílem práce zkoumána možnost zpřesnit detekci umístění pomocí segmentace celého objektu. Jako základ pro segmentaci byla využita detekční síť z kapitoly 4.3, která je zjednodušením převzaté segmentační architektury. Pomocí vhodného způsobu trénování bylo dosaženo ostrých segmentací objektů v obraze, které se často kvalitou blíží úrovni člověka. Použití navržené segmentační sítě na novou datovou sadu (bez nového trénování) ukazuje i dobrou generalizační schopnost. Díky zjednodušené architektuře trvá vyhodnocení jednoho snímku pouze 13,5 ms na moderní grafické kartě, což umožňuje použití pro segmentaci videa v reálném čase.

7 Literatura

7.1 Použitá literatura

ALEXEY. Darknet: Windows and Linux Version of Darknet Yolo v2 Neural Networks for Object Detection. 2017.

ARJOVSKY, M., CHINTALA, S. & BOTTOU, L. Wasserstein Gan. ArXiv Preprint ArXiv:1701.07875, 2017.

BAY, H., TUYTELAARS, T. & VAN GOOL, L. SURF: Speeded up Robust Features. In Computer Vision–ECCV 2006. Springer, 2006, pp. 404–417. ISBN 978-3-540-33838-3.

BERAN, L. & PETRÁNEK, K. Rozpoznání obličejů na mobilních zařízeních. In Sborník příspěvků z Letní školy „Mezioborové přístupy informatiky a kognitivní vědy“. Fakulta informatiky a managementu Univerzity Hradec Králové : Fakulta informatiky a managementu Univerzity Hradec Králové, 2011, pp. 8–15. ISBN 97880743513720.

BRADSKI, G. The OpenCV Library [online]. 2000, Dr. Dobb's. [Cit. 14.4.2015]. Dostupné z WWW: <http://www.drdobbs.com/open-source/the-opencv-library/184404319>.

BRADSKI, G.R. & KAEHLER, A. Learning OpenCV Computer Vision with the OpenCV Library. Sebastopol, Calif. : O'Reilly, 2008. ISBN 9780596516130 0596516134.

BREIMAN, L. Random Forests. Machine Learning, 2001, vol. 45, no. 1, pp. 5–32.

CHOLLET, F. CIFAR-10 CNN in Keras [online]. 2016a, GitHub. [Cit. 11.6.2016]. Dostupné z WWW: <https://github.com/fchollet/keras/blob/master/examples/cifar10_cnn.py>.

CHOLLET, F. Keras Examples [online]. 2016b, GitHub. [Cit. 11.6.2016]. Dostupné z WWW: <https://github.com/fchollet/keras/blob/master/examples/>.

CHOLLET, F. MNIST CNN in Keras [online]. 2016c, GitHub. [Cit. 11.6.2016]. Dostupné z WWW: <https://github.com/fchollet/keras/blob/master/examples/mnist_cnn.py>.

CHOLLET, F. MNIST MLP in Keras [online]. 2016d, GitHub. [Cit. 11.6.2016]. Dostupné z WWW: <https://github.com/fchollet/keras/blob/master/examples/mnist_mlp.py>.

COMANICIU, D. & MEER, P. Mean Shift: A Robust Approach toward Feature Space Analysis. Pattern Analysis and Machine Intelligence, IEEE Transactions On, 2002, vol. 24, no. 5, pp. 603–619.

DEL MORAL, P. Non-Linear Filtering: Interacting Particle Resolution. Markov Processes and Related Fields, 1996, vol. 2, no. 4, pp. 555–581.

DUCHI, J., HAZAN, E. & SINGER, Y. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. The Journal of Machine Learning Research, 2011, vol. 12, pp. 2121–2159.

EVERINGHAM, M., VAN GOOL, L., WILLIAMS, C.K.I., WINN, J. & ZISSERMAN, A. The Pascal Visual Object Classes (VOC) Challenge. International Journal of Computer Vision, 2010, vol. 88, no. 2, pp. 303–338.

FREUND, Y. An Adaptive Version of the Boost by Majority Algorithm. Machine Learning, 2001, vol. 43, no. 3, pp. 293–318.

FUNAYAMA, R. & YANAGIHARA, H. ROBUST INTEREST POINT DETECTOR AND DESCRIPTOR, US Patent 2009238460. US Patent, 2009.

GIRSHICK, R. Fast R-Cnn. In Proceedings of the IEEE International Conference on Computer Vision. 2015, pp. 1440–1448.

GIRSHICK, R., DONAHUE, J., DARRELL, T. & MALIK, J. Region-Based Convolutional Networks for Accurate Object Detection and Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2016, vol. 38, no. 1, pp. 142–158.

GLOROT, X., BORDES, A. & BENGIO, Y. Deep Sparse Rectifier Networks. In Proceedings of the 14th International Conference on Artificial Intelligence and Statistics. JMLR W&CP Volume. 2011, pp. 315–323.

GOODFELLOW, I., POUGET-ABADIE, J., MIRZA, M., XU, B., WARDE-FARLEY, D., et al. Generative Adversarial Nets. In Advances in Neural Information Processing Systems. 2014, pp. 2672–2680.

GRAVES, A., WAYNE, G. & DANIHELKA, I. Neural Turing Machines. ArXiv Preprint ArXiv:1410.5401, 2014.

HARTIGAN, J.A. & WONG, M.A. Algorithm AS 136: A k-Means Clustering Algorithm. Applied Statistics, 1979, vol. 28, no. 1, pp. 100–108.

HE, K., ZHANG, X., REN, S. & SUN, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016, pp. 770–778.

HE, K., ZHANG, X., REN, S. & SUN, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. ArXiv:1502.01852 [Cs], 2015.

HELD, D., THRUN, S. & SAVARESE, S. Learning to Track at 100 FPS with Deep Regression Networks. In Computer Vision – ECCV 2016. Lecture Notes in Computer Science. Springer, Cham, 2016, pp. 749–765. ISBN 978-3-319-46447-3.

HENON, Y. Keras-Frcnn on GitHub [online]. 2017, GitHub. [Cit. 28.9.2017]. Dostupné z WWW: <https://github.com/yhenon/keras-frcnn>.

HINTON, G. RMSProp [online]. 2012. Dostupné z WWW: <http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf>.

HINTON, G., NITISHSRIVASTAVA, A. & SALAKHUTDINOV, I.R.R. Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors. ArXiv Preprint ArXiv:1207.0580, 2012a.

HINTON, G.E. & SALAKHUTDINOV, R. A Better Way to Pretrain Deep Boltzmann Machines. In Advances in Neural Information Processing Systems. 2012b, pp. 2456–2464. ISBN 978-1-62748-003-1.

HINTON, G.E. & SALAKHUTDINOV, R.R. Reducing the Dimensionality of Data with Neural Networks. Science, 2006, vol. 313, no. 5786, pp. 504–507.

HOARE, C.A.R. Algorithm 65: Find. Commun. ACM, 1961, vol. 4, no. 7, pp. 321–322. ISSN 0001-0782.

HOCHREITER, S. The Vanishing Gradient Problem during Learning Recurrent Neural Nets and Problem Solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 1998, vol. 6, no. 02, pp. 107–116.

HOCHREITER, S. & SCHMIDHUBER, J. Long Short-Term Memory. Neural Computation, 1997, vol. 9, no. 8, pp. 1735–1780.

HORPRASERT, T., HARWOOD, D. & DAVIS, L. A Statistical Approach for Real-Time Robust Background Subtraction and Shadow Detection. IEEE ICCV, 1999, vol. 99, pp. 1–19.

HOUGH, P.V.C. Method and Means for Recognizing Complex Patterns, US Patent 3,069,654. 1962.

IOFFE, S. & SZEGEDY, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. ArXiv:1502.03167 [Cs], 2015.

ISOLA, P., ZHU, J.-Y., ZHOU, T. & EFROS, A.A. Image-to-Image Translation with Conditional Adversarial Networks. ArXiv Preprint ArXiv:1611.07004, 2016.

JAIN, R., KASTURI, R. & SCHUNCK, B.G. Machine Vision. New York : McGraw-Hill New York, 1995. ISBN 0-07-032018-7.

JANEČKA, P., JEŽEK, B., VANĚK, J. & PETRÁNEK, K. Využití Flash Stage3D API k prostorovým vizualizacím v prostředí Internetu. In Sborník příspěvků 31. konference o geometrii a počítačové grafice. Ostrava-Poruba: Vysoká škola báňská – Technická univerzita Ostrava, 2011, pp. 107–116. ISBN 978-80-248-2524-3.

JANEČKA, P., PETRÁNEK, K., JEŽEK, B. & VANĚK, J. Vizualizace implicitních ploch metodou Raymarching. In Sborník příspěvků 33. konference o geometrii a grafice. Horní Lomná : Vysoká škola báňská - Technická univerzita v Ostravě, 2013, pp. 113–123. ISBN 978-80-248-3251-73.

JEŽEK, B., VANĚK, J., PROCHÁZKA, M., HALAJČUK, T. & PETRÁNEK, K. Využití simulace a GIS při přípravě na mimořádnou událost. In Konference GIS Esri v ČR. Praha, 2013.

JEŽEK, B., VANĚK, J., PROCHÁZKA, M. & PETRÁNEK, K. Použití počítačové grafiky při řešení pokrytí zásahového území leteckou záchrannou službou. In Sborník příspěvků 34. konference o geometrii a grafice. Vlachovice : Jihočeská univerzita v Českých Budějovicích, 2014, pp. 115–122. ISBN 978-80-7394-470-4.

KALAL, Z., MATAS, J. & MIKOLAJCZYK, K. Online Learning of Robust Object Detectors during Unstable Tracking. In Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference On. 2009, pp. 1417–1424. ISBN 978-1-4244-4442-7.

KALAL, Z., MATAS, J. & MIKOLAJCZYK, K. P-N Learning: Bootstrapping Binary Classifiers by Structural Constraints. In Proceedings of the 23rd IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2010a.

KALAL, Z., MIKOLAJCZYK, K. & MATAS, J. Face-Tld: Tracking-Learning-Detection Applied to Faces. In ICIP 2010 Proceedings. Hong Kong : IEEE, 2010b, pp. 3789–3792. ISBN 978-1-4244-7994-8.

KALMAN, R.E. & OTHERS. A New Approach to Linear Filtering and Prediction Problems. Journal of Basic Engineering, 1960, vol. 82, no. 1, pp. 35–45.

KINGMA, D. & BA, J. Adam: A Method for Stochastic Optimization. ArXiv Preprint ArXiv:1412.6980, 2014.

KRIZHEVSKY, A. & HINTON, G. Learning Multiple Layers of Features from Tiny Images. Computer Science Department, University of Toronto, Tech. Rep, 2009, vol. 1, no. 4, p. 7.

KRIZHEVSKY, A., SUTSKEVER, I. & HINTON, G.E. ImageNet Classification with Deep Convolutional Neural Networks. , 2012.

LE, Q.V., MONGA, R., DEVIN, M., CHEN, K., CORRADO, G.S., DEAN, J. & NG, A.Y. Building High-Level Features Using Large Scale Unsupervised Learning. In Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference On. IEEE, 2013, pp. 8595–8598.

LECUN, Y. & BENGIO, Y. Convolutional Networks for Images, Speech, and Time-Series. In The Handbook of Brain Theory and Neural Networks. MIT, 1995. ISBN 978-0-262-01197-6.

LECUN, Y., BOTTOU, L., BENGIO, Y. & HAFFNER, P. Gradient-Based Learning Applied to Document Recognition. Proceedings of the IEEE, 1998, vol. 86, no. 11, pp. 2278–2324.

LECUN, Y. & CORTES, C. The MNIST Database of Handwritten Digits [online]. 2017. [Cit. 11.6.2016]. Dostupné z WWW: <http://yann.lecun.com/exdb/mnist/>.

LEE, H., WU, C. & AGHAJAN, H. Nonstationary Background Removal Via Multiple Camera Collaboration. 2007 First ACM/IEEE International Conference on Distributed Smart Cameras, 2007, pp. 321–327.

LEUNG, T. & MALIK, J. Representing and Recognizing the Visual Appearance of Materials Using Three-Dimensional Textons. International Journal of Computer Vision, 2001, vol. 43, no. 1, pp. 29–44. ISSN 0920-5691, 1573-1405.

LEWIS, J.P. Fast Template Matching. In Proceedings of the Canadian Image Processing and Pattern Recognition Society Conference on Vision Interface. Quebec City, 1995, pp. 120–123.

LI, Y., QI, H., DAI, J., JI, X. & WEI, Y. Fully Convolutional Instance-Aware Semantic Segmentation. ArXiv Preprint ArXiv:1611.07709, 2016.

LIAO, S., ZHU, X., LEI, Z., ZHANG, L. & LI, S. Learning Multi-Scale Block Local Binary Patterns for Face Recognition. Advances in Biometrics, 2007, pp. 828–837.

LIN, T.-Y., MAIRE, M., BELONGIE, S., HAYS, J., PERONA, P., et al. Microsoft COCO: Common Objects in Context. In Computer Vision – ECCV 2014. Lecture Notes in Computer Science. Springer, Cham, 2014, pp. 740–755. ISBN 978-3-319-10601-4.

LINDEBERG, T. Scale-Space Theory in Computer Vision. Dordrecht, The Netherlands : Springer Science & Business Media, 1993. ISBN 0-7923-9418-6.

LIU, W., ANGUELOV, D., ERHAN, D., SZEGEDY, C., REED, S., FU, C.-Y. & BERG, A.C. Ssd: Single Shot Multibox Detector. In European Conference on Computer Vision. Springer, 2016, pp. 21–37.

LONG, J., SHELHAMER, E. & DARRELL, T. Fully Convolutional Networks for Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015, pp. 3431–3440.

LOWE, D. Method and Apparatus for Identifying Scale Invariant Features in an Image and Use of Same for Locating an Object in an Image, US Patent 6711293. 2004.

LOWE, D.G. Object Recognition from Local Scale-Invariant Features. In Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference On. 1999, pp. 1150–1157. ISBN 0-7695-0164-8.

LUCAS, B.D., KANADE, T. & OTHERS. An Iterative Image Registration Technique with an Application to Stereo Vision. In IJCAI. 1981, pp. 674–679. ISBN 0-934613-02-8.

MARIMONT, R.B. & SHAPIRO, M.B. Nearest Neighbour Searches and the Curse of Dimensionality. IMA Journal of Applied Mathematics, 1979, vol. 24, no. 1, pp. 59–70. ISSN 0272-4960, 1464-3634.

MATAS, J. & ŠOCHMAN, J. AdaBoost [online]. 2012. Dostupné z WWW: <www.robots.ox.ac.uk/~az/.../adaboost_matas.pdf>.

MILKOVÁ, E. & PETRÁNEK, K. Puzzle - on the Complexity of a Path Avoiding Forbidden Pairs of Vertices. In 3rd World Conference on Innovation and Computer Science (INSODE-2013). Antalya : Global Journal on Technology, 2013.

MILKOVÁ, E. & PETRÁNEK, K. Restricted Version of Paths Avoiding Forbidden Pairs – Complexity and Algorithm. Recent Patents on Computer Science, 2016, vol. 2017, no. 10. ISSN 1874-4796.

MOODY, J.E., HANSON, S.J., KROGH, A. & HERTZ, J.A. A Simple Weight Decay Can Improve Generalization. Advances in Neural Information Processing Systems, 1995, vol. 4, pp. 950–957.

MOZER, M.C. A Focused Back-Propagation Algorithm for Temporal Pattern Recognition. Complex Systems, 1989, vol. 3, no. 4, pp. 349–381.

NEBEHAY, G. & PFLUGFELDER, R. Clustering of Static-Adaptive Correspondences for Deformable Object Tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015, pp. 2784–2791.

NESTEROV, Y. Gradient Methods for Minimizing Composite Objective Function. In CORE Discussion Paper. Catholic University of Louvain : Core, 2007.

NG, A., NGIAM, J., FOO, C.Y., MAI, Y. & SUEN, C. UFLDL Stanford - Sparse Coding [online]. 2015a. Dostupné z WWW: <http://ufldl.stanford.edu/wiki/index.php/Exercise:Sparse_Coding>.

NG, A., NGIAM, J., FOO, C.Y., MAI, Y. & SUEN, C. UFLDL Stanford - Whitening [online]. 2015b. Dostupné z WWW: <http://ufldl.stanford.edu/wiki/index.php/Exercise:PCA_and_Whitening>.

NIELSEN, M.A. Neural Networks and Deep Learning [online]. 2015. [Cit. 14.4.2015]. Dostupné z WWW: <http://neuralnetworksanddeeplearning.com>.

OJALA, T., PIETIKAINEN, M. & MAENPAA, T. Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns. Pattern Analysis and Machine Intelligence, IEEE Transactions On, 2002, vol. 24, no. 7, pp. 971–987.

OLSHAUSEN, B.A. & FIELD, D.J. Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1? Vision Research, 1997, vol. 37, no. 23, pp. 3311–3325. ISSN 0042-6989.

PERAZZI, F., PONT-TUSET, J., MCWILLIAMS, B., VAN GOOL, L., GROSS, M. & SORKINE-HORNUNG, A. A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016, pp. 724–732.

PETRÁNEK, K., JANEČKA, P. & VANĚK, J. Using Local Binary Patterns for Object Detection in Images. Global Journal of Computer Sciences, 2015, vol. 5, no. 1, pp. 7–12. ISSN 2301-2587.

PETRÁNEK, K., VANĚK, J. & JEŽEK, B. Local Binary Patterns a jejich použití v počítačovém vidění. In Sborník příspěvků 32. konference o geometrii a grafice. Plzeň: Vydavatelský servis, 2012, pp. 173–179. ISBN 978-80-86843-40-7.

PETRÁNEK, K., VANĚK, J., JEŽEK, B. & JANEČKA, P. Akcelerace algoritmů pro segmentaci obrazu. In Sborník příspěvků 31. konference o geometrii a počítačové grafice. Ostrava-Poruba: Vysoká škola báňská – Technická univerzita Ostrava, 2011, pp. 189–197. ISBN 978-80-248-2524-3.

PETRÁNEK, K., VANĚK, J., JEŽEK, B. & JANEČKA, P. Detekce Pozadí Scény Při Stereoskopickém Snímání. In Sborník Příspěvků 30. Konference o Geometrii a Počítačové Grafice. Zlenice, 2010. ISBN 978-80-7378-136-1.

PETRÁNEK, K., VANĚK, J., JEŽEK, B. & JANEČKA, P. Registrace GPS dat na silniční síť. In Sborník příspěvků 34. konference o geometrii a grafice. Vlachovice : Jihočeská univerzita v Českých Budějovicích, 2014, pp. 199–204. ISBN 978-80-7394-470-4.

PETRÁNEK, K., VANĚK, J., JEŽEK, B. & JANEČKA, P. Robustní porovnání obrazů pomocí vzájemné informace. In Sborník příspěvků 33. konference o geometrii a grafice. Horní Lomná : Vysoká škola báňská - Technická univerzita v Ostravě, 2013, pp. 199–203. ISBN 978-80-248-3251-73.

PETRANEK, K., VANEK, J. & MILKOVA, E. Avoiding the Curse of Dimensionality in Local Binary Patterns. In Computational Collective Intelligence. Lecture Notes in Computer Science. Springer, Cham, 2016, pp. 208–217. ISBN 978-3-319-45242-5.

PINHEIRO, P.O., COLLOBERT, R. & DOLLÁR, P. Learning to Segment Object Candidates. In Advances in Neural Information Processing Systems. 2015, pp. 1990–1998.

PINHEIRO, P.O., LIN, T.-Y., COLLOBERT, R. & DOLLÁR, P. Learning to Refine Object Segments. In European Conference on Computer Vision. Springer, 2016, pp. 75–91.

PONT-TUSET, J., PERAZZI, F., CAELLES, S., ARBELÁEZ, P., SORKINE-HORNUNG, A. & VAN GOOL, L. The 2017 Davis Challenge on Video Object Segmentation. ArXiv Preprint ArXiv:1704.00675, 2017.

REAL, E., SHLENS, J., MAZZOCCHI, S., PAN, X. & VANHOUCKE, V. YouTube-BoundingBoxes: A Large High-Precision Human-Annotated Data Set for Object Detection in Video. ArXiv Preprint ArXiv:1702.00824, 2017.

REDMON, J., DIVVALA, S., GIRSHICK, R. & FARHADI, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016a, pp. 779–788.

REDMON, J. & FARHADI, A. YOLO9000: Better, Faster, Stronger. ArXiv:1612.08242 [Cs], 2016b.

REHMAN, M.Z. & NAWI, N.M. The Effect of Adaptive Momentum in Improving the Accuracy of Gradient Descent Back Propagation Algorithm on Classification Problems. In J. M. Zain, W. M. bt W. Mohd, & E. El-Qawasmeh, eds.Software Engineering and Computer Systems. Communications in Computer and Information Science. Springer Berlin Heidelberg, 2011, pp. 380–390. ISBN 978-3-642-22169-9.

REN, S., HE, K., GIRSHICK, R. & SUN, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Advances in Neural Information Processing Systems. 2015, pp. 91–99.

RUMELHART, D.E., HINTON, G.E. & WILLIAMS, R.J. Learning Representations by Back-Propagating Errors. In Cognitive Modeling. 1988, p. chapter 8. ISBN 0-262-66116-0.

RUSSAKOVSKY, O., DENG, J., SU, H., KRAUSE, J., SATHEESH, S., et al. ImageNet Large Scale Visual Recognition Challenge. ArXiv Preprint, ArXiv:1409.0575, 2014a.

RUSSAKOVSKY, O., DENG, J., SU, H., KRAUSE, J., SATHEESH, S., et al. ImageNet Large Scale Visual Recognition Challenge. 2014b.

SALAKHUTDINOV, R. & HINTON, G.E. Deep Boltzmann Machines. In International Conference on Artificial Intelligence and Statistics. 2009, pp. 448–455.

SCHARSTEIN, D. & SZELISKI, R. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. International Journal of Computer Vision, 2002, vol. 47, no. 1–3.

SE, S., LOWE, D. & LITTLE, J. Vision-Based Mobile Robot Localization and Mapping Using Scale-Invariant Features. In Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference On. 2001, pp. 2051–2058.

SHI, J. & MALIK, J. Normalized Cuts and Image Segmentation. Pattern Analysis and Machine Intelligence, IEEE Transactions On, 2000, vol. 22, no. 8, pp. 888–905.

SIMONYAN, K., VEDALDI, A. & ZISSERMAN, A. Deep inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. ArXiv Preprint ArXiv:1312.6034, 2013.

SPRINGENBERG, J., DOSOVITSKIY, A., BROX, T. & RIEDMILLER, M. Striving for Simplicity: The All Convolutional Net. , 2015.

SZEGEDY, C., LIU, W., JIA, Y., SERMANET, P., REED, S., et al. Going Deeper with Convolutions. ArXiv:1409.4842 [Cs], 2014.

TARR, M. & VUONG, Q. Visual Object Recognition [online]. Brown University, 2002. [Cit. 26.4.2012]. Dostupné z WWW: <http://www.staff.ncl.ac.uk/q.c.vuong/pdfs/TarrVuong2002.pdf>.

TOBOLA, J., VANĚK, J., PETRÁNEK, K. & JEŽEK, B. Triviální algoritmus pro interaktivní simulaci vody. In Sborník příspěvků 34. konference o geometrii a grafice. Vlachovice : Jihočeská univerzita v Českých Budějovicích, 2014, pp. 239–246. ISBN 978-80-7394-470-4.

VOIGTLAENDER, P. & LEIBE, B. Online Adaptation of Convolutional Neural Networks for Video Object Segmentation. ArXiv:1706.09364 [Cs], 2017.

WEISSTEIN, E.W. Method of Steepest Descent [online]. 2015. [Cit. 14.4.2015]. Dostupné z WWW: <http://mathworld.wolfram.com/MethodofSteepestDescent.html>.

WILSON, A.C., ROELOFS, R., STERN, M., SREBRO, N. & RECHT, B. The Marginal Value of Adaptive Gradient Methods in Machine Learning. ArXiv Preprint ArXiv:1705.08292, 2017.

ZABIH, R. & WOODFILL, J. Non-Parametric Local Transforms for Computing Visual Correspondence. Computer Vision—ECCV'94, 1994, pp. 151–158.

ZEILER, M.D. ADADELTA: An Adaptive Learning Rate Method. ArXiv Preprint ArXiv:1212.5701, 2012.

7.2 Vlastní publikace

BERAN, L. & PETRÁNEK, K. Rozpoznání obličejů na mobilních zařízeních. In *Sborník příspěvků z Letní školy „Mezioborové přístupy informatiky a kognitivní vědy“*. Fakulta informatiky a managementu Univerzity Hradec Králové : Fakulta informatiky a managementu Univerzity Hradec Králové, 2011, pp. 8–15. ISBN 97880743513720.

GREGOR, M., PETRÁNEK, K. & BURIAN, M. Robotická ruka - konstrukce a použití. In *Sborník příspěvků z Letní školy „Mezioborové přístupy informatiky a kognitivní vědy“*. Fakulta informatiky a managementu Univerzity Hradec Králové : Fakulta informatiky a managementu Univerzity Hradec Králové, 2011, pp. 8–15. ISBN 97880743513720.

JANEČKA, P., JEŽEK, B., VANĚK, J. & PETRÁNEK, K. Prezentace prostorové scény na webu. In *Sborník příspěvků 30. konference o geometrii a grafice*. Praha: Matfyzpress, 2010, pp. 125–134. ISBN 978-80-7378-136-1.

JANEČKA, P., JEŽEK, B., VANĚK, J. & PETRÁNEK, K. Využití Flash Stage3D API k prostorovým vizualizacím v prostředí Internetu. In *Sborník příspěvků 31. konference o geometrii a počítačové grafice*. Ostrava-Poruba: Vysoká škola báňská – Technická univerzita Ostrava, 2011, pp. 107–116. ISBN 978-80-248-2524-3.

JANEČKA, P. & PETRÁNEK, K. Implicit Surface Visualisation Using Adaptive Raymarching. In *4th World Conference on Innovation and Computer Science (INSODE-2014)*. Rome : Global Journal on Technology, 2014. ISBN 2147-5369.

JANEČKA, P., PETRÁNEK, K., JEŽEK, B. & VANĚK, J. Vizualizace implicitních ploch metodou Raymarching. In *Sborník příspěvků 33. konference o geometrii a grafice*. Horní Lomná : Vysoká škola báňská - Technická univerzita v Ostravě, 2013, pp. 113–123. ISBN 978-80-248-3251-73.

JEŽEK, B., VANĚK, J., JANEČKA, P. & PETRÁNEK, K. Jak nelhat při vizualizaci. In *Sborník příspěvků 30. konference o geometrii a grafice*. Praha: Matfyzpress, 2010, pp. 135–143. ISBN 978-80-7378-136-1.

JEŽEK, B., VANĚK, J., PROCHÁZKA, M., HALAJČUK, T. & PETRÁNEK, K. Využití simulace a GIS při přípravě na mimořádnou událost. In *Konference GIS Esri v ČR*. Praha, 2013.

JEŽEK, B., VANĚK, J., PROCHÁZKA, M. & PETRÁNEK, K. Použití počítačové grafiky při řešení pokrytí zásahového území leteckou záchrannou službou. In *Sborník příspěvků 34. konference o geometrii a grafice*. Vlachovice : Jihočeská univerzita v Českých Budějovicích, 2014, pp. 115–122. ISBN 978-80-7394-470-4.

MILKOVÁ, E. & PETRÁNEK, K. Improving Programming Courses Using Aptitude Testing and Learning Styles. In *Recent Advances in Computer Science, Proc. of the 19th International Conference on Computers (Part of CSCC'15)*. Zakynthos Island, Greece, 2015, pp. 211–214. ISBN 978-1-61804-320-7.

MILKOVÁ, E. & PETRÁNEK, K. Programming Courses Reflecting Students' Aptitude Testing and Implementing Learning Style Preferences Research Results. *International Journal of Mathematics and Computers in Simulation*, 2016a, vol. 2016, no. 10, pp. 218–225. ISSN 1998-0159.

MILKOVÁ, E. & PETRÁNEK, K. Puzzle - on the Complexity of a Path Avoiding Forbidden Pairs of Vertices. In *3rd World Conference on Innovation and Computer Science (INSODE-2013)*. Antalya : Global Journal on Technology, 2013.

MILKOVÁ, E. & PETRÁNEK, K. Restricted Version of Paths Avoiding Forbidden Pairs – Complexity and Algorithm. *Recent Patents on Computer Science*, 2016b, vol. 2017, no. 10. ISSN 1874-4796.

MILKOVÁ, E., PETRÁNEK, K. & JANEČKA, P. Programming capabilities evaluation. In *Efficiency and Responsibility in Education (ERIE 2012)*. Prague : Czech University of Life Sciences Prague, 2012, pp. 310–318.

PETRÁNEK, K. & JANEČKA, P. Using Assignment Models to Analyze Programming Performance. In *Efficiency and Responsibility in Education 2014*. Czech University of Life Sciences Prague Faculty of Economics and Management, 2014a, pp. 571–576. ISBN 978-80-213-2468-8.

PETRÁNEK, K., JANEČKA, P. & MILKOVÁ, E. Testing Programming Aptitude: An in-Depth Analysis. In *Efficiency and Responsibility in Education (ERIE 2013)*. Prague, 2013a, pp. 497–502.

PETRÁNEK, K., JANEČKA, P. & VANĚK, J. Using Local Binary Patterns for Object Detection in Images. *Global Journal of Computer Sciences*, 2015, vol. 5, no. 1, pp. 7–12. ISSN 2301-2587.

PETRÁNEK, K., VANĚK, J. & JEŽEK, B. Local Binary Patterns a jejich použití v počítačovém vidění. In *Sborník příspěvků 32. konference o geometrii a grafice*. Plzeň: Vydavatelský servis, 2012, pp. 173–179. ISBN 978-80-86843-40-7.

PETRÁNEK, K., VANĚK, J., JEŽEK, B. & JANEČKA, P. Akcelerace algoritmů pro segmentaci obrazu. In *Sborník příspěvků 31. konference o geometrii a počítačové grafice*. Ostrava-Poruba: Vysoká škola báňská – Technická univerzita Ostrava, 2011, pp. 189–197. ISBN 978-80-248-2524-3.

PETRÁNEK, K., VANĚK, J., JEŽEK, B. & JANEČKA, P. Detekce pozadí scény při stereoskopickém snímání. In *Sborník příspěvků 30. konference o geometrii a grafice*. Praha: Matfyzpress, 2010, pp. 195–200. ISBN 978-80-7378-136-1.

PETRÁNEK, K., VANĚK, J., JEŽEK, B. & JANEČKA, P. Registrace GPS dat na silniční síť. In *Sborník příspěvků 34. konference o geometrii a grafice*. Vlachovice : Jihočeská univerzita v Českých Budějovicích, 2014b, pp. 199–204. ISBN 978-80-7394-470-4.

PETRÁNEK, K., VANĚK, J., JEŽEK, B. & JANEČKA, P. Robustní porovnání obrazů pomocí vzájemné informace. In *Sborník příspěvků 33. konference o geometrii a grafice*. Horní Lomná : Vysoká škola báňská - Technická univerzita v Ostravě, 2013b, pp. 199–203. ISBN 978-80-248-3251-73.

PETRANEK, K., VANEK, J. & MILKOVA, E. Avoiding the Curse of Dimensionality in Local Binary Patterns. In *Computational Collective Intelligence*. Lecture Notes in Computer Science. Springer, Cham, 2016, pp. 208–217. ISBN 978-3-319-45242-5.

TOBOLA, J., VANĚK, J., PETRÁNEK, K. & JEŽEK, B. Triviální algoritmus pro interaktivní simulaci vody. In *Sborník příspěvků 34. konference o geometrii a grafice*. Vlachovice : Jihočeská univerzita v Českých Budějovicích, 2014, pp. 239–246. ISBN 978-80-7394-470-4.