TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Odhad čekací doby pomocí zpracování obrazu

Bakalářská práce

Studijní program: B2646 – Informační techologie
Studijní obor: 1802R007 – Informační technologie

Autor práce: **Matěj Chumlen**
Vedoucí práce: Mgr. Jiří Vraný, Ph.D.

Liberec 2019

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

Queue Time Estimation by Image Processing

Bachelor thesis

Study programme: B2646 – Information technology
Study branch: 1802R007 – Information technology

Author: **Matěj Chumlen**
Supervisor: Mgr. Jiří Vraný, Ph.D.

Liberec 2019

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Zadání bakalářské práce

Odhad čekací doby pomocí zpracování obrazu

Jméno a příjmení: **Matěj Chumlen**
Osobní číslo: M16000190
Studijní program: B2646 Informační technologie
Studijní obor: Informační technologie
Zadávající katedra: Ústav nových technologií a aplikované informatiky
Akademický rok: **2018/2019**

Zásady pro vypracování:

1. Seznamte se s aktuálním stavem vývoje algoritmů vhodných pro detekci objektů v obraze.
2. Vybrané algoritmy porovnejte z hlediska přesnosti, náročnosti implementace, výpočetní náročnosti a náročnosti na množství i kvalitu tréninkových dat. Na základě porovnání vyberte nejvhodnější algoritmus pro praktickou implementaci.
3. S využitím zvoleného algoritmu implementujte prototyp systému, který umožní odhadnout délku čekání osob ve frontě na základě rozpoznávání čekajících osob ve video záznamu.

Rozsah grafických prací:	dle potřeby
Rozsah pracovní zprávy:	30 – 40 stran
Forma zpracování práce:	tištěná/elektronická

Seznam odborné literatury:

[1] GOODFELLOW, Ian, Yoshua BENGIO a Aaron COURVILLE. Deep learning. Cambridge, Massachusetts: The MIT Press, [2016]. ISBN 02-620-3561-8.
[2] DAVIES, E. R. Computer vision: theory, algorithms, practicalities. 5th edition. Cambridge, CA: Elsevier, 2017. ISBN 978-0128092842.
[3] ANGELOVA, Anelia, Alex KRIZHEVSKY, Vincent VANHOUCKE, Abhijit OGALE a David FERGUSON. Real-Time Pedestrian Detection With Deep Network Cascades. In: Proceedings of BMVC 2015 [online]. 2015, s. 12 [cit. 2018-10-04]. Dostupné z: https://ai.google/research/pubs/pub43850
[4] TOM?, D., et. al. Deep Convolutional Neural Networks for pedestrian detection. Signal Processing: Image Communication. 2016, (47), 482-489.
[5] SHAOQING R., KAIMING H., ROSS G., and JIAN S., Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Trans. Pattern Anal. Mach. Intell. 39, 6 (June 2017), 1137-1149. arXiv:1506.01497 [cs.CV]

Vedoucí práce:	Mgr. Jiří Vraný, Ph.D. Ústav nových technologií a aplikované informatiky
Datum zadání práce:	18. října 2018
Předpokládaný termín odevzdání:	30. dubna 2019

L. S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

Ing. Josef Novák, Ph.D.
vedoucí ústavu

V Liberci 18. října 2018

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum:

Podpis:

Abstrakt

Tato práce řeší všeobecný problém čekání ve frontě. Cílem práce je navrhnout systém, který na základě obrazové informace z videozáznamu fronty odhadne čekací dobu zákazníka. Tento problém byl řešen za pomoci objektového detektoru založeného na konvolučních neuronových sítích *Faster R-CNN* a trackování pomocí KCF trackeru. Ze zjištěných informací o časech průchodu a počtu čekajících je vypočten odhad čekací doby. Vytvořené řešení je využitelné v optimálních podmínkách. Jeho funkčnost závisí na umístění kamery vzhledem ke scéně, velikosti scény a kvalitě obrazu. Navržený systém by mohl pomoci ke zpříjemnění čekání ve frontě.

Klíčová slova: rozpoznávání obrazu, počítačové vidění, neuronové sítě, fronta, detekce osob, čas čekání, trackování, objektový detektor, Faster R-CNN

Abstract

This work addresses common problem of queuing. The goal of the work is to propose a system that estimates queuing time from video. The proposed solution is based on convolutional neural network based object detector *Faster R-CNN* and KCF tracker. The queuing time is then estimated based on knowledge of serving time per person and number of waiting people. The proposed solution is usable in optimal conditions. Its functionality depends on the position of the camera relative to the scene, scene size and image quality. The proposed system could help to make waiting in the queue more enjoyable.

Keywords: image recognition, computer vision, neural networks, queue, pedestrian detection, waiting time, tracking, object detector, Faster R-CNN

Poděkování

Rád bych poděkoval vedoucímu mé práce Ing. Jiřímu Vranému Ph.D. za podporu, rychlé reakce na dotazy a vstřícné konzultace. Dále bych rád poděkoval Ing. Karlu Palečkovi Ph.D. za tipy a rady z oblasti strojového učení a Ing. Josefu Chudobovi Ph.D. za konzultaci ke statistické části práce.

V neposlední řadě patří můj dík mé rodině, přátelům a mé přítelkyni, která měla pochopení pro to, že jsem měsíce zavřený v jedné místnosti, není se mnou řeč a nevycházím.

Obsah

Seznam zkratek

FPS	snímky za sekundu
HOG	histogram orientovaných gradientů
MLP	vícevrstvý perceptron
SVM	metoda podpůrných vektorů
RPN	síť pro návrh regionů
IoU	Jaccardův koeficient
TP	skutečně pozitivní
FP	falešně pozitivní
FN	falešně negativní
P	přesnost
R	úplnost
AP	průměrná přesnost
AR	průměrná úplnost
mAP	střední průměrná přesnost
mAR	střední průměrná úplnost
ILSVRC	ImageNet Large Scale Visual Recognition Competition
ECCV	European Conference on Computer Vision
SIFT	Scale-invarient feature transform

Seznam obrázků

1 Úvod

Úloha detekce osob zaznamenává v souvislosti s rozvojem v oblasti samořiditelných automobilů, robotiky a sledování osob bouřlivý vývoj. Zatímco klasické detekční metody jako *HOG* dosahovaly na svou dobu dobrých výsledků, za přelom v oblasti detekce lze považovat až rozvoj konvolučních neuronových sítí.

Ačkoliv jedno z prvních využití konvolučních neuronových sítí pro klasifikaci se datuje do roku 1989, kdy Yann LeCun aplikoval výcevrstvou neuronovou síť na problém rozpoznávání ručně psaných číslovek [2], výrazněji se začaly využívat po roce 2012, kdy síť AlexNet [3] dosáhla v soutěži ILSVRC o více než 10 procent nižší top-5 chyby než ostatní soutěžní vstupy. To bylo umožněno využitím grafických procesorů.

Tato práce se zabývá návrhem prototypu systému, který by s využitím detekčních algoritmů umožnil odhadnout čekací dobu ve frontě. Takový systém může významně pomoci s optimalizací využívání zdrojů v provozech jako jsou letiště, supermarkety, menší obchody a další služby a tím zpříjemnit zákaznický prožitek při udržení co nejnižších nákladů. Jeho nasazení počítá s využitím stávajících kamerových systémů. Ačkoliv podobné systémy již existují (viz kapitola 7), jedná se o komerční produkty. Dokumentace těchto produktů nenabízí informace o technologii použité při implementaci, ani informace o úspěšnosti.

V rámci této práce jsou prozkoumány existující algoritmy, které jsou pro řešení problému odhadu čekací doby z videozáznamu potřebné (8.2). Implementace těchto algoritmů jsou následně otestovány a porovnány z hlediska přesnosti a výpočetní náročnosti. Za jejich využití je implementován prototyp systému pro odhad čekací doby(8.3).

Potenciál navrhovaného systému spočívá v široké škále provozů, kde může být uplatněn, vzhledem k relativně malým nákladům, které jsou s tím spojeny.

Kapitola 2 se zabývá formálním vymezením fronty. V kapitole 3 jsou popsány klasické přístupy řešení úlohy detekce osob a v kapitole 4 jsou stručně představeny neuronové sítě a současné detekční algoritmy. Kapitola 5 popisuje používané datasety pro objektovou detekci. Knihovnami pro strojové učení se zaobírá kapitola 6.

2 Teorie hromadné obsluhy

Za frontu můžeme obecně označit takový systém, ve kterém zákazníci (osoby, požadavky, automobily atp.) čekají na obsloužení omezeným množstvím obslužných uzlů (bezpečnostní pás, pokladna, hraniční kontrola, čerpací stanice). Formálním ukotvením fronty se zabývá odvětví aplikované matematiky označované Teorie hromadné obsluhy. Jeho cílem je navrhnout takový systém, který optimalizuje množství prostředků nutné pro obsloužení zákazníků. Ve svých závěrech vychází ze statistiky a teorie pravděpodobnosti.

Zákazníci mohou být obsluhováni podle několika různých klíčů - frontových režimů. Nejběžnější systém obsluhy je založen na principu FIFO - příchozí, který čeká ve frontě nejdelší dobu je obsloužen nejdříve. Fronta může být obsluhována jako zásobník LIFO (skladovací systémy). Poslední příchozí je obsloužen jako první. Dalším rozšířením těchto přístupů může být zavedení priorit. Zákazník s vyšší prioritou je obsloužen přednostně (model používaný na letištích při bezpečnostní kontrole i boardingu). Pořadí obsloužení může také záviset na době jeho trvání. Zákazník, jehož obsloužení bude trvat nejkratší dobu, dostane přednost (expresní fronty v obchodech).

Frontu lze dále rozlišovat podle jejího uspořádání vzhledem k obslužným uzlům. Lze uvažovat frontu s jedním uzlem obsluhy, kde má obsluha jednu fázi (např. drogerie), nebo více fází (např. automyčka). Další typ fronty je fronta s více uzly obsluhy, kdy zákazník využije jakýkoliv uzel, který je právě nevytížený (např. čekání na pisoáry).

„Teorie front se snaží popsat výkonnost systému obsluhy popsaného následujícími náhodnými veličinami: počet zákazníků v systému, počet čekajících zákazníků, čas trvání obsluhy zákazníka, čas, po který uzel obsluhy není vytížený a čas po který uzel obsluhy vytížený je.“ [4]

Pro popis a klasifikaci front se v teorii hromadné obsluhy využívá Kendallova notace, která popisuje systém fronty pomocí šesti znaků:

$$A/B/m/K/n/D \tag{2.1}$$

- A - distribuční funkce časů mezi příchody
- B - distribuční funkce času obsluhy jednoho zákazníka
- m - počet bodů obsluhy
- K - kapacita systému

- n - velikost populace
- D - režim fronty

První tři znaky jsou zadané standardně. Pokud nejsou zadané zbylé tři, předpokládá se, že K a n jsou nekonečné. Výchozí režim fronty D je FIFO.

Předpokladem funkce modelů systémů hromadné obsluhy je, že vstupní i výstupní tok požadavků je stacionární proces s konzistentní intenzitou, kde jsou intervaly mezi příchody spojitá náhodná veličina se stejným rozdělením pravděpodobnosti. Využívá se následující notace:

- λ - intenzita vstupního toku - střední hodnota počtu vstupujících zákazníků za časovou jednotku
- μ - intenzita obsluhy - střední počet zákazníků obsloužených jednou linkou
- $\rho = \lambda/\mu$ - intenzita provozu

3 Klasický přístup detekce osob

Detekce osob v obraze je podmnožinou úlohy detekce objektů v obraze. Tato kapitola se věnuje krokům, které vedou k detekci. Tyto kroky jsou s obměnami společné všem metodám detekce. V prvním kroku jsou v obraze nalezeny kandidátní oblasti, které by potenciálně mohly obsahovat objekt. V dalším kroku jsou získány příznaky objektů navržených prvním krokem a na základě těchto příznaků je určena příslušnost ke třídě. V případě klasifikace do dvou tříd tedy *je objekt*, a nebo *není objekt*.

Obrázek 3.1: Roura detekce osob v obraze

3.1 Hledání kandidátních oblastí

Kvalitní metoda hledání kandidátních oblastí musí splňovat následující podmínky. Musí být výpočetně rychlá. Důvod jejího použití je snížení počtu klasifikovaných oblastí, protože klasifikace je standardně výpočetně náročná a je úzkým hrdlem detekční roury. Zároveň musí vést k co nejlepší úplnosti, tj. nevynechávat skutečně pozitivní oblasti.

3.1.1 Posuvné okénko

Naivní přístup by znamenal pixel po pixelu posouvat všechny možné rozměry okénka ohraničujícího kandidátní oblast. Takový přístup by však znamenal velkou výpočetní zátěž a byl by příliš pomalý. Tento přístup lze vylepšit na základě znalosti vlastností hledaných objektů. Je možné předpokládat určitou minimální a maximální velikost objektu v obraze a nějakou konečnou množinu poměrů stran, které může hledaný objekt mít (například obdélník s poměrem stran 1:3 bude reprezentovat osobu nebo automobil spíše, než obdélník s poměrem stran 1:20). Této metodě se říká *posuvné*

okénko a i přes popsané úpravy generuje příliš velké množství kandidátních oblastí. Zároveň se nesprávně zvoleným poměrem stran a velikostmi hledaných oblastí zvyšuje šance na ztrátu úplnosti nalezených objektů.

3.1.2 Edge Boxes

Jednou z metod využívajících informace z obrazu je metoda *Edge Boxes* [5]. Algoritmus využívá rychlosti hledání hran v obraze metodou Structured Edge. Pomocí posuvného okénka jsou následně ohodnoceny oblasti podle toho, jak uzavřenou hranu obsahují – jaká je šance, že obsahují celý objekt. Tyto oblasti jsou následně profiltrovány. V úloze detekce osob v obraze dosahuje podobně jako Selective Search dobrých výsledků [6].

3.1.3 Selective Search

Další možnou metodou je *Selective Search* [7]. Algoritmus vybírá pouze oblasti, v nichž je pravděpodobné, že obsahují objekt. K tomu využívá segmentaci obrazu. Obraz je segmentován podle hodnot v různých barevných prostorech, jak je popsáno v [8]. Jednotlivé segmenty jsou označeny ohraničujícím obdélníkem jako kandidátní oblasti. Následně jsou určeny podobnosti mezi sousedními regiony a dva sousední regiony, které jsou si nejpodobnější jsou sjednoceny. Tento hladový algoritmus se opakuje, dokud není celý obraz jedním regionem. Výsledkem je několikanásobně menší množství kandidátních regionů, což znamená menší výpočetní náročnost klasifikace. Zároveň tento algoritmus nestanovuje pevně daná měřítka objektů a poměry stran, což znamená, že dosahuje lepší úplnosti.

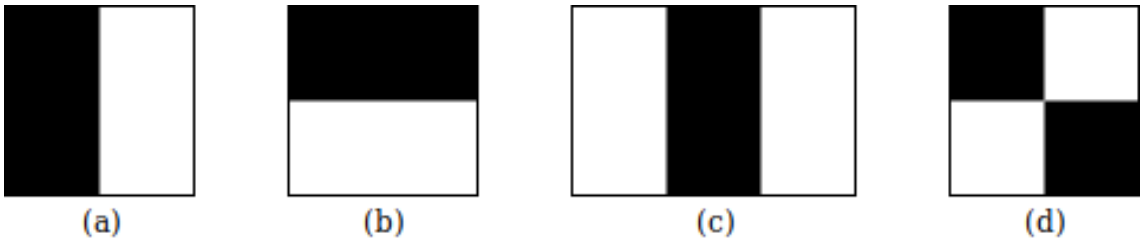
Obrázek 3.2: Lokalizace objektů pomocí metody Selective Search [7]

3.2 Extrakce příznaků

Poté, co jsou nalezeny kandidátní oblasti je třeba z nich získat informace, které jsou následně využity pro klasifikaci. Tedy určení příslušnosti ke třídě.

3.2.1 Haarovy příznaky

Haarovy příznaky jsou příznaky, které využívají charakteristických jasových rozdílů mezi oblastmi objektu. Pomocí masek různých velikostí (3.3 klasifikátor získává informace o přítomnosti hran (a, b), nebo linek (c, d) v obraze. Nejznámější aplikaci Haarových příznaků je detektor *Viola-Jones* [9], který je primárně určen k rozpoznávání obličejů. Úspěšnost klasifikace založené na Haarových příznacích závisí na úhlu záběru a klasifikátor je málo robustní vůči jasovým rozdílům a natočení objektu.



Obrázek 3.3: Haarovy příznaky

3.2.2 HOG

Větší úspěšnosti při detekci osob dosahuje metoda *HOG* (histogram orientovaných gradientů). Metoda využívá informace o velikosti a směru změny intenzity jasu v obraze. Na obraz je aplikován postupně Sobelův hranový detektor s jádrem velikosti 1 pro vodorovné a svislé hrany. Následně je pro každý bod v obraze spočítána velikost a směr gradientu podle vztahu:

$$g = \sqrt{g_x^2 + g_y^2} \tag{3.1}$$

$$\theta = \arctan \frac{g_y}{g_x} \tag{3.2}$$

Obraz je poté rozdělen na čtvercové oblasti $n \times n$. Jejich velikost je zvolena podle velikosti objektu. V těchto čtvercích je vypočten histogram gradientů, čímž se informace obsažená v obraze významně zredukuje. Histogramy jsou následně normalizovány v bloku $k \times k$ sousedních histogramů. Tyto bloky se vzájemně překrývají. Tím je docíleno větší robustnosti vůči nerovnoměrnému osvětlení scény. Normalizované histogramy jsou spojeny do finálního vektoru, který představuje vektor příznaků pro klasifikaci. Tato metoda je robustnější vůči jasovým nerovnostem, ale předpokládá vzpřímenou polohu osob a jejich dostatečnou viditelnost [10].

3.3 Klasifikace příznaků metodou SVM

Metoda podpůrných vektorů *SVM* je metodou strojového učení s učitelem. Používá se ke klasifikaci. Podstata klasifikace spočívá v rozdělení prostoru příznaků na dva

poloprostory, které obsahují data odlišných klasifikovaných tříd takovým způsobem, který maximalizuje minimální vzdálenost bodů od dělící nadroviny, kterou lze popsat rovnicí

$$\vec{w} \cdot \vec{x} + b = 0, \tag{3.3}$$

kde $\vec{w}$ je normála dělící nadroviny, $\vec{x}$ vektor příznaků a b posun dělící nadroviny od počátku soustavy souřadnic. Maximalizace vzdálenosti probíhá učením klasifikátoru na trénovacích datech, u kterých je známa příslušnost ke třídě. Výsledkem učení je vektor $\vec{w}$ a bias b. Pomocí jádrových funkcí lze použít SVM k nelineární klasifikaci a existují modifikace metody pro klasifikaci do více tříd. SVM je díky rozvolňujícím proměnným robustní vůči šumu a dosahuje dobrých výsledků na malých datech.

4 Neuronové sítě

4.1 Umělá neuronová síť

Umělá neuronová síť je výpočetní model, který je inspirován biologickou neuronovou sítí lidí a zvířat. Je to systém, který se podobně jako mozek učí vykonávat danou funkci podle příkladů, aniž by bylo třeba ho k tomu předprogramovávat. Umělá neuronová síť nachází uplatnění v situacích, kde nestačí použít rozhodovací stromy založené na jasně definovaných podmínkách.

Základním stavebním prvkem umělé neuronové sítě je podobně jako u biologické neuronové sítě neuron. Každý neuron dokáže zpracovat vstupní signál z jiných neuronů a vyslat výstupní signál dál. Schéma umělého neuronu popisuje obrázek 4.1.

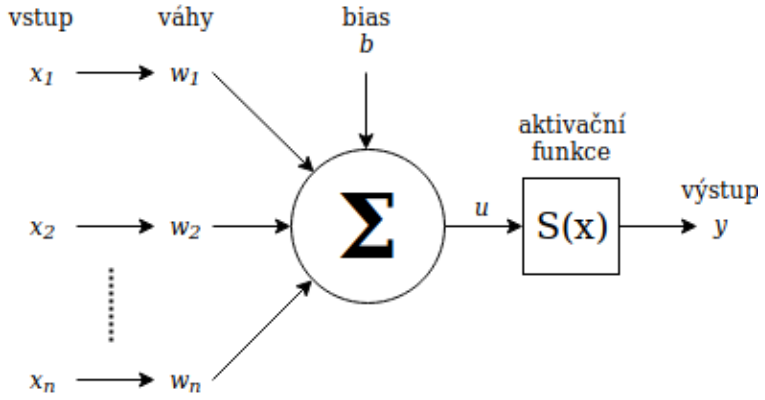
Každý neuron přijme vektor vstupů $\vec{x}$, který vynásobí vektorem vah $\vec{w}$, k sumě výsledku je přičten práh (bias). Výstup u je předán aktivační funkci. Aktivační funkce představuje nelinearitu neuronové sítě. Může být reprezentována například jednotkovým skokem, sigmoidou nebo funkcí ReLU. Výstup aktivační funkce y je předán dále do sítě. Funkci neuronu lze popsat následující rovnicí:

$$y = S(\sum_{i=1}^{N}(w_i x_i) + b) \tag{4.1}$$

Neurony jsou v neuronové síti typicky uspořádány do vrstev. Mezi vstupní a výstupní vrstvou se nachází skryté vrstvy. Síť s alespoň jednou skrytou vrstvou se nazývá vícevrstvý perceptron (MLP). Pokud obsahuje model sítě dvě a více skrytých vrstev, jedná se o hlubokou neuronovou síť.

Procesem učení neuronové sítě se rozumí hledání optimální sady parametrů modelu, kterými jsou váhové vektory a biasy jednotlivých neuronů. Optimální parametry jsou takové, které pro vstup x (nezávislá proměnná) vrací správný výstup y (závislá proměnná, predikce). Učení probíhá za pomoci trénovacích dat, což je soubor vstupních dat a očekávaných výstupních dat. Úspěšnost neuronové sítě je vyjádřena pomocí cost funkce. Cílem učení je minimalizovat její hodnotu.

Jelikož nelze analyticky určit parametry, pro které dosahuje cost funkce minima, probíhá hledání minima iterativně pomocí metody gradient descent.



Obrázek 4.1: Schéma umělého neuronu

4.2 Konvoluční neuronové sítě

Konvoluční neuronová síť představuje zvláštní typ neuronové sítě. Podobně jako síť typu MLP má vstupní vrstvu, skryté vrstvy a výstupní vrstvu s vahami a biasy. Na rozdíl od MLP, kde dochází na vstupní vrstvě k převedení obrazu na 1D vektor, je konvoluční síť schopna zachycovat prostorové vlastnosti obrazu (hrany, linie, rohy atp.). Základem tohoto principu je využití konvolučních filtrů, které představují parametry konvoluční vrstvy. Jde o analogický přístup jako u Haarových příznaků. Oproti tradičnímu přístupu k detekci však nejsou jednotlivé filtry vytvořeny ručně na základě předchozí znalosti vlastností hledaného objektu. Jejich podoba vzniká v průběhu učení sítě. Díky těmto vlastnostem jsou konvoluční neuronové sítě vhodným druhem umělých neuronových sítí pro klasifikaci obrazu.

Konvoluční neuronové sítě běžně sestávají z několika druhů vrstev. Konvoluční vrstva převádí vstup, což může být vstupní obraz nebo výstup z vyšší vrstvy, na příznakovou mapu za pomocí filtrů popsaných výše. Zatímco první konvoluční vrstva detekuje vysoce abstraktní příznaky, jako jsou hrany, každá další konvoluční vrstva na základě informace z vyšších vrstev detekuje méně abstraktní příznaky: kružnice, kola, automobil, bicykl. Výstup z konvoluční vrstvy prochází aktivační funkcí.

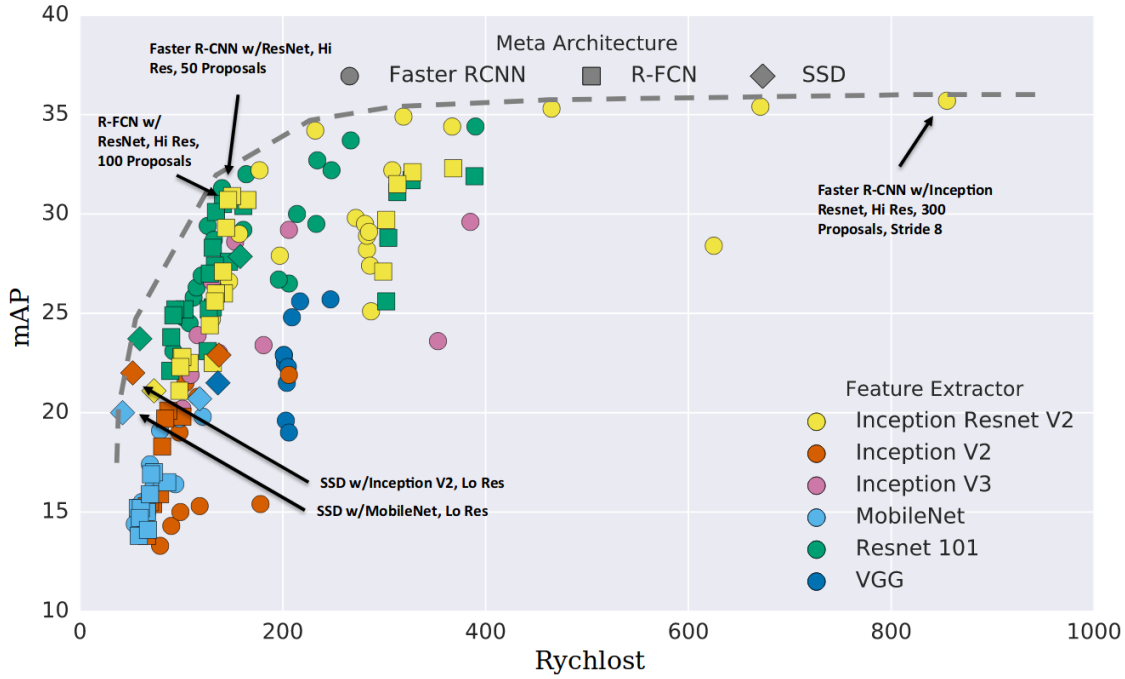
Za účelem redukce velikosti vstupních příznaků obsahuje konvoluční neuronová síť sdružovací vrstvy. Sdružovací vrstvy negenerují žádné vlastní příznaky. Místo toho prochází vstupní pole pomocí kernelu 2×2 s krokem 2, a buď zachovají největší hodnotu (max pooling), nebo průměrnou hodnotu (average pooling) z hodnot. Touto operací dojde k redukci vstupních dat o 75%. Vedle výrazného zmenšení velikosti parametrů se zmenšuje riziko přeučení sítě.

Konvoluční neuronová síť plní úlohu extraktoru příznaků. Běžně je za ní zařazeno několik plně propojených vrstev neuronů, které příznaky dále zredukují pro klasifikaci. Klasifikace je následně provedena pomocí klasifikátoru jako softmax, nebo SVM.

4.3 Neuronové sítě pro objektovou detekci

V následující části je představeno několik objektových detektorů. Jsou vybrané detektory, které jsou populární, hojně popsané a snadno dostupné jako open-source.

Popsané detektory lze rozdělit na dvě skupiny. Jednodušší, tzv. one-stage metody (*SSD, YOLOv3*), které celou detekci provádí během jednoho přímého průchodu sítí. Tyto metody jsou zpravidla rychlejší a výpočetně méně náročné. Druhou skupinou jsou tzv. two-stage metody (*Faster R-CNN, Mask R-CNN, R-FCN*).



Obrázek 4.2: Porovnání metod objektové detekce [11]

4.3.1 Faster R-CNN

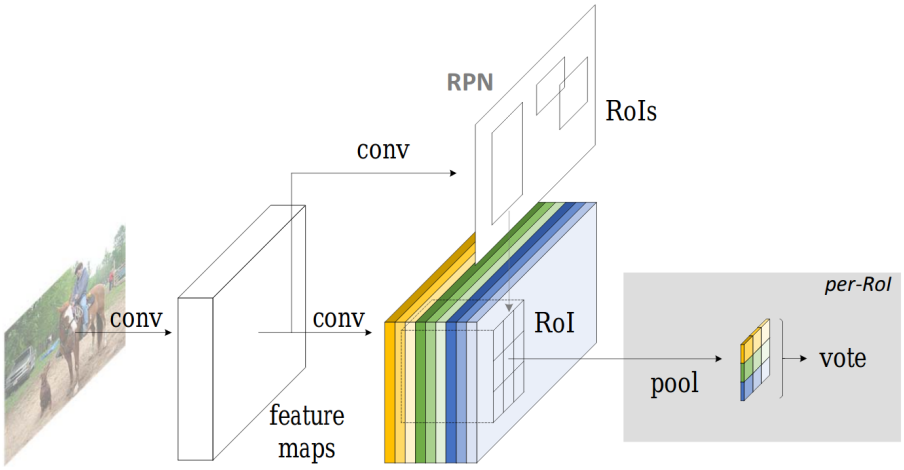
Faster R-CNN (Region-based convolutional neural network) je zdokonalením starších metod *R-CNN* a *Fast R-CNN*. Oproti svým předchůdcům nevyužívá výpočetně náročnou metodu hledání kandidátních oblastí Selective Search (3.1.3), která představovala výpočetní úzké hrdlo. Místo toho zapojuje konvoluční síť s několika málo vstvami (RPN) pro binární klasifikaci (objekt × neobjekt). Vstupní obraz prochází několika konvolučními vrstvami (extraktor příznaků). Výstupem těchto vrstev je příznaková mapa s optimální mírou abstrakce pro RPN. Ve vstupním obraze RPN je pomocí posuvného okna vybráno pro každý obrazový bod k oblastí o různých poměrech stran a velikostech. Pro každou oblast je predikováno s jakou pravděpodobností a jak přesně reprezentuje oblast s objektem. Oblasti s dostatečnou jistotou predikce jsou předány metodě Non–max Suppression (NMS) za účelem vyfiltrování překrývajících se predikcí. Výstupní predikce jsou předány ke klasifikaci.

4.3.2 Mask R-CNN

Mask R-CNN je rozšířením detektoru Faster R-CNN. Zatímco v první fázi dochází shodně k hledání kandidátních oblastí pomocí RPN, v druhé fázi jsou kandidátní oblasti kromě klasifikace podrobeny segmentaci. Výstupem segmentace je binární maska objektu.

4.3.3 R-FCN

R-FCN (Region-based fully convolutional network) adresuje problém výpočetní náročnosti klasifikace u detektorů typu R-CNN. Ty pro klasifikaci každé kandidátní oblasti vyžadují samostatný průchod klasifikační sítí. Tato výpočetně náročná vlastnost je v R-FCN odstraněna sdílením konvolučních vrstev i pro klasifikaci. Výsledkem tohoto přístupu je výrazné zrychlení oproti *Faster R-CNN* [12]. Podobně jako u *Faster R-CNN* je na příznakové mapě na výstupu extraktoru příznaků pomocí RPN vybrán soubor kandidátních regionů. Na následující vrstvě je vytvořena sada map o počtu $k^2 \times (C + 1)$. Pro každou z C tříd a pozadí (+1) je vytvořeno k^2 map skóre. Každá mapa reprezentuje skóre jedné z podoblastí objektu v mřížce $k \times k$. Průměrná hodnota všech podoblastí v mřížce představuje skóre pro jednotlivé třídy. Tato skóre jsou vstupem klasifikace pomocí funkce softmax.



Obrázek 4.3: Architektura R-FCN [12]

4.3.4 YOLOv3

YOLOv3 (You Only Look Once) využívá podobného přístupu jako RPN síť u předchozích detektorů. Na kandidátních oblastech však namísto binární klasifikace objekt × neobjekt provádí navíc i klasifikaci do tříd. Detekce je prováděna na třech různých měřítkách, aby bylo dosaženo dobré úspěšnosti na různě velkých instancích objektů. Na každém z měřítek je obraz rozdělen na mřížku. Každý prvek mřížky generuje 3 různé kandidátní oblasti popsané pomocí souřadnic pravého horního

rohu, šířky a výšky. Pro každou kandidátní oblast je určeno s jakou jistotou se jedná o objekt a skóre všech klasifikovaných tříd. Následně je provedeno prahování kandidátních oblastí. Výstupem jsou ohraničující obdélníky nalezených objektů a nejpravděpodobnější třída.

4.3.5 SSD

SSD (Single Shot MultiBox Detector) podobně jako YOLOv3 generuje kandidátní oblasti s různými poměry stran na mřížce a pro každou oblast určuje skóre jednotlivých tříd. Aby bylo dosaženo lepší úplnosti napříč měřítky objektů, jsou kandidátní oblasti vybírány z různých vrstev konvoluční sítě. Klasifikace probíhá funkcí softmax.

4.4 Hodnocení úspěšnosti detekčních algoritmů

Úloha detekce se sestává ze dvou dílčích úloh. Úlohy lokalizace objektu v obraze a úlohy stanovení příslušné třídy objektu. Aby bylo možné určit míru úspěšnosti detekčních algoritmů, je třeba určit hodnotící kritérium těchto úloh.

Nejčastěji využívaným [13][14] hodnocením úspěšnosti lokalizace je Jaccardův koeficient podobnosti, běžně označovaný jako IoU (Intersection over Union). Koeficient je definovaný jako

$$IoU = \frac{(B_p \cap B_{gt})}{(B_p \cup B_{gt})}, \tag{4.2}$$

kde B_p je predikovaný ohraničující obdélník objektu a B_{gt} očekávaný (správný) ohraničující obdélník. Pokud je hodnota IoU větší, než stanovený práh (typicky jsou používány meze *0,5; 0,75; 0,95*), je lokalizace považována za úspěšnou.

Jistota příslušnosti objektu ke třídě je vyjádřena pravděpodobností. Úspěšnost detekce na testovacích datech je určována vzhledem k míře jistoty, při které je přijata klasifikace objektu jako správná.

Aby bylo možné určit úspěšnost detekce napříč testovacími daty, je třeba hodnotit dvě kritéria. Přesnost a úplnost. Přesnost je definována jako:

$$P = \frac{TP}{TP + FP} \tag{4.3}$$

Úplnost jako:

$$R = \frac{TP}{TP + FN} \tag{4.4}$$

Kde TP představuje počet objektů, které byly úspěšně detekovány (True Positive). FP Je počet objektů, které byly detekovány chybně. Tedy se v obraze nenachází (False Positive). FN Je počet objektů, které se v obraze nachází, ale nebyly detekovány.

Úplnost a přesnost jsou na sobě nepřímo závislé. Pro klesající přesnost dochází k nárůstu úplnosti a opačně. Tento fenomén postihuje míra AP (Average Precision), kterou lze obecně vypočítat jako:

$$AP = \int_0^1 p(r)dr \tag{4.5}$$

Kde $p(r)$ je funkce vyjadřující závislost přesnosti na úplnosti. Tento vztah je v praxi nahrazen interpolací a funkce závislosti je před výpočtem vyhlazena.

K výpočtu AP dochází v rámci jedné třídy. V případě detekce více tříd dochází k další úpravě. Míra úspěšnosti je pak popsána pomocí mAP (mean Average Precision), která je vypočtena jako průměr přes AP všech tříd. Termíny AP a mAP jsou často zaměňovány.

AR je průměrná úplnost na trénovacích datech přes všechny snímky trénovacích dat a mAR průměrná AR přes všechny klasifikované třídy [15].

5 Datasety pro objektovou detekci

Úspěšnost detekčního systému závisí na vhodně navrženém modelu a nalezení vhodných parametrů modelu ve fázi jeho trénování. Úloha detekce je náročná na trénovací data z důvodu rozmanitosti reálných scén [3]. Známé datasety jako MNIST nebo CIFAR-10 jsou určeny pouze pro klasifikaci a obsahují malé množství tříd. Shromáždění datasetu pro klasifikaci a lokalizaci, který by byl robustní co se týče rozmanitosti scén z pohledu orientace objektů a osvětlení, představuje velkou časovou zátěž. Stejně tak následná anotace instancí objektů. Je tedy nevhodné, aby každá práce v oblasti strojového vidění začínala touto činností. Tyto důvody podnítily vznik datasetů, které jsou nezávislé a zároveň umožňuje standardizované testování detekčních systémů a stanovení nejlepšího z nich. Vybrané z nich popisuje následující sekce.

5.1 Pascal VOC

Jedním z prvních a dodnes hojně využívaných datasetů je *Pascal VOC* (M. Everingham et al.) [13], který vznikl v roce 2005. Výsledky detekce na Pascal VOC byly porovnávány každoročně vrámci soutěže VOC Challenge mezi lety 2005 a 2012. Vedle detekce byly soutěženy úlohy segmentace a klasifikace akcí. Detekční dataset obsahuje v poslední verzi 20 tříd, které jsou zastoupeny 27450 instancemi na 11540 snímcích. Snímky v datasetu jsou náhledy shromážděných fotografií z webu Flickr, který poskytuje komunitní sdílení fotografií svým uživatelům.

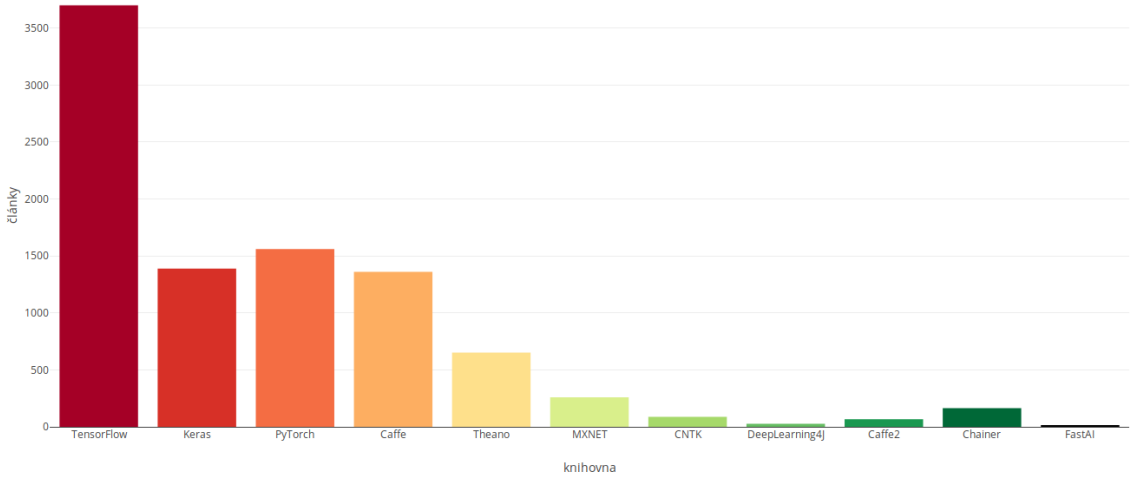
5.2 ImageNet

V současné době nejrozsáhlejší databází fotografií pro výzkum v oblasti strojového vidění je *ImageNet*. Tato databáze byla poprvé představena v roce 2009. Na jejím vývoji se podílí především Stanfordova univerzita a Princetonská univerzita. Databáze obsahuje přes 14 milionů náhledů fotografií, které pochází z webu Flickr. Fotografie jsou uspořádány podle sítě WordNet, která seskupuje slova do tzv. synsetů. Pro detekční úlohu jsou z WordNet vybrána podstatná jména. Těch je více než 80000. ImageNet obsahuje anotované fotografie pro 21841 z nich (mj. 120 ras psů). Pro úlohu detekce objektů je anotováno 1034908 fotografií. Podobně jako Pascal VOC je i ImageNet předmětem soutěže - ILSVRC. Pro soutěžní úlohu lokalizace je omezen počet tříd na 1000. Úloha detekce je omezena na 200 tříd.

5.3 COCO

Nejmladším datasetem ze zmíněných je *COCO*. Dataset COCO byl publikován v roce 2014 v článku „Microsoft COCO: Common Objects in Context“. Od předchozích se odlišuje tím, že anotace jsou doplněny o segmentace objektů na úrovni pixelů - každá instance objektu má svou masku. Dále obsahuje snímky, které byly vybrány podle specifického klíče. Na rozdíl od ImageNet a Pascal VOC, které obsahují kanonické zobrazení objektů (lidé en face atp.), COCO obsahuje scény z reálného prostředí, kde jsou objekty zachyceny z různých úhlů, jsou vzájemně okludovány a vytváří chaotické scény. Této vlastnosti tvůrci dosáhli tak, že namísto jednoho klíčového slova pro vyhledávání na Flickr využívají kombinace slov různých tříd. Dataset se zaměřuje na detekci 80 tříd. V těchto třídách je obsaženo všech 20 tříd z datasetu Pascal VOC, čímž je dosaženo zpětné kompatibility. Obsahuje 330 tisíc obrázků, z nichž je více než 200 tisíc anotováno pro objektovou detekci. Oproti výše zmíněným obsahuje COCO více instancí objektů na jeden obrázek. Zároveň pouze 10 procent scén obsahuje pouze jednu kategorii, oproti Pascal VOC a ImageNet, u kterých je to více než 60 procent. Vedle detekce objektů se COCO zaměřuje i na detekci pozice osob, generování popisků scén a další. Tyto úlohy jsou soutěženy v rámci konference ECCV.

6 Knihovny pro strojové učení



Obrázek 6.1: Počet zmínění knihoven na arXiv k 21. 8. 2018 podle [16]

6.1 TensorFlow

TensorFlow je open–source knihovna, která je vyvíjena společností Google. Je napsána v jazycích C++ a Python. Její výhodou je, že poskytuje API pro jazyky Python, JavaScript, Java, Go a Swift. TensorFlow zakládá na statických výpočetních grafech. To znamená, že před spuštěním výpočtu je nutné zadefinovat veškeré výpočetní operace a sestavit výpočetní graf. Díky tomu dochází při opakovaném spouštění výpočtu k úspoře výpočetního času. V důsledku toho však nejde v kódu používat *print* funkce, podmínky a cykly klasickým pythonovským způsobem a je potřeba používat speciální funkce z TensorFlow API. Tyto limitace souvisí s další slabou stránkou, kterou je těžší debugování navrženého modelu.

6.2 PyTorch

PyTorch je podobně jako TensorFlow open–source knihovna, která je zaměřená na strojové učení. Vývoj PyTorch zaštiťuje výzkumný tým společnosti Facebook. Narozdíl od TensorFlow využívá dynamické výpočetní grafy, které se sestavují za běhu

programu. To znamená, že lze používat podmínky, cykly a debugování klasickým způsobem. PyTorch verze 1.0.0 byl vydán 7. 12. 2018. Je tedy nejmladším ze jmenovaných knihoven. Díky záštitě Facebooku dochází ale k rychlému rozšiřování dokumentace, což byla nevýhoda oproti dobře dokumentovanému TensorFlow. Podle dat z [16] je popularita PyTorch stabilně rostoucí.

6.3 Keras

Keras je vysokoúrovňové API v Pythonu, které slouží jako front-end jiných knihoven pro strojové učení (TensorFlow, Theano, CNTK, MXNet, PlaidML). Keras cílí na uživatelskou přívětivost. Nedosahuje takového výkonu jako ostatní zmíněné knihovny a jeho záměr je rychlé prototypování a první seznámení s principy strojového a hlubokého učení.

6.4 Caffe

Caffe je knihovna vyvíjená výzkumníky z univerzity v Berkeley. Je napsána pro Python v jazyce C++. Modely druhé verze Caffe2 se dají využít v PyTorch. Vzhledem k tomu, že společnost Facebook používá jak Caffe2, tak PyTorch došlo v dubnu 2018 ke sloučení těchto dvou knihoven. Popularita Caffe klesá ve prospěch TensorFlow a PyTorch.

7 Rešerše existujících řešení

Nalezení optimálního způsobu obsluhy zákazníků s sebou nese mnoho výhod. Podnik může na základě této znalosti lépe plánovat směny zaměstnanců, to znamená, že zaměstnanci tráví čas v zaměstnání efektivně a nemají prostoje. Dále pak dochází k rychlejšímu obsloužení zákazníků, což v důsledku znamená méně zákazníků, kteří opustí frontu před obsloužením a více zákazníků, kteří se po příjemné zkušenosti vrátí a využijí služby znovu. Dále je v procesu optimalizace šance nalézt úzká hrdla celého systému. Pomalé zaměstnance nebo špatně navržené uspořádání podniku. Všechny tyto nabyté znalosti vedou ke zvýšení prodejnosti a snížení nákladů na provoz.

Problém hledání optima lze řešit několika způsoby. Analyticky pomocí metod z teorie hromadné obsluhy, pomocí simulace na základě nasbíraných dat, nebo v reálném čase.

Analytické řešení naráží na proměnlivost systému. Navržené analytické metody vyžadují pro své metody velmi specifické předpoklady, a proto je lze použít jen ve specifických případech. Nevýhodou simulačních metod je nutnost velkého množství vstupních dat [17]. Dále také cena simulačních nástrojů. Například licence softwaru SIMUL8 Professional stojí téměř 5000 dolarů.

Řešení, která využívají rozpoznávání obrazu mají výhodu v nízkých nákladech na montáž a hardware. Využívají stávající systém bezpečnostních kamer. Umožňují automatický sběr aktuálních dat pro reporting a využití ve výše zmíněných simulacích při určování parametrů vstupních náhodných veličin. Dokáží management a personál podniku v reálném čase informovat o nahromadění zákazníků a nutnosti otevřít další body obsluhy.

Komerčně dostupných řešení je na trhu velmi mnoho. Jsou limitující z různých důvodů. Například software TrueView Queue od společnosti Cognimatics, který je na českém trhu k dostání je možné používat pouze na IP kamerách Axis, což znamená zvýšení počátečních nákladů. Společnost Axis sama nabízí software Axis Queue Monitor, opět pouze pro své vlastní IP kamery. Software také předpokládá spolupráci s dalšími systémy z ekosystému této firmy. Na trhu jsou dostupná i řešení využívající stávající systém CCTV, jako například systém firmy Retail Sensing.

Bylo prozkoumáno velké množství existujících řešení s následujícími závěry. Velké množství nabízených systémů potenciálního uživatele limituje tím, že ho nutí využívat i další software nebo dokonce hardware stejného výrobce, čímž stírá výhody nastíněné výše. Žádné z prozkoumaných řešení neposkytuje na webových stránkách veřejně informaci o ceně. K žádnému z řešení využívajících bezpečnostní kamery není dostupná informace o principu jeho fungování, o použitých algoritmech nebo technologiích. Ve-

dle deklarativních prohlášení o úspěšnosti nabízené technologie chybí data, která by funkčnost prokazovala. Některé ze systémů nabízejí API pro komunikaci se systémy třetích stran, ale opět není veřejně k dispozici uspokojivá dokumentace.

Open-source řešení této problematiky se nepodařilo nalézt. Problém obsahující stejné dílčí kroky řeší IBM v repozitáři na GitHubu [1], ovšem detekční úlohu řeší pomocí své platformy IBM PowerAI Vision, která pracuje s velkou mírou abstrakce v cloudu a nastavení detektoru i použitý detekční algoritmus není z uživatelského GUI, kterým se platforma ovládá, možné ovlivnit.

Problém propojení kvalitní detekce a trackování, což je podstatná část problému, kterým se zabývá tato práce, řeší v repozitáři pod licencí MIT společnost Neuromation [18].

Projekt, který se nezabývá stejným tématem, ale obsahuje jeho dílčí části je také KERBEROS.IO [19], tento projekt je oproti předchozím zmíněným ucelenější. Pracuje jako webová aplikace s nástěnkou, na které je možné sledovat reporting. Primární účel projektu je však vytvoření dostupného zabezpečovacího zařízení pomocí Raspberry Pi s kamerou.

8 Tvorba prototypu systému pro odhad čekací doby ve frontě

Navrhovaný prototyp systému sestává z několika částí, které jsou na sobě jen omezeně závislé. Systém, který odhaduje čekací dobu ve frontě musí řešit následující dílčí problémy:

1. separace oblasti fronty v obraze
2. stanovení počtu čekajících osob ve frontě
3. stanovení rychlosti, jakou probíhá obsluha osob ve frontě

Sekce 8.2 se zabývá hledáním optimálních prostředků řešení těchto problémů. V následující sekci 8.3 je popsána navržená implementace systému pro odhad čekací doby.

8.1 Hardwarové a softwarové prostředky

Všechny údaje naměřené v této práci pochází z následující hardwarové a softwarové sestavy. Pro chod a správu virtuálních prostředí Python byla použita distribuce Anaconda Python a její balíčkovací systém Conda.

Hardware:

- RAM: 2 × 8GB SO-DIMM DDR4 2400MHz
- CPU: Intel Core i5-8300H
- GPU: Nvidia GP107M (GeForce GTX 1050 Mobile)
- SSD: Samsung MZVLW256HEHP

Software:

- Ubuntu 18.04.2 LTS
- CUDA 10.0.130
- cuDNN 7.4.2

- Python 3.6.8
- OpenCV 3.4.4
- PyTorch 0.4.1

Knihovna pro manipulaci s obrazem OpenCV dostupná v balíčkovacím systému Conda nepodporuje vykonávání funkcí na GPU, což je nutná vlastnost pro chod zvolených detekčních systémů. Zároveň neobsahuje oficiální distribuce moduly pro práci na GPU. Tyto moduly se nachází v repozitáři OpenCV Contrib, a tak bylo nutné ji zvlášť zkompilovat spolu se zmíněným repozitářem.

8.2 Výběr prostředků pro použití v systému

8.2.1 Testovací data

Hledání vhodných prostředků pro použití ve výsledném systému vyžaduje testovací data. Jedním z přístupů k jejich shromáždění bylo využití veřejně dostupných snímků front z internetu. Byly prozkoumány možnosti web scrapingu. Jako nejschůdnější se ukázala možnost použití služby Google Images, která indexuje snímky na internetu a poskytuje API pro jejich vyhledávání a filtrování. Podstatným parametrem filtru jsou práva pro další použití a rozlišení.

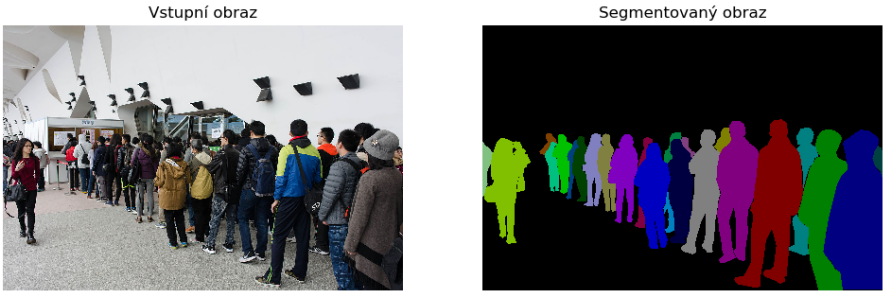
Pro dávkové stahování snímků byl použit volně dostupný program google-images-download [20], který využívá API Google Images a přidává vrstvu abstrakce pro pohodlné používání z příkazové řádky.

Bohužel bylo zjištěno, že obrázků front které splňují výše zmíněné parametry je relativně málo (desítky). Zároveň by bylo nutné dohledávat zdroj obrázků pro upřesnění licence pro opětovné použití, což je v mnoha případech obtížné. Velká část obrázků obsahovala ikonický záběr fronty, který neodpovídal realitě. Dále bylo pro testování trackerů nezbytné disponovat nejen snímky, ale i videozáznamy front. Tento postup byl tedy zamítnut ve prospěch vytvoření vlastních videozáznamů front.

Pro účely práce byla tedy dále vytvořena testovací data v podobě několika videozáznamů fronty (viz 8.2). Videozáznamy zachycovaly frontu z různých úhlů ve dvou variantách. První variantou byla optimální fronta s dobrou separací jednotlivých osob bez okluze. Fronta se pohybovala přímo. Druhá varianta zachycovala frontu bližší realitě: osoby v zákrytu, předbíhající osoby, osoby vybočující pohybem ze směru fronty.

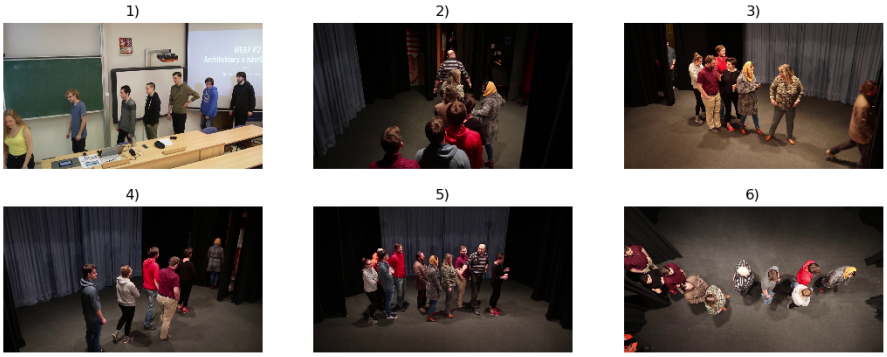
Z druhé varianty fronty byl navzorkován soubor snímků. Byl doplněn volně dostupnými snímky front z internetu a byla vybrána sada 20 snímků, které obsahovaly pro detekci a klasifikaci neoptimální scény. Osoby stojící ve frontě na těchto snímcích se značnou měrou zakrývají (např. snímky 2, 3, 5), nebo jsou viditelné z úhlu, který není v běžně používaných datasetech zaměřených na detekci osob (COCO, Caltech Pedestrian Dataset, INRIA) běžný (snímky 2 a 6). Na těchto snímcích bylo pomocí softwaru Labelme [21] označeno 220 instancí osob pomocí polygonů (viz 8.1).

V dalším kroku byl vytvořen testovací dataset. Vzhledem k oblíbenosti mezi výzkumníky a ucelenému způsobu vyhodnocování výsledků byl pro účely testování



Obrázek 8.1: Segmentace v softwaru Labelme

úspěšnosti detekčních modelů zvolen formát COCO výzkumného týmu firmy Microsoft. Software Labelme nenabízí možnost exportu všech anotovaných snímků a příslušných anotací do formátu, který by se dal využít na trénování nebo testování detekčního modelu. Tento krok bylo nutné zprogramovat. V rámci práce byl vytvořen skript, který na vstupu přijímá výstupní soubory s anotacemi ze softwaru Labelme ve formátu *.json a na výstupu vrací soubor anotací (*.json), který odpovídá specifikaci testovacího datasetu COCO pro detekci, tak jak je popsána v [22]. Tento dataset byl použit v další části práce na testování úspěšnosti detekce vybraných modelů.



Obrázek 8.2: Ukázka snímků z testovacích dat

8.2.2 Porovnání modelů detekce

Při rešerši bylo zjištěno, že tradiční klasifikační metody založené na ručně vytvářených příznacích (HOG, SIFT) nedosahují ani zdaleka takových výsledků, jako metody založené na konvolučních neuronových sítích [24][2][23], do výběru tedy nebyly zahrnuty.

Při volbě detekčních algoritmů bylo přihlédnuto k tomu, že ve frontě se z podstaty lidé pohybují pomalu (viz 8.3.5). Navrhovaný systém by měl nalézt uplatnění v institucích jako jsou letištní odbavení, letištní bezpečnostní kontrola nebo hypermarket. Dá se tedy předpokládat, že v mezním případě, kdy je fronta vyprázdněná a přijde do ní nová osoba, bude se ve sledované oblasti nacházet minimálně 10 sekund (čas průchodu scénou + čas obsluhy).

Vzhledem k tomu není potřeba detekce v reálném čase. Tato podmínka umožňuje volbu detektoru, který dosahuje lepší přesnost při klasifikaci na úkor času. Na základě práce „Speed/accuracy trade-offs for modern convolutional object detectors“ [11], která provnávala metody Faster R-CNN R-FCN a SSD byla pro testování vybrána metoda Faster R-CNN, která vykazuje nejlepší přesnost klasifikace za cenu pomalejší detekce.

Další zvolenou metodou byl detektor YOLOv3 [25], který oproti Faster R-CNN vykazuje větší rychlost detekce, ale o něco nižší přesnost.

Kritéria pro volbu detektoru byla zvolena následovně:

- Detektor dosahuje dobré úspěšnosti při detekci malých instancí objektů. Tento požadavek pramení z povahy videozáznamů z bezpečnostních kamer, které zabírají relativně velkou scénu, v které jedna osoba standardně zabírá oblast o velikosti desítek pixelů.
- Detektor dosahuje co nejvyšší hodnoty mAR. Pro správné stanovení čekací doby je podstatné určit kolik osob se ve frontě nachází. S rostoucím počtem neúspěšně detekovaných osob rychle roste chyba odhadu.
- Úspěšnost detekce při IoU 0,75 je dobrá. Pro zjišťování oblasti obsahující osobu není tak zásadní, aby byla oblast určena zcela přesně. Pro následné užití v trackeru je mAP při IoU 0,75 dostatečná a větší prioritu hraje mAR.
- Průměrný čas detekce na jednom snímku z testovacího datasetu by neměl zabrat déle než 10 sekund. Tento požadavek vychází z faktu, že pokud po detekci selže tracker ve sledování, může být sledovaná osoba během nastalých 10 sekund obsloužena a nezapočítána, což znamená nárůst chyby průměrné čekací doby jednoho člověka, která roste lineárně s počtem lidí ve frontě.

Vzhledem k dostupnosti kvalitních open-source implementací výše zmíněných detektorů a k nim odpovídajícím předtrénovaným vahám nebyl v rámci této práce vytvořen detektor. Pro porovnání byly využity existující implementace:

- Detectron.pytorch [26] – PyTorch implementace systému Detectron vyvíjeného firmou Facebook. V této implementaci byly otestovány detektory založené na Faster R-CNN a Mask RCNN. Při testování byly použity předtrénované váhy na datasetu COCO. Extraktor příznaků byl předtrénován na ImageNet, viz [29]
- Yolov3 [27] – PyTorch implementace YOLOv3. Při testování byly využity předtrénované váhy na databázi COCO. Extraktor příznaků byl předtrénován na ImageNet, viz [28]

Z dostupných modelů byl vybrán vzorek sedmi. Na těchto modelech byl spuštěn test na testovacím datasetu popsaném v 8.2.1. Velikost testovacích obrázků na vstupu byla u všech testovaných modelů byly nastaveny tak, aby kratší strana měla 800 pixelů. Vybrané výsledky z COCO metriky jsou k vidění v tabulce (viz 8.1)

Model	mAP[0,75]	mAP (malé instance)	mAR	čas
Faster RCNN X-101-32x8d-FPN	77,6	**41,9**	71,7	18,5
Faster RCNN X-101-64x4d-FPN	74,9	12,4	70,5	19,4
Mask RCNN R-50-C4	75,5	25,9	72,2	20,2
Mask RCNN X-101-32x8d-FPN	78,7	20,0	74,0	20,7
Mask RCNN X-101-64x4d-FPN	77,7	21,0	74,3	22,0
Mask RCNN X-152-32x8d-FPN-IN5k7	**86,3**	11,9	**81,6**	850,9
YOLOv3 Darknet-53	64,5	15,1	58,1	**8,0**

Tabulka 8.1: Výsledky testování detektorů

Z testování vyplynulo, že detektory mají všeobecně problém s detekováním malých instancí osob.

Dále test ukázal, že YOLOv3 je nejrychlejší, ale dosahuje špatné úspěšnosti na malých instancích. Tato vlastnost zřejmě souvisí se způsobem, jakým YOLO navrhuje kandidátní oblasti pro klasifikaci. Jeho autoři tento problém přímo adresují v [25] s poznámkou, že se týkal prvních dvou verzí a ve třetí verzi se jej podařilo odstranit. Tento test ukazuje opak. YOLO také dosáhlo nejnižší úspěšnosti v mAP[0,75] a mAR.

Největší úspěšnosti dosáhl model s hlubší architekturou ResNeXt-152-32x8d-FPN. Časová náročnost detekce s tímto modelem však výrazně přesahovala mez stanovenou pro použití.

Zbývajících 5 modelů dosáhlo na testovacím datasetu podobných výsledků mAP[0,75], mAR a času. Jediným výrazným rozdílem ve výsledcích byla úspěšnost klasifikace malých instancí objektů. Tato vlastnost je vzhledem k povaze navrhovaného systému podstatná (lze předpokládat, že kamera bude zabírat scénu z větší vzdálenosti), a proto byl pro další použití v prototypu zvolen model Faster RCNN X-101-32x8d-FPN.

8.2.3 Porovnání trackovacích algoritmů

Pro sledování rychlosti fronty je nezbytná znalost času, za jaký dojde k obsluze jedné čekající osoby. Objektový detektor ale nezachová identitu nalezených osob, což je nezbytná podmínka pro zjištění, zda osoba prošla oblastí obsluhy - byla obsloužena. Pro tento účel navrhovaný systém využívá trackovací algoritmus. Knihovna OpenCV obsahuje implementace osmi různých trackovacích algoritmů: MOSSE, KCF, CSRT, TLD, MIL, GOTURN, MedianFlow a Boosting. Ty byly v rámci této práce otestovány. Při použití Boosting trackeru program selže. Tato chyba se podle fór vyskytuje, ovšem v době psaní této práce se nepodařilo nalézt v oficiální dokumentaci informace o její opravě (OpenCV verze 3.4). Metoda testování ostatních algoritmů byla následující:

1. Na vstup bylo přivedeno video fronty z testovacích dat (8.2.1).

2. První frame videa byl detekován pomocí objektového detektoru a oblasti obsahující osoby byly předány trackeru.

3. Tracker následujících 100 framů sledoval předaný objekt. Pokud objekt prošel oblastí obsluhy (8.3.4, inkrementovalo se počítadlo osob a tracker byl odstraněn.

4. Po 100 framech byla provedena další detekce osob pomocí detektoru a výsledné oblasti byly porovnány s oblastmi, které sledoval tracker. Pokud oblast detekovaná a oblast trackovaná dosáhly skóre IoU >0,3, trackeru byla předána detekovaná oblast jako nová oblast sledování a byla zachována identita objektu. Pokud trackovaná oblast neměla odpovídající dvojici (žádná z detekovaných oblastí neodpovídala trackované oblasti se skóre IoU >0,3) byl detektor oblasti odebrán.

5. Kroky 3 a 4 se opakovaly do konce testovacího videa.

Výstupem z testování byly tři hodnoty. Počet osob, které prošly oblastí obsluhy, počet unikátních ID, které byly přiděleny trackovaným oblastem v průběhu celého videa a průměrný počet FPS. Ke každému videu použitému při testování byly hodnoty počtu osob, které prošly a počtu unikátních osob, které se ve videu objevily také spočítány ručně pro referenci. Výsledky tohoto testu zachycuje tabulka 8.2.

Test ukázal, že i při relativně přehledné scéně testované trackery často ztrácí sledovaný objekt, ale pokud je oblast obsluhy přehledná a sledované osoby do ní vstupují jednotlivě, lze na základě informací z trackeru počítat procházející osoby s velkou mírou konfidence.

Jednou z požadovaných vlastností trackeru je malá výpočetní náročnost. Mělo by být možné provádět trackování i v situacích, kdy se ve scéně nachází desítky osob. Některé z testovaných trackerů v nepřehledných scénách s více osobami, kdy docházelo k častému selhání sledování, vzniku nových sledovaných oblastí a zvýšení výpočetní náročnosti, začaly výrazně zpomalovat, což je činí nepoužitelnými.

Zajímavé bylo chování trackeru GOTURN, který je ze všech testovaných trackerů jako jediný založený na konvolučních neuronových sítích. V rámci implementace v OpenCV je ke stažení předtrénovaný model. Tento tracker podle tvůrců [30] dosahuje na VOT 2014 datasetu špičkových výsledků, ale při použití na testovacích videích této práce okamžitě po předání ztrácel sledovaný objekt a vytvářel velké množství falešně pozitivních oblastí sledování.

Ze všech otestovaných trackerů dosahoval dobré přesnosti za podmínky zachování velké rychlosti zpracování jednotlivých snímků tracker KCF, a proto byl zvolen jako optimální pro další použití v navrhovaném prototypu systému.

Tracker	Video	ID	Počet	FPS
Očekávaný	1	11	6	
	2	13	9	
	3	11	6	
CSRT	1	21	6	6,21
	2	34	9	3,69
	3	23	5	1,79
GOTURN	1	94	16	15,12
	2	101	15	27,64
	3	50	6	33,5
KCF	1	16	6	10,87
	2	32	9	6,43
	3	20	4	2,5
MedianFlow	1	30	6	37,38
	2	53	12	24,64
	3	35	4	8,97
MIL	1	18	7	4,44
	2	34	11	2,74
	3	22	5	2,05
MOSSE	1	36	6	54,26
	2	66	8	40,17
	3	25	6	17,42
TLD	1	31	6	6,33
	2	51	9	4,13
	3	32	1	1.53

Tabulka 8.2: Výsledky testování trackerů

8.2.4 Vliv umístění kamery

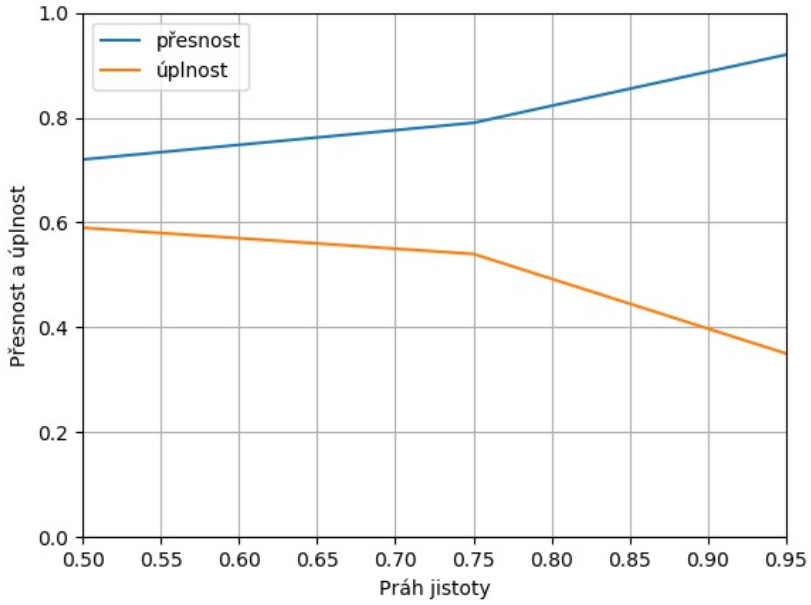
Jak již bylo zmíněno, velká část problémů navrženého systému vychází z faktu, že dostupné detekční a především trackovací algoritmy špatně zvládají okluzi sledovaných objektů. Tato nevýhoda je v případě fronty lidí inherentní. Její odstranění by bylo teoreticky možné do značné míry, pokud by kamera zabírající frontu byla umístěna kolmo k rovině tvořené podlahou. Toto řešení naráží v realitě na dva problémy.

První problém je ekonomický. Nasazení navrhovaného systému počítá s využitím již instalovaných kamer, které jsou bezpečnostní a přirozeně tedy umístěné tak, aby bylo možné identifikovat osoby na záběrech, tzn. přibližně v úhlu 45°. Instalace jednoúčelových kamer představuje velkou investici.

Druhý problém představuje robustnost detekčního systému. K jejímu otestování byly z testovacího datasetu vybrány snímky stejné scény ve stejném čase zachycené ze dvou úhlů: kolmo a rovnoběžně k podlaze. Obě scény zachycovaly frontu bez okluze. Část snímků byla pro testovací účely uměle zašuměna a rozšířena zrcadlovým převrácením. Ve snímcích byly detekovány osoby na třech prazích jistoty (0,5; 0,75; 0.95). U rovnoběžné scény dosáhl detektor 100% přesnosti i úplnosti. Výsledky testu

na kolmé scéně jsou vidět v grafu 8.3. Detektor nebyl úspěšný ani u osob, které byly dobře separované od pozadí. Z toho lze vyvodit, že trénovací data neobsahují instance osob zachycených shora v dostatečném množství. I přes to, že třída osoby je v COCO datasetu zastoupena téměř milionem instancí a je tedy nejpočetnější ze všech 81 tříd [14].

Tento problém by bylo možné odstranit dotrénováním detekčního modelu datasetem obsahujícím snímky pořízené z vhodného úhlu v dostatečném množství. V době vytváření práce se nepodařilo odpovídající dataset nalézt.



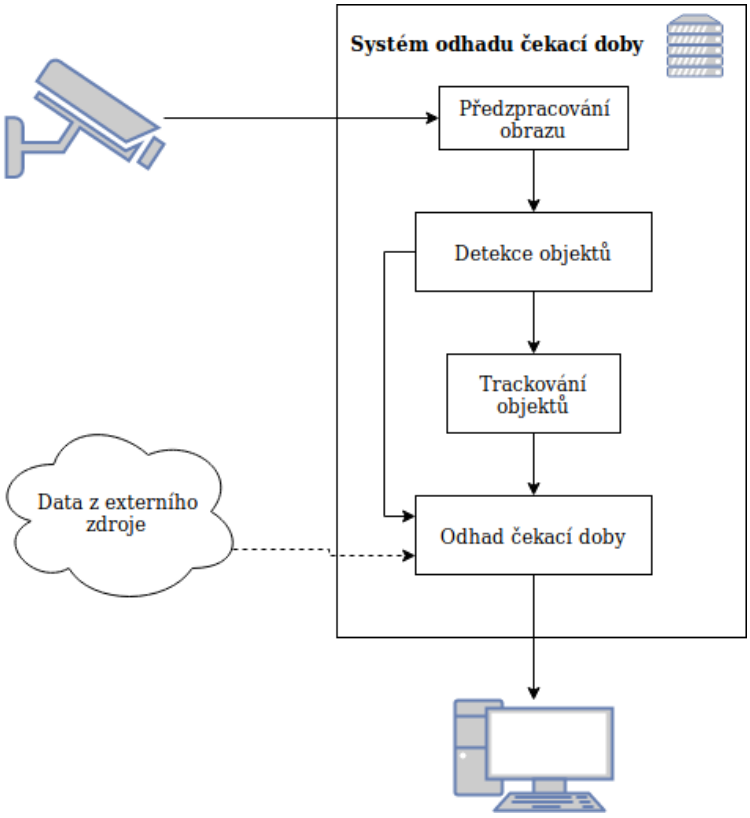
Obrázek 8.3: Úspěšnost detekce na kameře umístěné shora

8.3 Systém

Následující sekce se zabývá implementací systému pro odhad čekací doby. Způsob fungování systému popisuje schéma 8.4. Systém je navržen modulárně tak, aby jeho klíčové části – detekce a trackování – byly s minimem komplikací vyměnitelné. Tato vlastnost reflektuje fakt, že algoritmy detekce i trackování velice rychle zastarávají.

8.3.1 Izolace fronty ve scéně

Jak je vidět na obrázku originální scény 8.5, v reálném prostředí je třeba počítat s tím, že ve většině případů kamera nezabírá pouze oblast fronty. V důsledku nelze provádět detekci na celém záběru, neboť by zjištěný počet osob zahrnoval i osoby procházející a rostla by chyba odhadu. Tuto situaci lze vyřešit vymaskováním oblasti, která není validní, pomocí binární masky, tak jak je to znázorněno na obrázku 8.5.

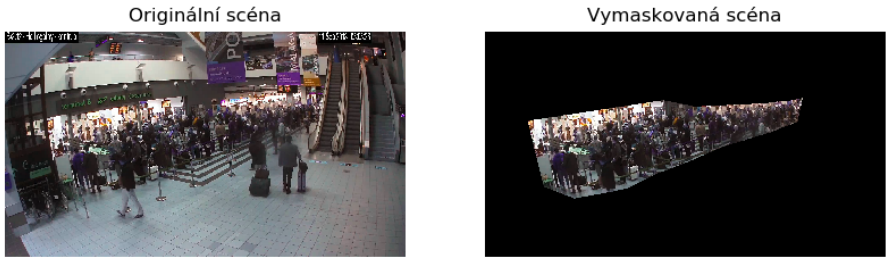


Obrázek 8.4: Schéma systému pro odhad čekací doby

Tento přístup s sebou nese další výhodu, a tou je aplikace detekce a trackování pouze na validní část záběru. Díky tomuto přístupu lze snížit výpočetní náročnost práce se záznamem, neboť se veškeré výpočty dějí s menším objemem dat.

Navrhovaný systém umožňuje řešit vytvoření binární masky dvěma způsoby. Prvním způsobem je načtení souboru *.json, který obsahuje slovník s polem bodů definujících v obraze polygon fronty. Z tohoto souboru bodů je pomocí knihovny PIL vytvořeno pole typu numpy, které obsahuje nulové hodnoty na všech souřadnicích, které se nachází vně validní oblasti. Vstupní obraz je pak za běhu programu vymaskován pomocí logického indexování, což je výpočetně nenáročné.

Druhý způsob se liší způsobem, jakým je načteno pole bodů polygonu. Jeho vznik byl motivován skutečností, že oblast, ve které se fronta nachází, se může s časem měnit (obsluha provozu využívá přenosných vymezovacích sloupků podle aktuální situace a vytížení), a vytváření *.json souboru je v tu chvíli časově nevýhodné. Možným řešením této situace je využití programu třetí strany, jako je například Labelme (viz 8.2.1) a jeho úprava. Tato možnost byla zamítnuta z důvodu zbytečného bobtnání kódu programu a závislosti na programu třetí strany. Vhodným řešením s optimální mírou abstrakce je použití OpenCV, které umožňuje vytvoření callback funkce, která zachytává souřadnice kliknutí na zobrazený snímek. Byla vytvořena funkcionalita, která po načtení videa zobrazí uživateli první snímek a umožní mu ohraničit oblast fronty. Po dokončení předá načtené souřadnice funkci, která vytváří binární masku.



Obrázek 8.5: Vymaskování fronty pomocí binární masky

8.3.2 Detekce osob ve frontě

Při inicializaci systému je detekční systém inicializován souborem yaml s parametry detekčního algoritmu, souborem s váhami ve formátu pickle a prahu jistoty, který je vyjádřen jako desetinné číslo v intervalu [0,1] a určuje, zda bude nalezená instance objektu vrácena, nebo ne.

Detekční systém komunikuje s hlavním programem funkcí *detect*, která jako parametr přijme výřez z předzpracovaného obrazu obsahující oblast fronty. Na tomto obraze provádí detekci osob a jako návratovou hodnotu předává dvojici objektů – list ohraničujících obdélníků pro nalezené instance osob a vstupní výřez s anotacemi.

Váhy použitého modelu zvoleného v sekci 8.2.2 jsou předtrénovány na datasetu COCO, který klasifikuje na výstupní vrstvě 81 tříd objektů. Toto je neoptimální z několika důvodů. Detekované objekty by bylo v hlavním programu nutné dále třídit s tím, že objekty klasifikované jinak, než jako osoby, by byly zahozeny. Detektor také určuje predikovanou třídu jako třídu s největším skóre na vstupu do funkce softmax. V situaci, kdy by detekovaný objekt představoval osobu, ale skóre predikce osoby by nebylo maximální, osoba by nebyla detekována.

Tento problém lze vyřešit přenastavením předtrénovaných vah a biasů na vstupu do funkce softmax. Váhy a biasy pro třídy 0 a 1, které v COCO představují pozadí a osobu byly ponechány, zatímco zbylé váhy byly nastaveny na zápornou hodnotu, což je při detekci diskriminuje a dochází pouze k detekci tříd pozadí a osoba.

Dále byl model nastaven, aby vracel 200 predikcí s největší jistotou namísto původních 100. Tato změna vychází z faktu, že v testovací scéně z letiště ve Vratislavi je 121 instancí osob.

Minimální míra jistoty detekce byla nastavena na 0,2 z původních 0,05 aby nevznikaly málo pravděpodobné predikce, které se musí dále zpracovávat. Dále bylo nastaveno použití Soft NMS namísto normální NMS, což slibuje lepší schopnost zachovat predikce objektů, které se vzájemně významně překrývají. Toto nastavení podle [31] může znamenat na třídě osoba více jak 2% zlepšení (testováno na COCO).

Nakonec byl navýšen počet návrhů na výstupu RPN na 2000.

8.3.3 Trackování osob ve frontě

Trackovací API knihovny OpenCV neobsahuje prostředky pro trackování více objektů. Za účelem vyřešení tohoto problému byla vytvořena třída trackování. Tato třída udržuje list aktivních trackerů. S hlavním programem komunikuje funkcemi *update*, *add* a *remove*. Třída dále uchovává informace o identitě jednotlivých objektů. Při spuštění systému je vytvořena instance třídy a seznam trackerů je inicializován vstupem z detekce provedené na prvním framu vstupního videa.

8.3.4 Oblast obsluhy

Odhad čekací doby lze provést několika způsoby. Jednou z nabízených možností je měření rychlosti, jakou se pohybují čekající osoby. Tento přístup naráží na limity zpracování obrazu. S výjimkou scény, kdy tvoří fronta rovnou linii a kamera ji zabírá kolmo a scény, kdy je kamera umístěna shora, je stanovení rychlosti měřením posunu po obraze těžko proveditelné, neboť vzdálenosti jsou deformovány a ztrácí se linearita vzdálenosti v obraze a reálné vzdálenosti. Taková situace by vyžadovala měření reálné vzdálenosti pokaždé, kdy by došlo ke změně linie fronty (pomocí vytyčovacích pásek). Takovýto přístup by také kladl mnohem větší nároky na trackery. Dalším problémem tohoto řešení je fakt, že předpokládá rovnoměrnou hustotu osob ve frontě. Ve chvíli, kdy by se ve frontě nacházela hustší a řidší místa by odhad touto metodou nebyl správný. Bylo by dále nutné, v případě fronty jiného půdorysu než přímky, tvar této fronty stanovit jako křivku a pohyb osob v ní počítat v nejbližším bodě křivky. Kvůli těmto překážkám je takový postup prakticky neaplikovatelný.

Přístup, který používá navržený systém vychází ze základního znaku fronty. Přístup předpokládá, že každá fronta má jednu nebo více oblastí obsluhy, které jsou v čase ve scéně nepohyblivé a dobře viditelné. Tyto oblasti obsluhy, tvoří úzké hrdlo celé fronty. Osoby čekající ve frontě do oblastí obsluhy vstupují jednotlivě, nedochází v ní tedy k okluzi.

Oblast obsluhy je v navrhovaném systému reprezentována obdélníkovou oblastí v obraze. Tato oblast je buď načtena při spuštění systému z předaného souboru *.json, který obsahuje i informace o binární masce (viz 8.3.1), a nebo zadána následně po zadání binární masky v GUI.

8.3.5 Odhad čekací doby

Znalost vlastností fronty je zásadní pro správné fungování systému. V první řadě znamená, že zákazník nebo management bude mít správnou informaci o době čekání ve frontě. Dále také znamená, že lze optimálně stanovit, s jakou frekvencí je nutné provádět detekci, aby nedošlo k průchodu zákazníka uzlem obsluhy aniž by byl zaznamenán.

Tyto vlastnosti byly zjištěny experimentálně. Bylo provedeno 5 měření dob obsluhy v obchodech. Časy měření byly zvoleny tak, aby zahrnovaly doby, kdy lze předpokládat menší provoz (všední dny dopoledne), a doby, kdy lze předpokládat větší provoz (víkend, všední dny odpoledne). Byl měřen čas od načtení první položky

nákupu do předání dokladu o zaplacení zákazníkovi. Předpokládá se, že během měření je tok zákazníků stacionární. Podrobnosti o měřeních zachycuje tabulka 8.3.

	Počet vzorků	Čas měření (od)	Místo měření
Měření č. 1	50	28. 3. 2019 10.00	Albert Atrium Flóra
Měření č. 2	50	31. 3. 2019 10.00	Kaufland Ml. Boleslav
Měření č. 3	50	1. 4. 2019 10.00	Albert Ml. Boleslav
Měření č. 4	50	4. 4. 2019 9.00	Kaufland Ml. Boleslav
Měření č. 5	50	8. 4. 2019 16.00	Albert Ml. Boleslav

Tabulka 8.3: Měření dob obsluhy

Na naměřených datech byly provedeny testy dobré shody za účelem zjištění příslušnosti k rozdělení pravděpodobnosti. Na hladině spolehlivosti 0.05 byla otestována hypotéza o původu dat z Weibullova rozdělení. Pro všech pět měření nebyla hypotéza zamítnuta. Dále tedy lze předpokládat, že data pochází z tohoto rozdělení. Z distribuční funkce teoretických rozdělení pravděpodobnosti bylo zjištěno následující: Průměrná pravděpodobnost, že doba obsluhy bude kratší než 5 sekund je 2,29%, pro dobu kratší než 1 sekunda je to 0.18%. Tímto postupem lze stanovit vhodnou vzorkovací frekvenci pro detekci. Dále lze využít znalost parametrů teoretického rozdělení k výpočtu střední doby obsluhy

$$E = k * [\Gamma(1 + b^{-1})] \tag{8.1}$$

a směrodatné odchylky

$$\sigma = k^2[\Gamma(1 + 2b^{-1} - \Gamma(1 + b^{-1}] \tag{8.2}$$

kde k a b jsou parametry Weibullova rozdělení.

Z těchto informací je možno vypočítat čekací dobu nově příchozího zákazníka jako $n \times E$, kde n je počet detekovaných čekajících.

Je nutno podotknout, že naměřený soubor má omezený rozsah, a proto závěry nemají obecnou platnost. Předpoklad původu dat z Weibullova rozdělení může být pro jiné typy provozů chybný. U některých provozů (Starbucks, Costa Coffee atp.) se z podstaty jejich fungování dá předpokládat téměř rovnoměrné rozdělení časů obsluhy. Naproti tomu v hypermarketech, jak se ukázalo i při sběru dat v 8.3, můžou být rozdíly mezi minimem a maximem velké.

Přístup k tomuto problému v této práci je výpočet času obsluhy na základě posledních k časů průchodů, což může při malém k lépe popisovat časový úsek (například je pravděpodobnější předpoklad, že ráno v menším obchodě budou mít časy odbavení menší rozptyl, než v průběhu celého dne), zároveň však může dojít k většímu zkreslení v důsledku výskytu extrému.

Zároveň bylo zjištěno, že na naměřených datech se střední hodnota vypočtená podle 8.1 a aritmetický průměr neliší více, než o zlomek sekundy.

8.3.6 Běh systému pro odhad čekací doby

Spuštění systému je prováděno z příkazové řádky. Uživatel pomocí přepínačů může nastavit práh jistoty, pro který bude přijata detekce, cestu k binární masce fronty, cestu k videosouboru se záznamem fronty, práh IoU mezi trackovaným objektem a detekovaným objektem, pro který bude zachována identita objektu a vzorkovací frekvenci, s kterou bude prováděna detekce osob.

Poté co je program spuštěn je načtena binární maska a oblast obsluhy (buď ze souboru nebo z GUI). Program zároveň stanoví validní oblast v obraze, která bude zpracovávána následovně.

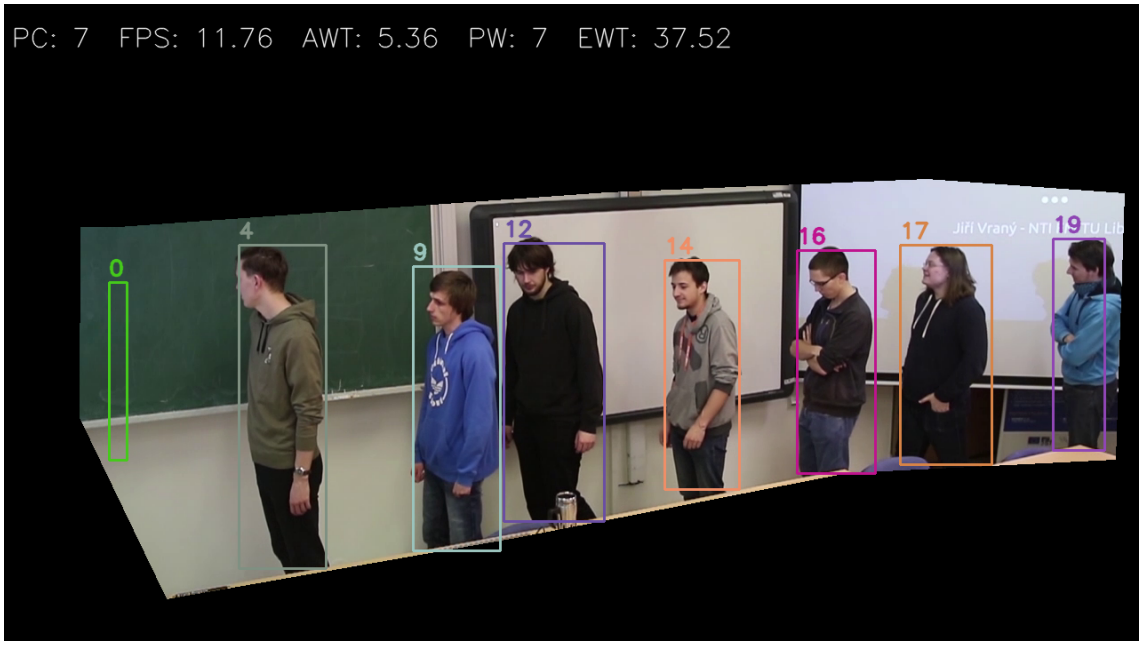
Program provede detekci osob na prvním framu videa a vrácené ohraničující obdélníky předá třídě trackeru, který inicializuje pro každý obdélník ID a tracker. Ačkoliv je trackování méně výpočetně náročné, není tak spolehlivé a není možné předpokládat, že dojde k úspěšnému trackování osob skrz celou scénu. Zároveň by nedocházelo k inicializaci nových trackerů pro osoby, které nově vstoupí do scény. Tento problém je řešen následovně: na každém n–tém snímku je znovu provedena detekce. Ohraničující obdélníky z výstupu detektoru a trackeru jsou vzájemně porovnány. Pokud je mezi dvěma obdélníky hodnota $IoU > 0,3$, je trackeru předán obdélník z detektoru a je zachována identita trackované osoby. Pokud obdélník z trackeru neodpovídá žádnému obdélníku z detektoru, předpokládá se, že tracker selhal a je odstraněn. V případě, že detekovaný obdélník neodpovídá žádnému obdélníku z aktivních trackerů, předpokládá se, že detekovaná osoba je buď nově příchozí, nebo došlo k selhání jejího trackeru. V takovém případě je inicializován nový tracker s novými identitami.

Konstanta n je parametrem celého systému a představuje detekční frekvenci. Vzhledem k tomu, že detekce je oproti trackování circa $10\times$ pomalejší, ale pomáhá zpřesňovat trackované oblasti, je možné její změnou prioritizovat buď rychlost, nebo přesnost celého systému.

Systém za běhu kontroluje, zda se některý z obdélníků trackeru nepřekrývá s obdélníkem oblasti obsluhy. Pokud taková situace nastane, znamená to, že trackovaná osoba byla obsloužena. Její tracker je odstraněn, počítadlo průchodů osob je inkrementováno a je uložen údaj o počtu framů mezi aktuálním a předchozím průchodem. Tento údaj reprezentuje čekací dobu osoby ve frontě. V případě, že by ve stejnou chvíli došlo k detekci, což by způsobilo vznik nového trackeru právě započítané osoby, je v okamžik průchodu oblast obsluhy deaktivována po dobu dvou sekund, což je dostatečný čas, aby procházející osoba opustila sledovanou scénu.

Na základě dat o počtu aktivních trackerů a časech obsluhy je vypočtena čekací doba pro nově příchozího zákazníka způsobem popsaným v 8.3.5.

Na obrázku 8.6 je náhled vytvořeného systému. Zelená oblast s ID 0 představuje oblast obsluhy. Dále jsou k dispozici informace o počtu osob, které prošly PC, rychlosti zpracování FPS, průměrné čekací době jednoho člověka AWT, počtu aktivních trackerů (počtu čekajících osob) PW a odhadované čekací době nově příchozí osoby EWT.



Obrázek 8.6: Screenshot běžícího systému pro odhad čekací doby

8.4 Budoucí práce

8.4.1 Dotrénování modelu z reálných záznamů

Prototyp představený v této práci využívá detekční model vytrénovaný na COCO datasetu. Pokud by v rámci nasazení systému v reálném provozu byl nasbírán videozáznam v dostatečné délce, bylo by možné na jeho základě vytvořit dataset pro dotrénování (transfer learning) detekčního modelu. To by mohlo vést ke zmírnění problému detekce na záznamu z kamery umístěné shora.

8.4.2 Vstup dat z externího zdroje

Významným zdrojem potenciálních chyb v odhadu čekací doby je oblast obsluhy reprezentovaná oblastí v obraze.

Možným budoucím řešením je vstup dat o časech obsluhy z externího systému (viz 8.4). Může se jednat o informace z pokladního systému (čas mezi následujícími dokončení nákupu), z bezpečnostních rámů, z turniketů nebo z čidel. Tento přístup potenciálně nabízí dvě výhody. Jednak méně chybové informace o časech obsluhy a také úsporu výpočetního výkonu. Systém s vnějším vstupem dat nevyžaduje běh trackerů a zároveň nehrozí, že zákazník projde dříve, než je detekován – je nezapočten. Pokud by systém disponoval takovým vstupem dat, mohl by být například nasazen model detekce s vyšší přesností za cenu vyšší časové náročnosti detekce.

8.4.3 Tracker využívající GPU

Implementace KCF trackeru z OpenCV, která byla využita pro navržený prototyp, využívá pro svůj chod CPU. Možným zlepšením je tedy nasazení implementace využívající GPU, což by mohlo vést ke zrychlení běhu systému.

8.4.4 Úprava kotev pro volbu kandidátních oblastí

Detekce v navrhovaném prototypu probíhá pro všechny poměry stran a měřítka. Za předpokladu, že jsou známy vlastnosti scény, kterou zabírá kamera s běžícím prototypem by bylo možné přizpůsobit kotvy pro volbu kandidátních oblastí tak, aby nedocházelo k návrhům oblastí, které jsou reálně příliš velké nebo mají špatný poměr stran, než aby mohly obsahovat osobu.

9 Závěr

Cílem této práce bylo seznámit se se současným stavem vývoje detekčních algoritmů, vybrané algoritmy porovnat a na základě porovnání vybrat nejvhodnější algoritmus pro praktickou implementaci prototypu systému. V dalším kroku byl s využitím vybraného algoritmu implementován prototyp systému pro odhad čekací doby ve frontě na základě videozáznamu.

V úvodní části práce byl ve stručnosti představen obor teorie hromadné obsluhy, jehož formální ukotvení pomohlo k lepšímu pochopení problému optimalizace čekací doby (2). Dále byly prozkoumány algoritmy detekce osob s důrazem na jejich vhodnost pro řešení úlohy z reálného prostředí. Byly popsány neuronové sítě a konvoluční neuronové sítě jako state-of-the-art technologie pro rozpoznávání objektů v obraze (3, 4). Krátce byly také uvedeny detekční algoritmy, které neuronové sítě využívají a jsou aktuálně hojně využívané. Pro účely dalšího využití v části testování byly představeny metody hodnocení úspěšnosti detekčních algoritmů (4). V 5 byly popsány nejpoužívanější datasety v oblasti počítačového vidění, klasifikace a detekce objektů. Kapitola 6 představuje knihovny pro strojové učení s ohledem na jejich specifika, aktuálnost a použitelnost (míra abstrakce, dokumentace). Dále byla v 7 popsána existující komerční i open-source řešení, která se zabývají stejnou nebo podobnou problematikou jako tato práce. Jsou zhodnoceny z praktického hlediska.

Kapitola 8.3 je věnována implementovanému prototypu. V sekci 8.2 je popsáno testování prostředků pro použití v systému. Pro provedení testů byly v rámci této práce vytvořeny videozáznamy front z různých úhlů. Ze snímků z těchto záznamů a z volně dostupných snímků byl vytvořen testovací dataset ve formátu COCO pro otestování implementací detekčních algoritmů. Pro testování byly na základě rešerše vybrány implementace algoritmů Faster R-CNN, Mask R-CNN a YOLOv3. Na základě testových kritérií, které se zaměrovaly na úspěšnost detekce na malých objektech, všeobecnou úspěšnost detekce a čas detekce byl pro použití v prototypu vybrán algoritmus Faster R-CNN s FPN a extraktorem příznaků ResNeXt-101-32x8d.

Pořízené videozáznamy byly dále použity k otestování trackerů. Byly testovány trackery z Tracking API OpenCV. Na základě testů byl pro použití zvolen KCF tracker.

Dále bylo na snímcích z testovacích dat provedeno zkoumání závislosti úhlu kamery k rovině země a úspěšnosti detekce. Zkoumání ukázalo, že úhly blízké 90° znamenají velký propad úspěšnosti detekce.

Výstupem práce je prototyp systému pro odhad čekací doby. Jeho fungování je popsáno v 8.3. Systém umožňuje izolaci fronty ve scéně, detekci osob ve frontě

a jejich následné trackování. Systém také pomocí *oblasti obsluhy* v obraze počítá průchozí osoby. Na základě těchto informací systém vypočítává odhad čekací doby pro nově příchozí osoby.

Přístup k problému odhadu čekací doby popisuje sekce 8.3.5. Časy odbavení byly experimentálně zkoumány. Na jejich základě byl navržen přístup k odhadu čekací doby v této práci. Jsou zde informace o způsobu výpočtu odhadu čekací doby a jeho limity.

V sekci 8.4 byly popsány možná vylepšení navrženého detekčního systému, která nebylo z různých důvodů možné realizovat.

Ukázalo se, že problém odhadu čekací doby může být účinně řešen pomocí rozpoznávání obrazu. Byly popsány limity tohoto přístupu a způsoby, jak je některé z nich možné překonat.

Literatura

[1] STURDEVANT, Mark. *Recognize and Count Objects in a Video*. GitHub [online]. 2019 [cit. 2019-04-27]. Dostupné z: https://github.com/IBM/powerai-counting-cars/blob/master/notebooks/counting_cars.ipynb

[2] LECUN, Y., B. BOSER, J. S. DENKER, D. HENDERSON, R. E. HOWARD, W. HUBBARD a L. D. JACKEL. *Backpropagation Applied to Handwritten Zip Code Recognition*. Neural Computation. 1989, 1(4), 541-551. DOI: 10.1162/neco.1989.1.4.541. ISSN 0899-7667. Dostupné také z: http://www.mitpressjournals.org/doi/10.1162/neco.1989.1.4.541

[3] KRIZHEVSKY, Alex, Ilya SUTSKEVER a Geoffrey E. HINTON. *ImageNet classification with deep convolutional neural networks*. Communications of the ACM. 2017, 60(6), 84-90. DOI: 10.1145/3065386. ISSN 00010782. Dostupné také z: http://dl.acm.org/citation.cfm?doid=3098997.3065386

[4] SZTRIK, János. *Basic queueing theory*. University of Debrecen, Faculty of Informatics, 2012, 193.

[5] ZITNICK, C. Lawrence a Piotr DOLLÁR. *Edge Boxes: Locating Object Proposals from Edges*. Computer Vision – ECCV 2014. Cham: Springer International Publishing, 2014, 2014, , 391-405. Lecture Notes in Computer Science. DOI: 10.1007/978-3-319-10602-1_26. ISBN 978-3-319-10601-4. Dostupné také z: http://link.springer.com/10.1007/978-3-319-10602-1_26

[6] HAILONG LI, ZHENDONG WU a JIANWU ZHANG. *Pedestrian detection based on deep learning model*. 2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI). IEEE, 2016, 2016, , 796-800. DOI: 10.1109/CISP-BMEI.2016.7852818. ISBN 978-1-5090-3710-0. Dostupné také z: http://ieeexplore.ieee.org/document/7852818/

[7] UIJLINGS, J. R. R., K. E. A. VAN DE SANDE, T. GEVERS a A. W. M. SMEULDERS. *Selective Search for Object Recognition*. International Journal of Computer Vision. 2013, 104(2), 154-171. DOI: 10.1007/s11263-013-0620-5. ISSN 0920-5691. Dostupné také z: http://link.springer.com/10.1007/s11263-013-0620-5

[8] FELZENSZWALB, Pedro F. a Daniel P. HUTTENLOCHER. *Efficient Graph-Based Image Segmentation*. International Journal of Computer Vision. 2004, 59(2), 167-181. DOI: 10.1023/B:VISI.0000022288.19776.77. ISSN 0920-5691. Dostupné také z: http://link.springer.com/10.1023/B:VISI.0000022288.19776.77

[9] VIOLA, P. a M. JONES. *Rapid object detection using a boosted cascade of simple features*. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001. IEEE Comput. Soc, 2001, 59(2), I-511-I-518. DOI: 10.1109/CVPR.2001.990517. ISBN 0-7695-1272-0. ISSN 0920-5691. Dostupné také z: http://ieeexplore.ieee.org/document/990517/

[10] DALAL, N. a B. TRIGGS. *Histograms of Oriented Gradients for Human Detection*. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). IEEE, 2005, , 886-893. DOI: 10.1109/CVPR.2005.177. ISBN 0-7695-2372-2. Dostupné také z: http://ieeexplore.ieee.org/document/1467360/

[11] HUANG, Jonathan, Vivek RATHOD, Chen SUN, et al. *Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors*. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2017, 2017, , 3296-3297. DOI: 10.1109/CVPR.2017.351. ISBN 978-1-5386-0457-1. Dostupné také z: http://ieeexplore.ieee.org/document/8099834/

[12] DAI, Jifeng, et al. R-fcn: *Object detection via region-based fully convolutional networks*. Advances in neural information processing systems. 2016. p. 379-387. Dostupné také z: https://papers.nips.cc/paper/6465-r-fcn-object-detection-via-region-based-fully-convolutional-networks.pdf

[13] EVERINGHAM, Mark, Luc VAN GOOL, Christopher K. I. WILLIAMS, John WINN a Andrew ZISSERMAN. *The Pascal Visual Object Classes (VOC) Challenge*. International Journal of Computer Vision. 2010, 88(2), 303-338. DOI: 10.1007/s11263-009-0275-4. ISSN 0920-5691. Dostupné také z: http://link.springer.com/10.1007/s11263-009-0275-4

[14] LIN, Tsung-Yi, Michael MAIRE, Serge BELONGIE, James HAYS, Pietro PERONA, Deva RAMANAN, Piotr DOLLÁR a C. Lawrence ZITNICK. *Microsoft COCO: Common Objects in Context*. Computer Vision – ECCV 2014. Cham: Springer International Publishing, 2014, 2014, 88(2), 740-755. Lecture Notes in Computer Science. DOI: 10.1007/978-3-319-10602-1_48. ISBN 978-3-319-10601-4. ISSN 0920-5691. Dostupné také z: http://link.springer.com/10.1007/978-3-319-10602-1_48

[15] COCO Detection Evaluation. COCO [online]. [cit. 2019-04-27]. Dostupné z: http://cocodataset.org/#detection-eval

[16] HALE, Jeff. *Deep Learning Framework Power Scores 2018*. Kaggle [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://www.kaggle.com/discdiver/deep-learning-framework-power-scores-2018

[17] VORÁČOVÁ, Šárka. *Teorie hromadné obsluhy*. Fakulta dopravní ČVUT [online]. [cit. 2019-04-27]. Dostupné z: https://www.fd.cvut.cz/department/k611/pedagog/K611THO_soubory/webskriptum/

[18] NEUROMATION. *Tracking with darkflow*. GitHub [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://github.com/neuromation/Tracking-with-darkflow

[19] KERBEROS.IO. *Kerberos.io* [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://github.com/kerberos-io

[20] VASA, Hardik. *Google Images Download*. GitHub [online]. 2015 [cit. 2019-04-27]. Dostupné z: https://github.com/hardikvasa/google-images-download

[21] KENTARO, Wada. *labelme*. GitHub [online]. 2011 [cit. 2019-04-27]. Dostupné z: https://github.com/wkentaro/labelme

[22] COCO Data Format. COCO [online]. [cit. 2019-04-27]. Dostupné z: http://cocodataset.org/#format-data

[23] GIRSHICK, Ross, Jeff DONAHUE, Trevor DARRELL a Jitendra MALIK. *Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation*. 2014 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2014, 2014, , 580-587. DOI: 10.1109/CVPR.2014.81. ISBN 978-1-4799-5118-5. Dostupné také z: http://ieeexplore.ieee.org/document/6909475/

[24] LECUN, Y., L. BOTTOU, Y. BENGIO a P. HAFFNER. *Gradient-based learning applied to document recognition*. Proceedings of the IEEE. 86(11), 2278-2324. DOI: 10.1109/5.726791. ISSN 00189219. Dostupné také z: http://ieeexplore.ieee.org/document/726791/

[25] REDMON, Joseph a Ali FARHADI. Yolov3: *An icremental improvement*. ArXiv. 2018. Dostupné také z: https://arxiv.org/pdf/1804.02767.pdf

[26] ROYTSENG-TW. *A Pytorch Implementation of Detectron*. GitHub [online]. 2017 [cit. 2019-04-27]. Dostupné z: https://github.com/roytseng-tw/Detectron.pytorch

[27] ULTRALYTICS LLC. *Yolov3*. GitHub [online]. 2018 [cit. 2019-04-27]. Dostupné z: https://github.com/ultralytics/yolov3

[28] REDMON, Joseph. *YOLO: Real-Time Object Detection*. Joseph Chet Redmon [online]. 2018 [cit. 2019-04-28]. Dostupné z: https://pjreddie.com/darknet/yolo/

[29] FACEBOOK RESEARCH. *Detectron Model Zoo and Baselines*. GitHub [online]. 2018 [cit. 2019-04-28]. Dostupné z: https://github.com/facebookresearch/Detectron/blob/master/MODEL_ZOO.md

[30] HELD, David, Sebastian THRUN a Silvio SAVARESE. *Learning to Track at 100 FPS with Deep Regression Networks*. Computer Vision – ECCV 2016. Cham: Springer International Publishing, 2016, 2016-09-17, , 749-765. Lecture Notes in Computer Science. DOI: 10.1007/978-3-319-46448-0_45. ISBN 978-3-319-46447-3. Dostupné také z: http://link.springer.com/10.1007/978-3-319-46448-0_45

[31] BODLA, Navaneeth, Bharat SINGH, Rama CHELLAPPA a Larry S. DAVIS. *Soft-NMS — Improving Object Detection with One Line of Code*. 2017 IEEE International Conference on Computer Vision (ICCV). IEEE, 2017, 2017, , 5562-5570. DOI: 10.1109/ICCV.2017.593. ISBN 978-1-5386-1032-9. Dostupné také z: http://ieeexplore.ieee.org/document/8237855/