Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

BAKALÁŘSKÁ PRÁCE

Detekce a rozpoznávání SPZ

PLZEŇ, 2021 Michael Bosák

Prohlášení

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

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

V Plzni dne 24. května 2021

.....

Michael Bosák

Anotace

Tato práce se zabývá automatickou detekcí a rozpoznáváním registračních značek vozidel (SPZ). V teoretické části jsou představeny základní principy a algoritmy detekce objektů a optického rozpoznávání znaků. Praktická část je věnována přípravě vlastních datasetů. Na nich jsou poté trénovány a testovány algoritmy pro detekci objektů v obraze založené na neuronových sítích a testovány algoritmy pro optické rozpoznávání znaků. Ty jsou následně kombinovány pro účely detekce a rozpoznávání SPZ. Na základě výsledků testování je nalezena nejlepší kombinace těchto algoritmů. Na závěr je tato kombinace otestována na několika obrazech SPZ a je uveden praktický příklad využití výsledků této bakalářské práce.

Klíčová slova

Rozpoznávání SPZ, Detekce objektů, Konvoluční neuronové sítě, Počítačové vidění, OCR, YOLO, Faster R-CNN, OpenALPR, Tesseract OCR

Annotation

This thesis is focused on automatic license plate detection and recognition. In the theoretical part the basic principles and algorithms of object detection and optical character recognition are introduced. The practical part deals with its own datasets preparation. These datasets are then used for object detection algorithms based on neural networks training and testing and also for optical character recognition algorithms testing. In the next step these algorithms are combined for the purposes of license plate detection and recognition. Based on results the best combination of these algorithms is found. In the end this combination is tested on several license plate images and an example of practical usage of this thesis results is given.

Keywords

License plate recognition, Object detection, Convolutional neural network, Computer vision, OCR, YOLO, Faster R-CNN, OpenALPR, Tesseract OCR

Poděkování

Chtěl bych poděkovat Ing. Luboši Šmídlovi, Ph.D., vedoucímu mé bakalářské práce, za odborné vedení, cenné rady, věcné připomínky a čas, který mi věnoval při konzultacích.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Systémy pro detekci a rozpoznání SPZ mají v motoristickém prostředí celou řadu uplatnění. Můžeme je najít u silnic, kde snímají SPZ vozidel, které například porušili rychlost. Své uplatnění mají také na parkovištích s omezeným přístupem, kde takový systém na základě rozpoznané SPZ určuje, která vozidla mají povolený vjezd na dané parkoviště, a která nikoliv. Takové systémy lze najít nejen na veřejných parkovištích, ale i na branách pro vjezd na soukromé pozemky, či do osobních garáží. Na placených parkovištích může být na základě rozpoznané SPZ při vjezdu a odjezdu vozidla automaticky vypočítán poplatek za parkování.

Pro účely detekce a rozpoznání SPZ se tato bakalářská práce zabývá dvěma odvětvími počítačového vidění. Prvním z nich je detekce objektů. V teoretické části jsou představeny základní principy detekce objektů, několik modelů založených na neuronových sítích a způsoby jejich vyhodnocení a porovnání. Dva z těchto algoritmů jsou pak v praktické části natrénovány na vlastním datovém setu. Tyto natrénované algoritmy slouží k získání lokace SPZ v obraze.

Znalost této lokace je důležitá pro získání výřezu SPZ z původního obrazu. Vyříznutá SPZ je pak předložena jednomu z algoritmů pro optické rozpoznávání textů, což je druhé odvětví metod počítačového vidění, kterým se tato bakalářská práce zabývá. Zde jsou v teoretické části rovněž představeny základní principy tohoto odvětví a to jak obecné, tak přímo principy rozpoznávání SPZ. Také jsou zde popsány dva algoritmy. Jeden přímo pro účely rozpoznávání SPZ a druhý se širokou aplikací pro rozpoznávání textů.

Závěr praktické části je věnován kombinací různých přístupů rozpoznávání SPZ. Jsou zde implementovány dva algoritmy pro detekci textů SPZ a hodnotí se jejich úspěšnost a rychlost jak samostatně, tak v kombinaci s oběma natrénovanými modely pro detekci objektů SPZ.

1 Teoretická část

1.1 Rozpoznávání registračních značek

Registrační značky (zkráceně RZ) slouží jako jednoznačné a unikátní identifikační označení každého motorového vozidla, přívěsu, či návěsu. Termín registrační značka je oficiální legislativní název zavedený v České republice roku 2001. Nicméně v praxi se stále více používá termín státní poznávací značka (zkráceně SPZ).

Od rozmachu metod počítačového vidění se tyto metody začaly hojně využívat i pro účely automatického rozpoznávání SPZ a v současnosti existuje celá řada komplexních, již vyvinutých systémů pro tyto účely. Jedním z nich je GPP LPR od společnosti GREEN Center, největší české společnosti vyvíjející parkovací systémy. Tento systém se vyznačuje vysokou přesností a nízkou chybovostí při rozpoznávání znaků SPZ. Poskytuje i celou řadu doplňkových funkcí pro větší komfort při parkování, jako je například navádění nově přijíždějícího auta na nejbližší volné parkovací místo [4].

Existuje také už i velké množství softwarových programů zajišťujících rozpoznání SPZ po předložení obrazu z kamery. Jedním z nich je Carmen od maďarské společnosti Adaptive Recognition, který se vyznačuje schopností čtení registračních značek všech států světa a má široké využití nejen na parkovištích, ale i v dopravě [2].

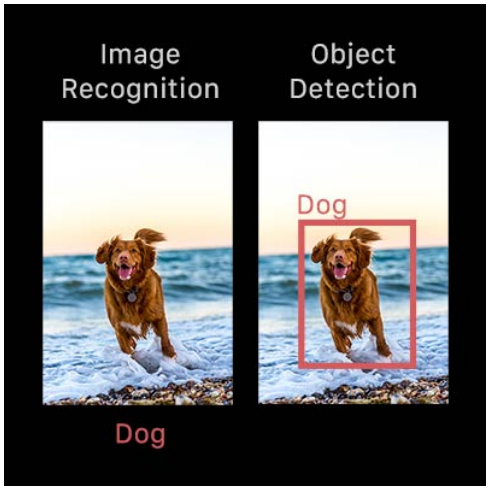
Tyto systémy a softwarové programy jsou ovšem velmi drahou záležitostí. V současné době je k dispozici již celá řada volně dostupných (tzv. open-source) technologií, pomocí kterých se dá softwarová část této problematiky zaštítit. Co se týče hardwaru, na kterém se dá takový systém poté implementovat, tak se jako ideální řešení jeví mikropočítač Raspberry Pi s jeho oficiálním kamerovým modulem V2. Tento celý hardwarový set se dá pořídit do 1500 Kč. V porovnání s cenou hotových systémů, která se pohybuje v řádech deseti tisíců korun je to zanedbatelná částka.

Ovšem, že profesionální systémy jsou mnohem lépe vyladěné a poskytují mnoho užitečných doplňkových funkcí. Pro běžné použití však postačuje i systém postavený na volně dostupných softwarových technologiích a zmíněném hardwaru.

1.2 Rozpoznávání obrazů a detekce objektů

Detekce objektů je úloha počítačového vidění umožňující identifikovat a klasifikovat objekty v obraze, nebo ve videu. Algoritmus nalezne pozici objektu a nakreslí kolem ní rámeček. Každý nalezený objekt je dále klasifikován do jedné ze tříd a je také vyjádřena míra důvěry pro náležitost k příslušné třídě.

Detekce objektů (Object Detection) je často zaměňována s jinou úlohou počítačového vidění, a to sice s rozpoznáváním obrazů (Image Recognition). Algoritmy pro rozpoznávání obrazů na rozdíl od těch pro detekci objektů poskytují pouze informaci o tom, jaký objekt se v daném obraze nachází. Neposkytují však informaci o počtu objektů v obraze, nebo o jejich přesné lokaci. Například, když bude algoritmu pro rozpoznávání obrazů předložen obrázek, na kterém jsou dva psi, tak vrátí jen informaci o tom, že na obrázku je pes. Na rozdíl od algoritmu pro detekci objektů, který rozpozná v obrázku dva psy a vrátí i jejich přesnou lokaci v podobě rámečků okolo každého z nich [1]. Rozdíl mezi úlohou detekce objektů a rozpoznání obrazů je možné vidět na obrázku 1.

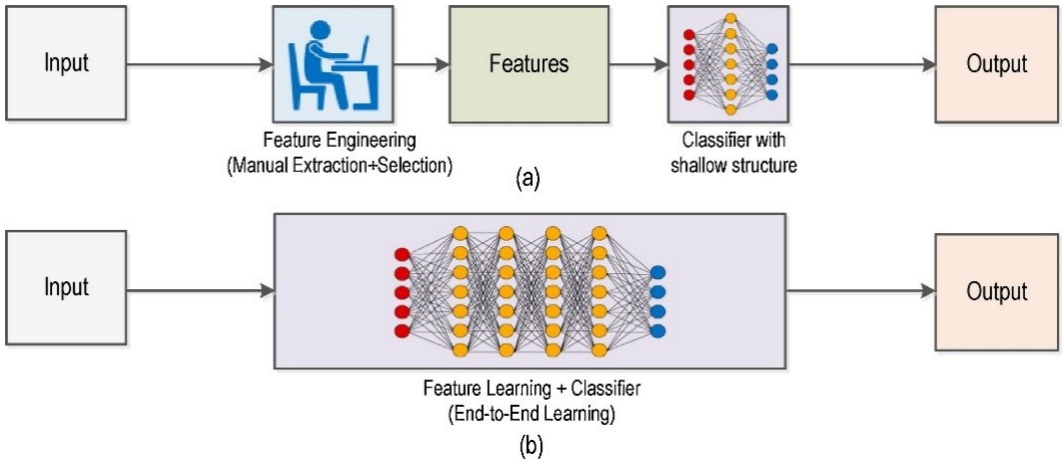


Obrázek 1: Rozdíl mezi dvěma úlohami počítačového vidění - rozpoznáváním obrazů a detekcí objektů [1]

Pro účely této bakalářské práce je důležité vědět nejen, že se v obraze nachází SPZ, ale také přesný počet SPZ v obraze a jejich umístění, aby mohla být oblast SPZ z obrazu vyříznuta a dále předána algoritmu pro rozpoznání textů SPZ. Z tohoto důvodu se tato bakalářská práce dále zabývá jen algoritmy pro detekci objektů.

1.2.1 Typy algoritmů

Pro úlohu detekce objektů existují v zásadě dva typy algoritmů a oba z nich jsou založeny na strojovém učení. Prvním typem jsou tradiční algoritmy, které můžeme zařadit k expertním systémům. Zakládají se totiž na rozpoznávání různých vlastností (Features) daného obrazu (Feature Engineering), jako jsou například identifikace skupin pixelů, nebo histogramu hran. Tyto vlastnosti jsou pak přidány do regresivního modelu (Classifier with shallow structure), který slouží jako klasifikátor a predikuje umístění daného objetu a také rámeček pro vizualizaci jeho lokace. Pro účely tohoto klasifikátoru může být použita neuronová síť.



Obrázek 2: Rozdíl mezi dvěma typy algoritmů pro detekci objektů - tradičními (a) a těmi založenými pouze na hlubokých neuronových sítích (b) [16]

Druhým, modernějším a v současnosti více používaným typem jsou metody detekce objektů založené pouze na neuronových sítích. Rozdíl mezi těmito dvěma typy algoritmů je popsán na obrázku 2.

1.2.2 Metody založené na neuronových sítích

Metody detekce objektů založené pouze na hlubokých neuronových sítích mají oproti tradičním metodám výhodu, že při jejich používání není nutné analyzovat jednotlivé vlastnosti daného obrazu zvlášť, nýbrž je možné předložit algoritmu obraz na vstup jako celek. Jedná se o tzv. end-to-end přístup a je při něm natrénován celý proces včetně analýzy a výběru příznaků daného obrazu. [1]. Z tohoto důvodu jsou tyto metody mnohem používanější

a zabývá se jimi i tato bakalářská práce.

V praktické části této bakalářské práce jsou natrénovány dva z nejvíce používaných algoritmů zakládajících se na přístupu end-to-end. Prvním z nich je YOLO, který pracuje na principu rozdělení vstupního obrazu do mřížky specifických rozměrů a za detekci objektů poté zodpovídají jednotlivé buňky této mřížky. Druhým natrénovaným algoritmem je Faster R-CNN, který je nejčastěji používanou verzí algoritmů pro detekci objektů z rodiny R-CNN. Dalšími velmi používanými algoritmy založenými na přístupu end-to-end jsou DETR a SSD.

1.2.3 Algoritmy pro detekci objektů

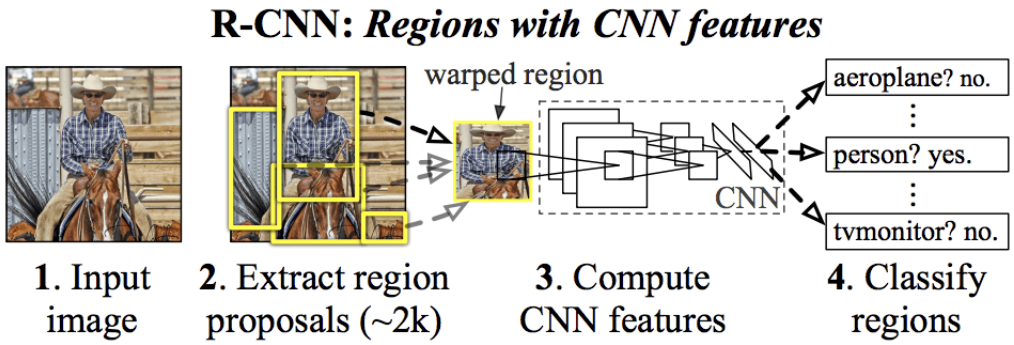
V této kapitole je popsáno pět algoritmů pro detekci objektů. První tři jsou z rodiny R-CNN a jedná se o průkopníky v oblasti algoritmů pro detekci objektů. Další dva, o něco modernější, jsou YOLO a DETR.

1.2.3.1 R-CNN

R-CNN byl poprvé popsán v publikaci "Rich feature hierarchies for accurate object detection and semantic segmentation" v roce 2014. Jedná se o jeden z prvních úspěšných modelů pro detekci a lokalizaci objektů založených na neuronových sítích.

Model je složen ze tří základní částí. První část po obdržení vstupního obrazu (Input image) pomocí algoritmu zvaného Selective search vygeneruje 2000 regionů (Extract region proposal), které potenciálně obsahují nějaký objekt. Ve druhé části je každý z těchto regionů přiveden na vstup neuronové sítě, která určí vlastnosti daného regionu (Compute CNN features). Třetí část pak zajišťuje klasifikaci daného regionu pomocí lineárních SVM klasifikátorů (Classify regions) a je zde určeno, jaký objekt se v daném regionu nachází [18]. Grafický popis těchto třech částí je zobrazen na obrázku 3.

R-CNN dosáhl na datasetu PASCAL VOC 2010 obsahující 20 tříd objektů v kategoriích osoby, zvířata, domácí vybavení a vozidla [15] hodnoty mAP 53.7 % [18]. Nicméně má tři zásadní problémy. První problém je jeho nemalá časová a paměťová náročnost jak při trénovaní, tak během procesu detekce objektů. Tento problém je způsoben tím, že pro každý ze 2000 vygenerovaných regionů obsahující potenciálně nějaký objekt jsou pomocí neuronové sítě nalezeny vlastnosti, ale některé vygenerované regiony se překrývají. Tím dochází zbytečně k nalezení některých vlastností několikrát. Druhým

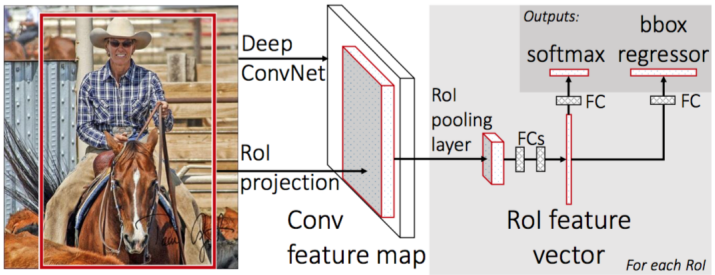


Obrázek 3: Tři základní části modelu R-CNN [18]

problémem je, že algoritmus Selective search pro vygenrování těchto 2000 regionů není nijak natrénován a může tedy generovat chybné regiony. Navíc je také velmi časově náročný. Posledním problémem modelu je jeho struktura. Celý model je rozdělený do třech samostatných částí, které musí být natrénovány samostatně, a z toho důvodu je obtížné najít jeho globální optimální nastavení. [11].

1.2.3.2 Fast R-CNN

Dva ze zmíněných problémů R-CNN řeší vylepšená verze tohoto modelu zvaná Fast R-CNN. Tento model může být trénovat najednou (end-to-end), je tedy snazší najít zmíněné optimální nastavení. Přináší také zlepšení v rychlosti detekce, protože již nedochází ke zbytečnému vícenásobnému generování vlastností některých částí obrazu. Nicméně problém se Selective search stále setrvává [11].



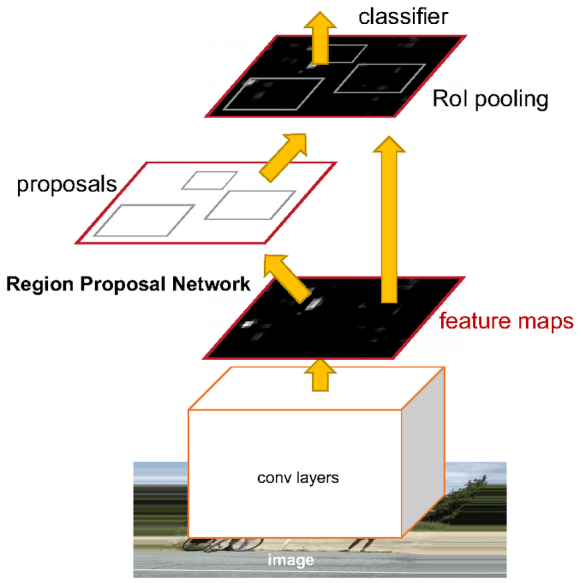
Obrázek 4: Architektura Fast R-CNN [11]

Vstupem do Fast R-CNN je obraz a sada regionů, kde by se potenciálně mohl nacházet nějaký objekt (Region of Interest - RoI). Celý obraz je

nejprve zpracován sekvencí konvolučně propojených vrstev a je vytvořena mapa vlastností pro celý obraz (Conv feature map). Poté je z této mapy pomocí speciální vrstvy (RoI pooling layer) vybrán vektor příznaků RoI (RoI feature vector) a tento vektor je současně přiveden na vstup dvou sekvencí s plně propojenými vrstvami (FC). Každá z nich poskytuje jeden výstup z celého procesu. První výstupem je pravděpodobnost náležitosti objektu k dané třídě a také pravděpodobnost přítomnosti pozadí spočítané pomocí funkce softmax. Druhým výstupem jsou pak čtyři souřadnice rámečku pro každý objekt [11]. Architektura Fast R-CNN je zobrazena na obrázku 4.

1.2.3.3 Faster R-CNN

Řešení posledního problému R-CNN přináší model Faster R-CNN. Zde už není použit algoritmus Selective search pro účely návrhu regionů obsahující potenciálně nějaký objekt. Tento úkol zastává neuronová síť.

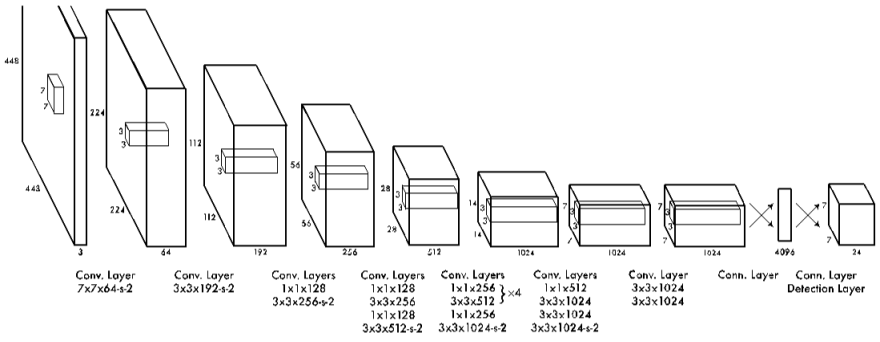


Obrázek 5: Architektura Faster R-CNN [19]

Vstupem je pouze obraz a podobně jako v případě Fast R-CNN je sekvencí konvolučních vrstev (conv layers) vytvořena mapa vlastností celého obrazu (feature maps). Následuje zmiňovaná hluboká neuronová síť tzv. Region Proposal Network, která navrhuje regiony, kde by se potenciálně mohl nacházet nějaký objekt (proposals). Dále je postup detekce obdobný jako v případě Fast R-CNN. Navržené regiony obsahující potenciálně nějaký objekt jsou předány tzv. RoI pooling sekvenci s plně propojenými vrstvami, která pro každý z nich vybere dané vlastnosti z mapy vlastností celého obrazu. Opět tedy vznikají vektory příznaků pro každý z těchto regionů, které jsou předány dvěma výstupním sekvencím s plně propojenými vrstvami. Výstupem ze sítě jsou opět dvě hodnoty pravděpodobnosti (pro třídu objektu a pozadí) a čtyři souřadnice rámečku okolo objektu [19]. Architektura Faster R-CNN je zobrazena na obrázku 5.

1.2.3.4 YOLO

YOLO byl poprvé popsán v publikaci "You Only Look Once: Unified, Real-Time Object Detection"z roku 2015 jako první model, který detekuje objekty jednofázově. Není tedy rozdělen na jednotlivé části pro generování regionů potencionálně obsahující nějaký objekt, pro klasifikaci jejich vlastností a pro jejich vyhodnocení. Celý model tvoří jedna neuronová síť se 24 konvolučními vrstvami a dvěma plně propojenými vrstvami na výstupu sítě [14]. Architektura sítě je zobrazena na obrázku 6.



Obrázek 6: Architektura YOLO [14]

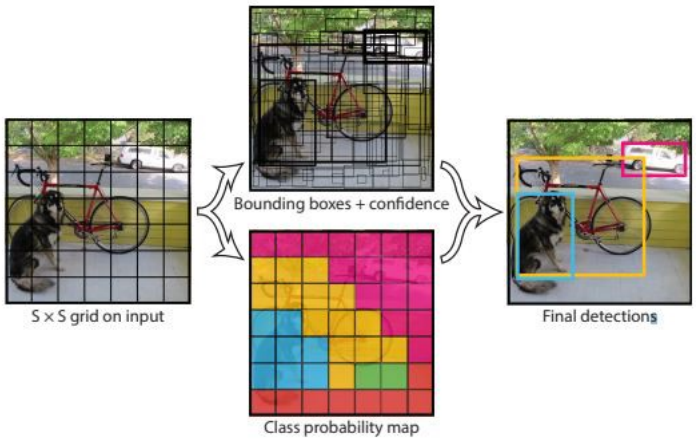
Vstupní obraz je systémem modelu nejprve rozdělen do mřížky S x S. Poté platí, že pokud se střed nějakého objektu, který se vyskytuje v obraze, nachází v jedné z buněk, tak je právě tato buňka zodpovědná za jeho detekci. Každá buňka může predikovat až B rámečků a také hodnotu spolehlivosti. Tato hodnota popisuje důvěru modelu k tomu, že se objekt v daném rámečku

nachází a popisuje také přesnost umístění rámečku. Pokud se v buňce objekt nenachází, tak tato hodnota je nula. V ostatních případech je rovna hodnotě IoU mezi predikovaným a skutečným rámečkem [14].

Hodnota spolehlivosti je prvním výstupem z modelu, který náleží každému predikovanému rámečku. Další čtyři výstupy se týkají umístění predikovaného rámečku v obraze. Hodnoty (x, y) určují střed predikovaného rámečku relativně k buňce a hodnoty (w, h) určují šířku a výšku predikovaného rámečku relativně k rozměrům celého obrazu. Všechny tyto výstupy ze sítě náležící ke každému predikovanému rámečku jsou v rozmezí mezi 0 a 1. Každá buňka také predikuje C pravděpodobností, přičemž každá z nich náleží jedné ze tříd, na které byl model natrénován. Tento počet C pravděpodobností je nezávislý na počtu predikovaných rámečků B [14]. Pro každou buňku je tedy výstupem ze sítě následující vektor:

$$y = [P_1\ x_1\ y_1\ w_1\ h_1\ P_2\ x_2\ y_2\ w_2\ h_2\ ...\ P_B\ x_B\ y_B\ w_B\ h_B\ c_1\ c_2\ ...\ c_C]$$

Celkový výstup sítě je zakódován v tenzoru o dimenzi S x S x (B * 5 + C). Pro každý objekt může být v průběhu trénování modelu predikováno danou buňkou více rámečků, nicméně na konci je zachován jen jeden a to sice ten s největší hodnotou IoU vzhledem ke skutečnému (požadovanému) rámečku okolo daného objektu. Jsou také odstraněny rámečky, jejichž spolehlivost je menší, než je zvolený práh [14].



Obrázek 7: Postup detekce objektů modelem YOLO [14]

Postup detekce objektů pomocí YOLO je zobrazen na obrázku 7 a dá se shrnout do třech následujících kroků:

1. Rozdělení vstupního obrazu na vstupu do mřížky (S x S grid on output)
2. Predikce rámečků a tříd v každé buňce (Bounding boxes + confidence, Class probability map)
3. Výběr rámečků s největšími hodnotami spolehlivosti (Final detections) [14]

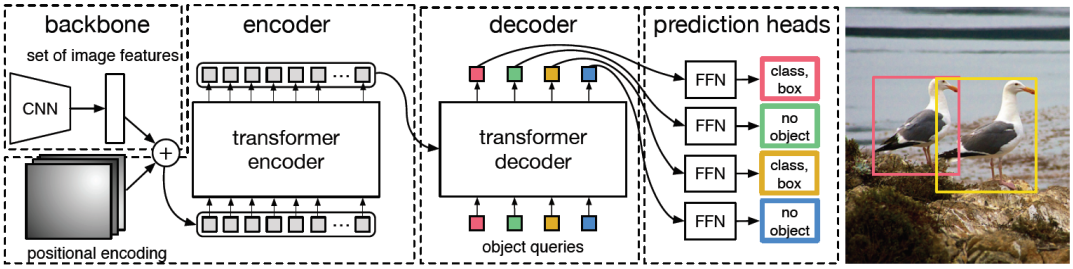
1.2.3.5 DETR

Nejnovějším z jednofázových modelů pro detekci objektů je DETR, který byl publikován v práci s názvem "End-to-End Object Detection with Transformers" v roce 2020. Jedná se o zcela nový přístup detekce objektů. Model obsahuje kromě konvoluční neuronové sítě na vstupu a neuronové sítě s plně propojenými vrstvami na výstupu i dva transformery použité pro účely enkodéru a dekodéru [17].

Transformery jsou novým stavebním kamenem algoritmů pro strojový překlad, zpracování řeči, či počítačové vidění, založeným na blocích mechanismů pozornosti. Mechanismus pozornosti je sekvence vrstev neuronové sítě, která shlukuje informace ze vstupní sekvence, kterou je například text k překladu. Velkou výhodou bloků mechanismů pozornosti je jejich schopnost provádět globální výpočty při použití malého obsahu paměti. To z nich dělá daleko přijatelnější řešení pro dlouhé sekvence, než jakým jsou rekurentní neuronové sítě [17].

Detekce objektů pomocí DETR probíhá ve třech základních krocích. V prvím kroku je vstupní obraz přiveden na vstup konvoluční neuronové sítě (CNN), která najde soubor vlastností daného obrazu (set of image features) [17].

Tento soubor vlastností je ve druhém kroku přiveden na vstup enkodéru, kde je nejprve redukován jednou konvoluční vrstvou a prostorové dimenze jsou sbaleny do jedné. Každá vrstva enkodéru má standardní architekturu tvořenou blokem mechanismu pozornosti a dopřednou neuronovou sítí. Protože je architektura transformeru permutačně-invariantní, což znamená, že nepředpokládá žádné prostorové vztahy mezi prvky, tak je ještě nutné na vstup každé vrstvy mechanismu pozornosti enkodéru navíc přivést data o pozičním kódování (positional encoding). Enkodér na základě předložených



Obrázek 8: Architektura modelu pro detekci objektů DETR [17]

vstupů zpracuje výstup, který je pak vstupem dekodéru. Architekrura dekodéru je totožná se standardní architekturou transformeru s tím rozdílem, že v tomto případě zpracovává jeden objekt v každé ze svých vrstev. Tato architektura je rovněž permutačně-invariantní a tak je nutné opět na vstup každé vrstvy mechanismu pozornosti přivést data o pozičním kódování, které v tomto případě představují dotazy na jednotlivé objekty v obraze (object queries). Pro každý z těchto dotazů na objekty dekodér na základě předložených vstupů, reprezentovanými výstupy z enkodéru, vygeneruje jeden výstup [17].

Ve třetím kroku je každý z těchto výstupů předložen dopředné neuronové síti (FFN) tvořené třemi vrstvami perceptronů s ReLU aktivační funkcemi. Tato síť každý nezávisle zpracuje. Konečným výstupem z celého modelu jsou čtyři souřadnice rámečku a informace o třídě pro každý z dotazů na jednotlivé objekty v obraze [17]. Architektura modelu DETR je zobrazena na obrázku 8.

1.2.3.6 Detectron2

Detectron2 je knihovna vyvinutá výzkumným týmem pro umělou inteligenci společnosti Facebook poskytující moderní algoritmy pro detekci a segmentaci objektů. Hlavním cílem této knihovny je podpora výzkumných projektů v odvětví počítačového vidění a produkčních aplikací Facebooku [3].

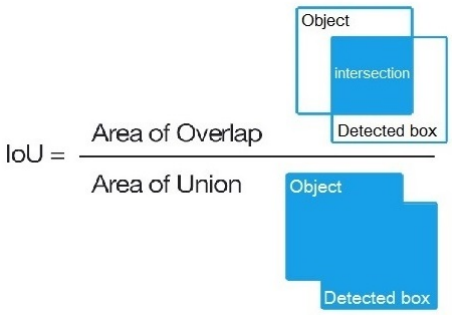
V oblasti detekce objektů poskytuje tato knihovna mnoho nových funkcionalit, jako je například rotace ohraničujících rámečků okolo objektů. Trénování algoritmů, které knihovna poskytuje, je při jejím použití mnohem rychlejší, než při použití jejich původních implementací [3]. V praktické části je pomocí této knihovny pro účely detekce objektů SPZ v obraze dotrénován na vlastních datech jeden z modelů Faster R-CNN.

1.2.4 Míry

Když je natrénován detektor objektů, tak dalším krokem bývá ověření jeho fungování. Natrénovaný model je obvykle schopný najít objekty, k jejichž nalezení byl natrénován. Tento fakt však nestačí k vyčíslení jeho úspěšnosti a už vůbec ne k porovnání příslušného modelu s jiným. K tomu se užívají míry, což jsou veličiny, které lze spočítat při ověřování modelu testovací sadou obrazů [20].

1.2.4.1 Intersection over Union (IoU)

Základní míra pro ověření správnosti fungování detektoru objektů se nazývá Intersection over Union (zkráceně IoU) a udává přesnost detektorem predikovaného rámečku. Je definována jako poměr sjednocení (Area of Overlap) ploch detektorem predikovaného rámečku (Detected box) se skutečným rámečkem (Object) a jejich průnikem (Area of Union) [20]. Vzorec a grafické znázornění IoU je vidět na obrázku 9.



Obrázek 9: Vzorec a grafické znázornění míry IoU [20]

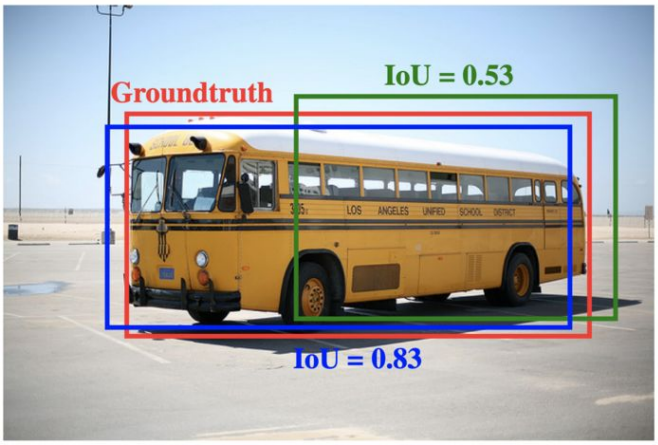
Hodnota IoU nabývá hodnot mezi 0 a 1. Když je tato hodnota rovna 1, tak se jedná o naprostou shodu predikovaného a skutečného rámečku. Když je hodnota IoU větší, než práh (nejčastěji 0.5), tak se rámeček považuje za správně predikovaný [20].

1.2.4.2 Míry odvozené od IoU

Vypočtením IoU pro všechny všechny objekty na všech obrazech z testovací sady je dosaženo přehledu o tom, kolik detekcí bylo vyhodnoceno jako:

- **True Positive (zkráceně TP)**, což znamená počet predikovaných rámečků s hodnotou IoU > práh (nejčastěji 0.5)

- **False Positive (zkráceně FP)**, což znamená počet predikovaných rámečků, tam kde se žádný objekt nenachází
- **False Negative (zkráceně FN)** , což znamená počet vůbec nepredikovaných rámečků okolo existujících objektů, nebo rámečků predikovaných s hodnotou IoU <= práh (nejčastěji 0.5) [20]



Obrázek 10: Příklad dvou rámečků s různými hodnotami IoU spočítaných vůči skutečnému (požadovanému) rámečku (Groundtruth) [20]

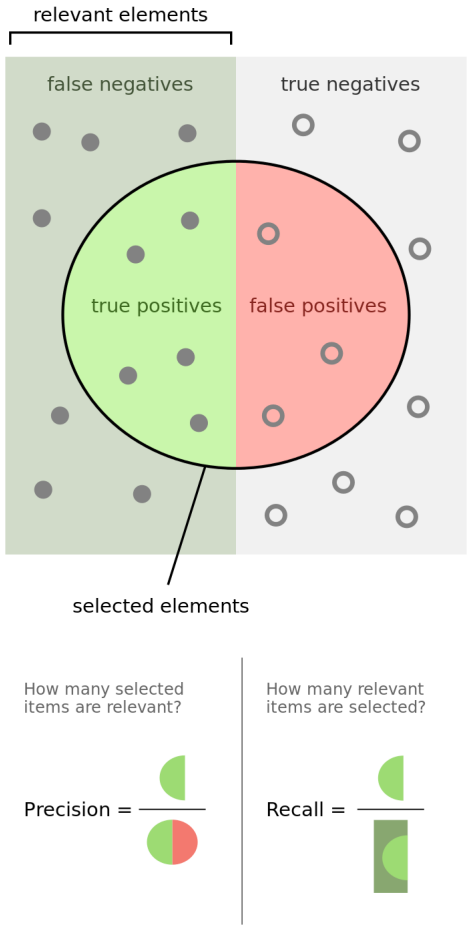
Při znalosti těchto počtů je možné vypočítat dvě důležité odvozené míry, konkrétně:

- **Precision**, která udává procento správně predikovaných rámečků ze všech predikovaných. Jinak řečeno, je to pravděpodobnost, že bude rámeček správně predikován.

$$Precision = \frac{TP}{TP + FP}$$

- **Recall**, která udává procento správně detekovaných objektů. Například recall 0.9 znamená, že 90 % objektů bylo správně detekováno [20].

$$Recall = \frac{TP}{TP + FN}$$



Obrázek 11: Grafické znázornění mír Precision a Recall a jejich vztahu k TP, FP a FN [5]

Třetí odvozenou mírou je **F1 skóre**, které lze vypočítat z Precision a Recall pomocí následujícího vztahu:

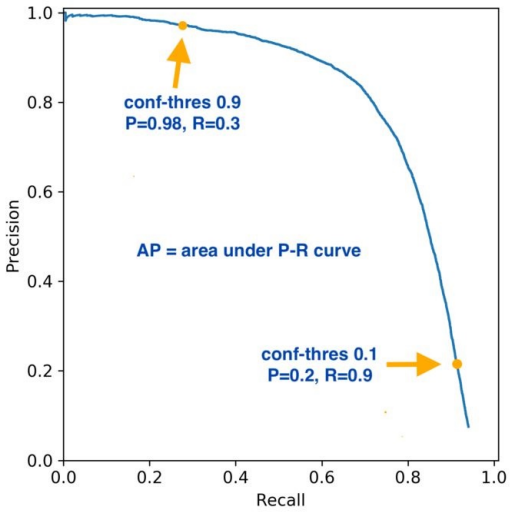
$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

1.2.4.3 mean Average Precision (mAP)

Poslední mírou poskytující vyhodnocení detektoru objektů je mean Average Precision (zkráceně mAP). Pro její výpočet je nutné znát hodnoty mír Precison a Recall popsaných víše.

Modely detekce objektů jsou většinou vyhodnocovány tak, že jsou spočítány hodnoty IoU všech rámečků pro různé hodnoty prahů rozhodující o správnosti predikce rámečku. Například nejen pro hodnotu prahu 0.5, ale i

pro hodnoty prahů v rozmezí 0.2 až 0.9 s krokem 0.05 [10]. Touto procedurou lze získat graf závislosti míry Precission na míře Recall pro různé hodnoty prahů IoU tak jak je to vidět na obrázku 12.



Obrázek 12: Příklad grafu závislosti míry Precission na míře Recall pro různé hodnoty prahů IoU (conf-thres) a míry AP reprezentované plochou pod křivkou v tomto grafu [20]

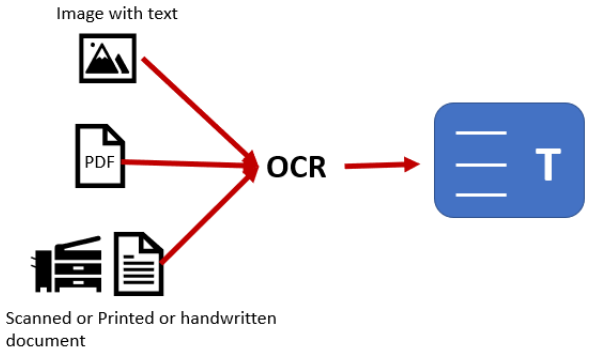
Plocha pod křivkou v tomto grafu (area under P-R curve) je rovna hodnotě **Average Precision (zkráceně AP)**, což je hodnota této míry pro jednu třídu. Pro získání hodnoty mAP pro všechny třídy je nutné hodnoty AP pro jednotlivé třídy zprůměrovat pomocí vzorce

$$mAP = \frac{1}{n} \sum_{k=1}^{n} AP_k \,,$$

kde AP_k je hodnota AP pro jednu třídu a n je počet tříd [10].

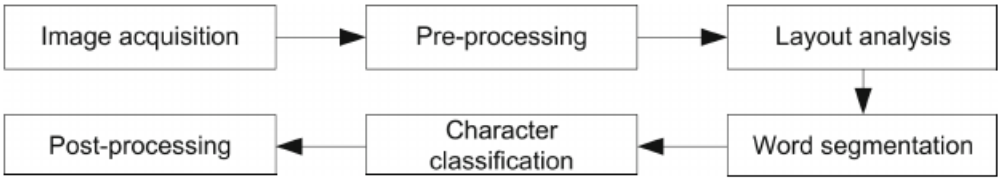
1.3 Optické rozpoznávání textů

Optické rozpoznávání znaků či OCR (z anglického názvu Optical Character Recognition) je jedním z nejstarších odvětví počítačového vidění. Zabývá se převodem naskenovaného, vytištěného, či ručně psaného textu (Scanned or Printed or handwritten document) na text, který může být zpracován počítačem. Dá se využít i na převod textu z obrazu (Image with text), kterým se pro účely rozpoznávání SPZ zabývá tato bakalářská práce.



Obrázek 13: Vizualizace použití optického rozpoznávání textů pro různé účely [9]

Optické rozpoznávání znaků je komplexní problém, zejména u ručně psaných textů, kde tuto komplexnost způsobuje mnoho stylů psaní. Jinými slovy, ten samý znak může mít pokaždé jinou velikost a také může být jinak naklopený [8]. V tištěných textech tuto komplexnost způsobují různé fonty písma. V případě rozpoznávání textů SPZ může mít například SPZ jiného státu odlišný font, než ta česká.



Obrázek 14: Blokové schéma jednotlivých kroků procesu optického rozpoznávání znaků [9]

Proces optického rozpoznávání znaků zahrnuje několik kroků. Prvním krokem je získání obrazu (Image acquisition), kdy je obraz pořízen například fotoaparátem, kamerou, nebo je naskenován skenerem. Následuje předzpracování obrazu (Pre-processing), kde je obraz upraven. Například je zde odstraněn šum, upravena velikost obrazu, nebo jsou vyhlazeny hrany objektů. Poté je pro účely zachycení struktury textu analyzováno rozložení jednotlivých znaků (Layout analysis) a následuje segmentace textu na jednotlivé znaky (Word segmentation). Předposledním krokem je samotná klasifikace jednotlivých znaků (Character clasification), kdy se používá rozpoznávacích vzorů. Každý znak je porovnán se všemi vzorovými a přiřadí se k tomu, kde je nalezena největší shoda. Posledním krokem je následné zpracování

(Pre-processing), kdy jsou nalezené znaky shromážděny a jsou spojovány do slov. Tento proces je opakem procesu segmentace [8]. Blokové schéma popisující jednotlivé kroky procesu optického rozpoznávání znaků je zobrazeno na obrázku 14.

1.3.1 Rozpoznávání textů SPZ

Jedním z konkrétních oblastí, kde se dá optické rozpoznávání textů využít je rozpoznávání textů SPZ. Tato úloha obsahuje v zásadě dvě základní části, konkrétně:

1. Nalezení SPZ - ta se provádí pomocí natrénovaného detektoru objektů a nejčastěji je tento detektor na bázi neuronové sítě
2. Extrakce textu z detekované SPZ - tu zajišťuje algoritmus pro optické rozpoznávání znaků

Specializované algoritmy pro rozpoznávání textů SPZ zpravidla obsahují obě tyto části. V následující kapitole je popsán jeden z nich.

1.3.2 Algoritmy pro rozpoznávání textů

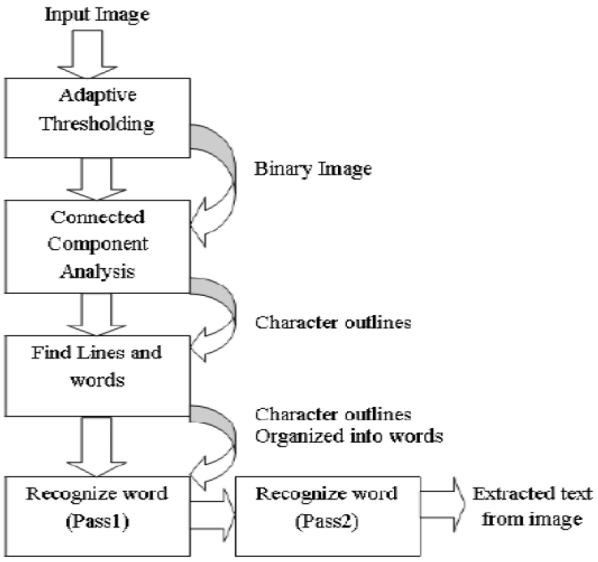
V této kapitole jsou popsány dva algoritmy pro rozpoznávání textů. Prvním z nich je Tesseract OCR, což je univerzální algoritmus, který je použitelný pro širokou škálu úloh od rozpoznávání textů z dokumentů, až po rozpoznávání drobných textů z obrazů. Druhým z nich je OpenALPR, což je specializovaný algoritmus pro rozpoznávání textů SPZ. Oba tyto algoritmy jsou pak v praktické části implementovány v programovacím jazyce Python a vyzkoušeny pro účely rozpoznávání textů SPZ.

1.3.2.1 Tesseract OCR

První verze Tesseract OCR byla vyvinuta v laboratořích Hawlett Packard mezi lety 1985 a 1994. V roce 1996 byl Tesseract OCR upraven pro účely použití v operačním systému Windows a v roce 1998 byl přepsán z jazyku C do C++. Jako volně dostupný software (open-source) je Tesseract OCR k dispozici od roku 2005. Vývoj softwaru je od roku 2006 sponzorován společností Google. Poslední verze, Tesseract v4.1.0, pochází z července 2019 [13].

Tesseract OCR je široce rozšířený software pro optické rozpoznávání znaků podporující širokou škálu jazyků. Jako knihovnu je ho možné implementovat v mnoha programovacích jazycích. Navíc poskytuje trénovací

metody pro vývojářské účely. Uživatelé ho tedy mohou natrénovat na rozpoznávání svých vlastních znaků [13].



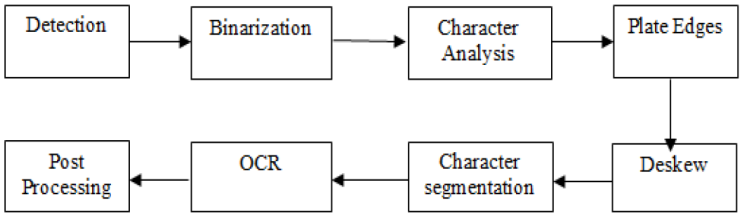
Obrázek 15: Základní kroky procedury optického rozpoznávání znaků pomocí Tesseract OCR [9]

Optické rozpoznávání znaků pomocí Tesseract OCR probíhá ve čtyřech základních krocích:

1. Převod vstupního obrazu na binární obraz (Adaptive Tresholding)
2. Extrakce okrajů znaků (Connected component analysis), jinými slovy jde o nalezení hran
3. Nalezení řádek a slov (Find lines and words) a vytvoření tzv. Blob objektů, které jsou poté spojeny dohromady za účelem vytvoření řádky textu k přečtení
4. Samotné rozpoznání probíhá ve dvou oddělených krocích
   - Rozpoznání jednotlivých písmen pomocí klasifikátoru (Pass 1)
   - Rozpoznání jednotlivých slov skládajících se z rozpoznaných písmen (Pass 2)

1.3.2.2 OpenALPR

OpenALPR je volně dostupná knihovna specializovaná pro rozpoznávání textů SPZ napsaná v programovacím jazyce C++. Tuto knihovnu je možné implementovat v mnoha programovacích jazycích, jako jsou například Python, Java, nebo C#. Vstupem do algoritmu může být obraz, nebo video. Výstupem je seznam možných kandidátů na SPZ v textové podobě spolu s hodnotou důvěry (Confidence) pro každého kandidáta z tohoto seznamu. Tato hodnota vyjadřuje do jaké míry algoritmus věří, že se jedná právě o tuto SPZ. Rozsah seznamu těchto kandidátů je volitelný a je ho možné nastavit při konfiguraci knihovny. Stejně tak se dá specifikovat stát, ze kterého budou rozpoznávané SPZ.



Obrázek 16: Blokové schéma znázorňující zpracování vstupního obrazu a jeho vyhodnocení pro účely převodu SPZ do textové podoby pomocí OpenALPR [7]

Zpracování vstupního obrazu a jeho vyhodnocení pro účely převodu SPZ do textové podoby probíhá v následujících etapách:

- Detekce (Detection) - nalezení potenciálních regionů, kde se může nacházet SPZ
- Binarizace (Binarization) - převedení obrazu do formy obsahující jen bílou a černou barvu
- Analýza znaků (Character Analysis) - nalezení tzv. Blobů (oblastí o velikosti znaků) v potencionálních regionech
- Analýza hran (Plate edges) - nalezení hran a tvaru SPZ
- Deskew - transformace perspektivy pro účely přímého pohledu na základě ideálního tvaru SPZ
- Segmentace znaků (Character segmentation) - izolace jednotlivých znaků a jejich pročištění, aby mohly být analyzovány jednotlivě

- OCR - analýza každého znaku a poskytnutí více možností, o jaký znak jde
- Následné zpracování (Post Processing) - vytvoření seznamu kandidátů SPZ v textové podobě [12]

1.3.3 Způsoby vyhodnocení rozpoznávání textů

V této bakalářské práci jsou pro účely vyhodnocení správnosti rozpoznaného textu SPZ použity dva přístupy. První z nich považuje za správně rozpoznaný text SPZ jen ten, který přesně odpovídá skutečnému textu SPZ. To znamená, že rozpoznaný text SPZ musí obsahovat všechny znaky jako skutečná SPZ a na stejných pozicích. Druhý přístup využívá Levenshteinovy vzdálenosti dvou řetězců.

1.3.3.1 Levenshteinova vzdálenost

Levenshteinova vzdálenost mezi dvěma řetězci je definovaná jako minimální počet operací vkládání, mazání a substituce takových, aby po jejich provedení byly zadané řetězce totožné. Levenshteinova vzdálenost byla zavedena v roce 1965 Vladimirem Levenshteinem [6].

Výpočet Levenshteinovy vzdálenosti je postaven na principu procházení a vyplňování matice, kde řádky odpovídají znakům řetězce A a sloupce znakům řetězce B [6]. Příklad počáteční podoby matice je zobrazen na obrázku 17.

		S	a	t	u	r	d	a	y
	0	1	2	3	4	5	6	7	8
S	1								
u	2								
n	3								
d	4								
a	5								
y	6								

Obrázek 17: Příklad počáteční podoby matice pro výpočet Levenshteinovy vzdálenosti [6]

		S	a	t	u	r	d	a	y
	0	1	2	3	4	5	6	7	8
S	1	0	1	2	3	4	5	6	7
u	2	1	1	2	2	3	4	5	6
n	3	2	2	2	3	3	4	5	6
d	4	3	3	3	3	4	3	4	5
a	5	4	3	4	4	4	4	3	4
y	6	5	4	4	5	5	5	4	**3**

Obrázek 18: Příklad konečné podoby matice pro výpočet Levenshteinovy vzdálenosti [6]

Při procházení matice M platí pro její hodnotu na každém i-tém řádku a j-tém sloupci následující:

$$M[i][j] = min(M[i-1][j]+1,\ M[i][j-1]+1,\ M[i-1][j-1]+const)\ ,$$

kde $const = 0$, pokud $A[i] = B[j]$, jinak $const = 1$ [6]

Konečná podoba matice pro výpočet Levenshteinovy vzdálenosti z obrázku 17 je zobrazena na obrázku 18. Výsledná hodnoty Levenshteinovy vzdálenosti je vždy rovna poslednímu spočtenému prvku matice M a v tomto případě je to 3 [6].

2 Praktická část

2.1 Trénovací a testovací data

Aby mohla být natrénována jakákoliv neuronová síť, tak jsou zapotřebí trénovací data. Obvykle platí, že čím více jich je, tím lépe. Jednou z praktických částí této bakalářské práce je trénování modelů pro detekci objektů SPZ. Pro účely tohoto trénování jsou vytvořeny dvě trénovací datové sady popsané v tabulce 1.

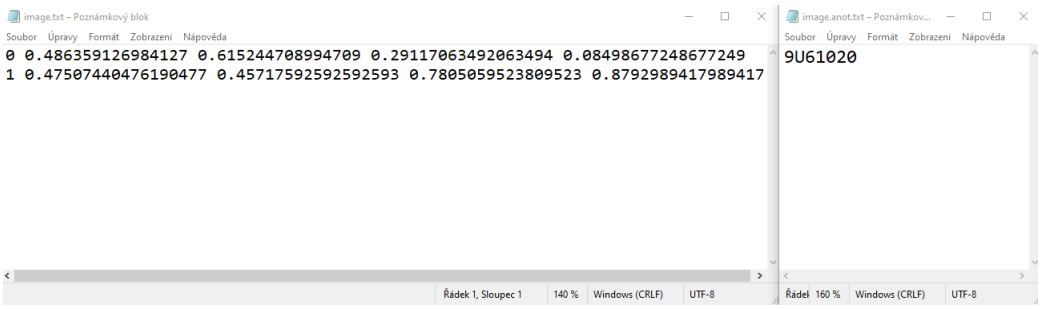
Tabulka 1: Trénovací sady dat

Trénovací sada	Rozsah				Augmentace						
	Celkový	Trénovací část	Validační část	Testovací část	Počet augmentací v jednom obraze	Otočení	Přiblížení	Rotace	Jas	Rozostření	Šum
A	2402	2103	199	100	3	Horizontální, vertikální	0 až 30 %	-15 až 15 °	-30 až +30 %	-	-
B	2402	2103	199	100	3	Horizontální, vertikální	0 až 45 %	-25 až 25 °	-50 až +50 %	1.5 px	10 % pixelů a více

Postup vytváření trénovacích dat zahrnuje celkem čtyři kroky. V prvním kroku je pořízeno video, zpravidla na nějakém větším parkovišti, například u supermarketu. Druhým krokem je nastříhání videa na jednotlivé obrazy obsahující SPZ, auta a osoby. Pro tyto účely je použit software Lightworks (viz `https://www.lwks.com/`).

Obrázek 19: Příklad anotace jednoho obrazu (červený rámeček vždy náleží třídě 0 - SPZ a zelený třídě 1 - auto)

Ve třetím kroku jsou nastříhané obrazy anotovány. Anotace znamená, že jsou v každém obraze pomocí rámečků vyznačeny objekty, které chceme, aby model po natrénování byl schopný detekovat. Pro účely této anotace je vytvořen program v programovacím jazyce Python. Jako výstup program vytvoří pro každý obraz dva textové soubory. První textový soubor obsahuje anotace ve formátu YOLOv5 PyTorch TXT. Tento anotační formát poskytuje pro každý objekt (rámeček okolo něj) jeden řádek v tomto souboru. Tento řádek obsahuje pět čísel. Prvním z nich je číslo třídy, v tomto případě 0 pro SPZ, 1 pro auto a 2 pro osobu. Další čtyři čísla obsahují informace o lokaci rámečku, ve kterém se objekt nachází. První dvě jsou souřadnice středu rámečku a druhé dvě jeho šířka a výška. Všechny tyto čtyři čísla jsou přepočítány na relativní hodnoty vůči šířce a výšce celého obrazu. Jsou tedy v rozmezí 0 až 1. Druhý textový soubor obsahuje pro každou SPZ řádek, na němž je textový řetězec dané SPZ zaznamenaný programem jako vstup při anotaci. Příklad anotace pomocí tohoto programu je zobrazen na obrázku 19 a příklad jeho výstupů na obrázku 20. Tímto způsobem je nastříháno a anotováno 1000 obrazů.



Obrázek 20: Příklad dvou výstupních textových souborů z programu pro anotaci obrazů

Posledním krokem je vytvoření augmentací v každém obraze. Vytvoření augmentace znamená, že je obraz trochu pozměněn, například je upraven jeho jas, nebo je vertikálně, či horizontálně otočen. Účelem těchto augmentací je zajištění robustnějšího modelu po natrénování skrze trénovací data. Jinými slovy chceme, aby model byl schopný dané objekty v praxi detekovat i za zhoršených podmínek. Pro účely vytvoření těchto augmentací byly trénovací data nahrány do webové aplikace Roboflow (viz `https://roboflow.com/`). Jedná se o aplikaci pro vytváření modelů počítačového vidění, kde je mimo jiné možné ukládat trénovací data a vytvářet augmentace. Z 1000 anotovaných obrazů je v této aplikaci vytvořeno 2402 obrazů obsahující vždy tři náhodné augmentace s náhodnými parametry. Těchto 2402

obrazů je rozděleno pro účely trénování a testování na trénovací, validační a testovací podmnožinu. Trénovací podmnožina slouží pro vlastní trénování neuronové sítě a z dat této podmnožiny je v každém kroku trénování vypočtena trénovací chyba. Validační podmnožina slouží v průběhu trénování k ověření, zda nedochází k přetrénování sítě. K tomu dochází v momentě, kdy validační chyba začíná růst. Testovací podmnožina pak slouží k otestování sítě po natrénování. Toto rozdělení a použité augmentace popisuje tabulka 1.

Stejným způsobem, akorát bez augmentací, jsou vytvořeny i čtyři testovací sady. Pro tyto účely je nastříháno a anotováno celkem 501 obrazů. Ty jsou použity pro vyhodnocení výsledků jak rozpoznání textů SPZ, tak detekce objektů SPZ. Popis těchto testovacích sad je v tabulce 2.

Tabulka 2: Testovací sady dat

Testovací sada	Rozsah	Počet objektů SPZ na obraze	Účely testování
A	103	1	Rozpoznávání textů SPZ
B	182	1	Rozpoznávání textů SPZ
C	65	1	Rozpoznávání textů SPZ
D	151	2 a více	Detekce objektů SPZ

2.2 Použití algoritmů pro rozpoznání textů SPZ

V této části jsou implementovány dva algoritmy pro optické rozpoznávání znaků a vyzkoušeny pro účely rozpoznávání textů SPZ. Vyhodnocení probíhá na testovacích sadách A, B a C (viz tabulka 2).

2.2.1 Implementace a vyhodnocení Tesseract OCR

Pro účely rozpoznávání SPZ je v této části implementován Tesseract OCR v programovacím jazyce Python. Jedná se o verzi pytesseract 0.3.7 (viz `https://pypi.org/project/pytesseract/`), což je nejnovější verze Tesseract OCR pro Python.

Text SPZ je rozpoznáván nejdřív z původního obrazu, který není pro účely rozpoznávání nijak upraven. Poté rozpoznávání probíhá z upraveného obrazu. Ten vzniká tak, že je nejprve původní obraz převeden na binární

obraz, poté je z něho odstraněn šum a nakonec jsou v něm zvýrazněny hrany. Pro tyto úpravy je použita knihovna OpenCV (viz `https://pypi.org/project/opencv-python/`). Úspěšnosti rozpoznání jsou hodnoceny jak pro celou SPZ, kdy musí celý rozpoznaný text odpovídat skutečnému, tak Levenshteinovou vzdáleností rozpoznaného řetězce od skutečného (viz kapitola 1.3.3). V obou případech je ještě rozpoznaný textový řetězec oříznut o bílé znaky a pomocí vytvořeného algoritmu je v něm nalezena česká SPZ.

Tabulka 3: Úspěšnost rozpoznávání textů SPZ pomocí Tesseract OCR a průměrný čas rozpoznání jedné SPZ

Testovací sada	Průměrná úspěšnost rozpoznání SPZ				Průměrný čas rozpoznání SPZ
	Neupravený obraz		Upravený obraz		
	Celá SPZ	Levenshteinova vzdálenost	Celá SPZ	Levenshteinova vzdálenost	
A	0.98%	18.26%	0.98%	16.79%	1643.35 ms
B	1.10%	18.07%	1.10%	18.21%	1656.04 ms
C	4.62%	23.46%	0.00%	16.92%	1333.54 ms

Jak je možné vidět v tabulce 3, tak úspěšnost rozpoznávání textů SPZ pomocí Tesseract OCR není vůbec vysoká. Nejvyšší průměrné hodnoty úspěšnosti rozpoznání 2.23 % je dosaženo z neupraveného obrazu. Je to dáno tím, že Tesseract OCR je obecný algoritmus pro optické rozpoznávání znaků a nedisponuje žádnou částí, která by dokázala detekovat SPZ. Z toho důvodu rozpoznává text nejen z oblasti SPZ, ale i z ostatních částí obrazu. To vede nejen k nízké úspěšnosti rozpoznání, ale i dlouhému času potřebnému k rozpoznání. Ani úprava obrazu před jeho předložení algoritmu v tomto případě nevede k lepším výsledkům. Pro jejich zlepšení je potřeba algoritmu předávat jen oblast SPZ, což zařídí natrénovaný detektor objektů SPZ.

2.2.2 Implementace a vyhodnocení OpenALPR

Tato část se zabývá implementací a vyhodnocením OpenALPR, což je specializovaný algoritmus pro rozpoznávání textů SPZ. V programovacím jazyce Python je implementována jeho poslední verze 1.1.0 (viz `https://pypi.org/project/openalpr/`).

Tento algoritmus poskytuje seznam kandidátů SPZ (viz kapitola 1.3.2.2), jehož délka je v tomto případě nastavená na deset. Proto se v případě hodnocení úspěšnosti rozpoznání celé SPZ hodnotí v jakém rozmezí tohoto seznamu byla správná SPZ nalezena, např. na 1. - 3. místě. Levenshteinova

vzdálenost rozpoznaného řetězce SPZ od skutečného se v tomto případě hodnotí jen pro řetězce na 1. místě tohoto seznamu.

Tabulka 4: Úspěšnost rozpoznávání textů SPZ pomocí OpenALPR a průměrný čas rozpoznání jedné SPZ

Testovací sada	Průměrná úspěšnost rozpoznání SPZ					Průměrný čas rozpoznání SPZ
	Celá SPZ				Levenshteinova vzdálenost (1. místo)	
	1. místo	1. - 3. místo	1. - 5. místo	1. - 10. místo		
A	38.24%	56.86%	59.80%	61.76%	89.33%	399.66 ms
B	62.98%	74.59%	77.35%	79.01%	94.81%	363.35 ms
C	50.77%	67.69%	70.77%	73.85%	90.98%	355.19 ms

Jak je vidět v tabulce 4, tak úspěšnost rozpoznání textů SPZ pomocí OpenALPR je o hodně vyšší, než v případě Tesseract OCR (viz tabulka 3). Průměrná hodnota úspěšnosti rozpoznání SPZ na 1. - 10. místě v seznamu kandidátů je 71.54 %. Důvodem vyšší úspěšnosti rozpoznání je, že tento algoritmus již disponuje částí, která je schopná najít SPZ v obraze. Tomu odpovídají i časy rozpoznání. Nicméně tyto výsledky mohou být ještě zlepšeny, pokud tomuto algoritmu bude předložena pouze oblast SPZ, která bude výstupem z natrénovaného detektoru objektů.

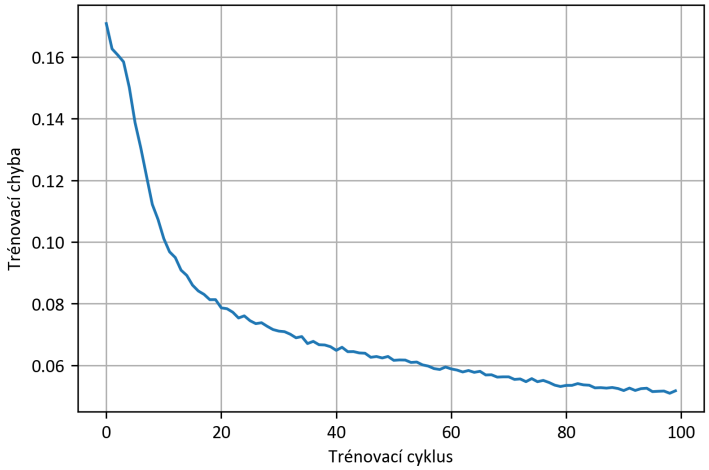
2.3 Trénování algoritmů pro detekci SPZ

Tato kapitola se zabývá trénováním několika modelů pro detekci objektů SPZ a jejich následným porovnáním. Modely jsou trénovány na trénovacích sadách A a B (viz tabulka 1). Jejich vyhodnocení a porovnání probíhá na datech z testovací sady D (viz tabulka 2).

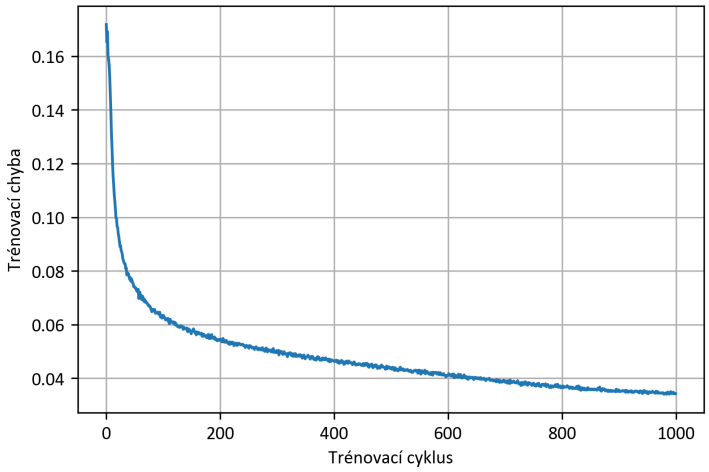
2.3.1 Trénování YOLO modelů

Na vlastních datech jsou natrénovány dva modely YOLOv5. Model YOLOv5 je nejnovější verzí modelu YOLO, který je popsán v kapitole 1.2.3.4. Oba modely jsou natrénovány v prostředí Google Colab použitím oficiálního skriptu připraveného pro účely trénování tohoto modelu (viz `https://colab.research.google.com/drive/1gDZ2xcTOgR39tGGs-EZ6i3RTs16wmzZQ`). Pro účely použití natrénovaných modelů je využita a upravena hotová následující implementace: `https://github.com/ultralytics/yolov5`.

První z modelů, nazvaný YOLOv5 1, je natrénován daty z trénovací sady A (viz tabulka 1) během 100 trénovacích cyklů. Druhý model, nazvaný YOLOv5 2, je natrénován pomocí dat z trénovací sady B (viz tabulka 1) během 1000 trénovacích cyklů. Průběhy trénovacích chyb v závislosti na trénovacích cyklech obou modelů jsou zobrazeny na obrázcích 21 a 22.



Obrázek 21: Průběh trénovací chyby modelu YOLOv5 1 v závislosti na trénovacích cyklech

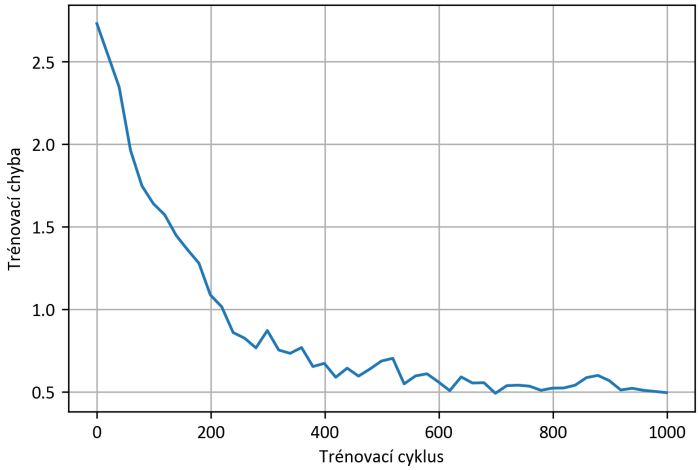


Obrázek 22: Průběh trénovací chyby modelu YOLOv5 2 v závislosti na trénovacích cyklech

2.3.2 Trénování modelu Faster R-CNN

Využitím knihovny Detectron2 (viz kapitola 1.2.3.6) je pomocí trénovací sady B (viz tabulka 1) dotrénován model Faster R-CNN X-101-FPN (viz `https://github.com/facebookresearch/detectron2/blob/master/MODEL_ZOO.md#coco-object-detection-baselines`). Toto dotrénování (tzv. fine-tuning) je provedeno během 1000 trénovacích cyklů.

Model je rovněž natrénován v prostředí Google Colab a je použit oficiální skript pro trénování modelů poskytovaných knihovnou Detectron2 (viz `https://colab.research.google.com/drive/1-TNOcPm3Jr3fOJG8rnGT9gh60mHUsvaW#scrollTo=kc8MmgZugZWR`). Graf průběhu trénovací chyby je zobrazen na obrázku 23. Pro účely použití modelu je využito stejných funkcí knihovny Detectron2 v prostředí Google Colab, jako při trénovaní modelu.



Obrázek 23: Průběh trénovací chyby modelu Faster R-CNN v závislosti na trénovacích cyklech

2.3.3 Porovnání natrénovaných modelů

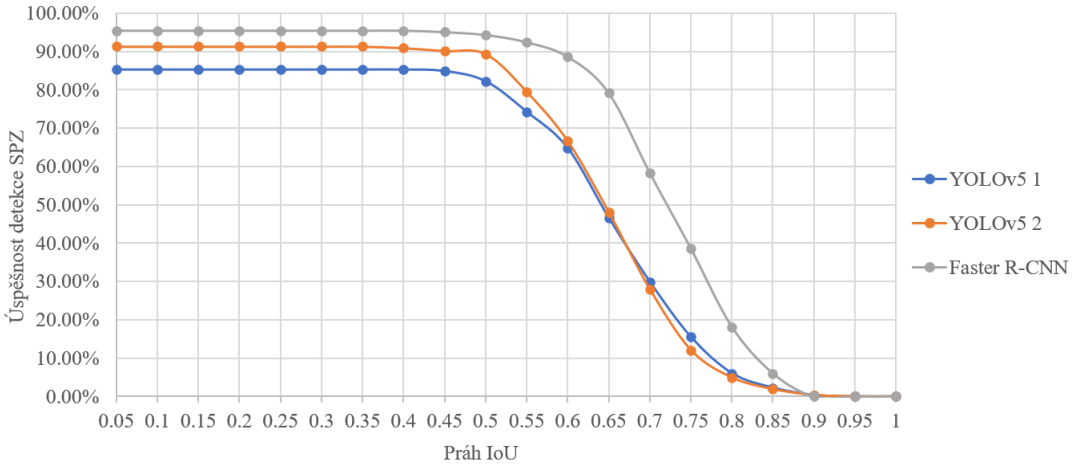
V tabulce 5 jsou zaznamenány výsledky vyhodnocení natrénovaných modelů na testovací sadě D (viz tabulka 2). Pro každý z modelů jsou vyhodnoceny všechny detekce objektů SPZ. Nejdřív je napočítán počet správně detekovaných SPZ (detekcí hodnocených jako TP) a počet nenalezených SPZ (nedetekovaných FN). Z těchto dvou hodnot je vypočítán Recall, který reprezentuje hodnotu úspěšnosti detekce SPZ. Dále je pro každý model napočítán počet falešných poplachů. To je případ, kdy je SPZ nalezena, tam kde není

a jedná se o detekce hodnocené jako FP. Všechny detekce jsou vyhodnoceny při různých hodnotách prahů IoU (viz kapitola 1.2.4).

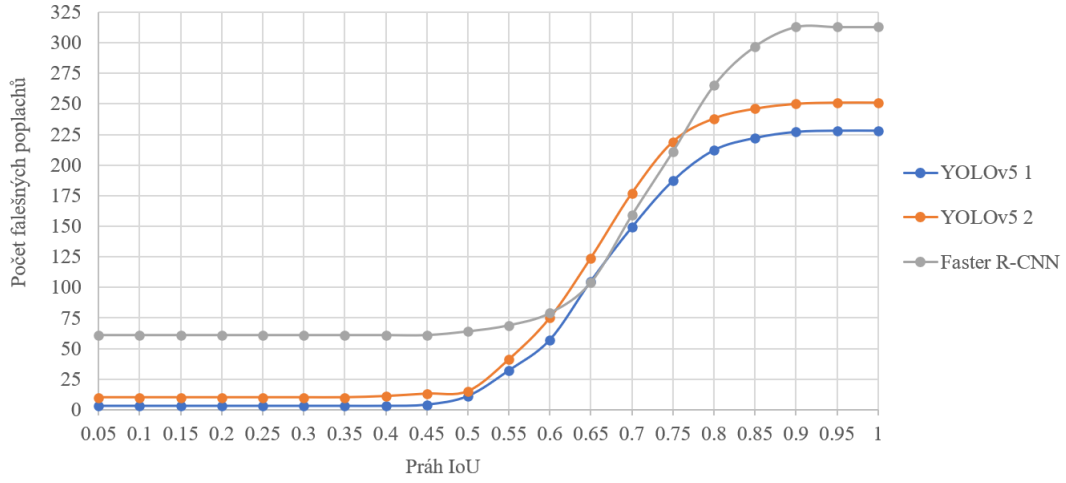
Tabulka 5: Porovnání natrénovaných modelů pro detekci objektů SPZ

Práh IoU	YOLOv5 1		YOLOv5 2		Faster R-CNN	
	Úspěšnost detekce SPZ	Počet falešných poplachů	Úspěšnost detekce SPZ	Počet falešných poplachů	Úspěšnost detekce SPZ	Počet falešných poplachů
0.05	85.23%	3	91.29%	10	95.45%	61
0.1	85.23%	3	91.29%	10	95.45%	61
0.15	85.23%	3	91.29%	10	95.45%	61
0.2	85.23%	3	91.29%	10	95.45%	61
0.25	85.23%	3	91.29%	10	95.45%	61
0.3	85.23%	3	91.29%	10	95.45%	61
0.35	85.23%	3	91.29%	10	95.45%	61
0.4	85.23%	3	90.91%	11	95.45%	61
0.45	84.85%	4	90.15%	13	95.08%	61
0.5	82.20%	11	89.39%	15	94.32%	64
0.55	74.24%	32	79.55%	41	92.42%	69
0.6	64.77%	57	66.67%	75	88.64%	79
0.65	46.59%	105	48.11%	124	79.17%	104
0.7	29.92%	149	28.03%	177	58.33%	159
0.75	15.53%	187	12.12%	219	38.64%	211
0.8	6.06%	212	4.92%	238	18.18%	265
0.85	2.27%	222	1.89%	246	6.06%	297
0.9	0.38%	227	0.38%	250	0.00%	313
0.95	0.00%	228	0.00%	251	0.00%	313
1	0.00%	228	0.00%	251	0.00%	313

Z grafu závislosti úspěšnosti detekce SPZ na různých hodnotách prahů IoU zobrazeného na obrázku 24 vyplývá, že z hlediska úspěšnosti detekce SPZ si nejlépe vede model Faster R-CNN, který má nejvyšší úspěšnost detekce při všech hodnotách prahů IoU. Nicméně co se týče počtu falešných poplachů, tak je na tom Faster R-CNN naopak nejhůř ze všech třech modelů. V tomto hodnocení vychází nejlépe model YOLOv5 1, který si tuto hodnotu drží nejnižší při všech hodnotách prahů IoU, jak je vidět na obrázku 25.



Obrázek 24: Úspěšnost detekce SPZ v závislosti na různých hodnotách prahu IoU všech třech modelů



Obrázek 25: Počet falešných poplachů při detekci SPZ v závislosti na různých hodnotách prahu IoU všech třech modelů

2.4 Různé přístupy rozpoznávání SPZ a jejich vyhodnocení

V této části jsou natrénované modely pro detekci objektů SPZ kombinovány s algoritmy pro rozpoznávání textů SPZ. Detektor SPZ najde v obraze oblast, kde se SPZ nachází a vyřízne ji. Tento výřez je pak předán algoritmu pro rozpoznávání textů. K tomuto výřezu samotné SPZ je ještě pro účely širšího testování úspěšnosti rozpoznání přidáváno okolí SPZ. Například okolí SPZ 50 % znamená, že když má vyříznutá SPZ rozměry 100 x 50 px, tak s okolím bude mít 200 x 100 px. Šířka výřezu je rozšířena na každou stranu o 50 % šířky SPZ. Obdobně je to s výškou výřezu. Vyhodnocení opět probíhá na testovacích sadách A, B a C (viz tabulka 2).

2.4.1 Kombinace YOLO a Tesseract OCR

Pro účely této i všech následující kombinací je z YOLOv5 modelů použit model YOLOv5 2, protože prokazuje lepší výsledky z hlediska úspěšnosti detekce SPZ (viz kapitola 2.3.3). V tabulce 6 je vidět úspěšnost rozpoznání textů SPZ pomocí kombinace YOLO a Tesseract OCR v závislosti na velikosti okolí vyříznutého společně s SPZ z původního obrazu. Vyhodnocovány jsou opět upravené a neupravené obrazy (viz kapitola 2.2.1) a rovněž dvěma způsoby vyhodnocení (viz kapitola 1.3.3). Tabulka 7 obsahuje průměrné časy rozpoznání jedné SPZ v závislosti na okolí.

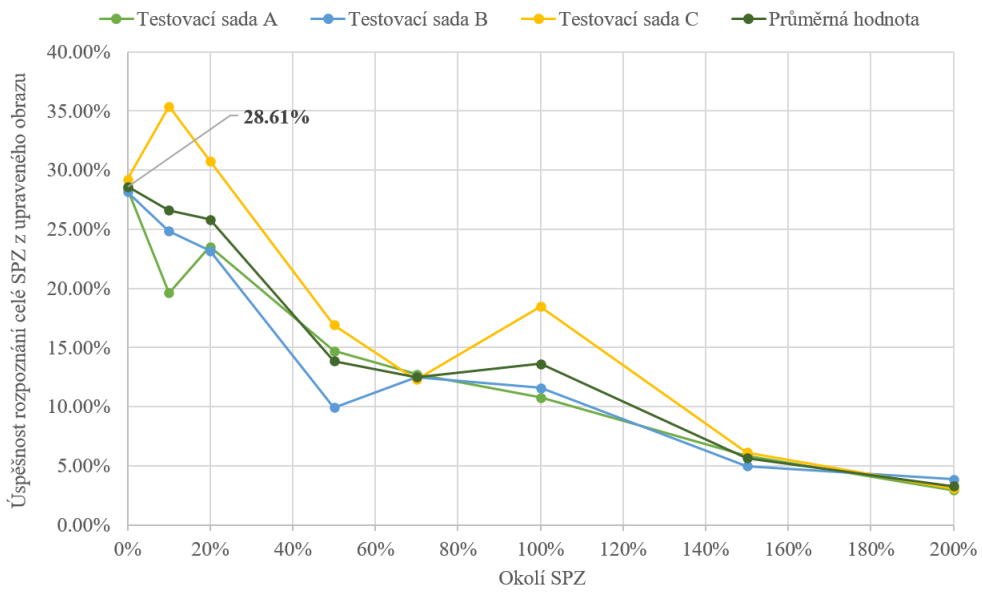
Na obrázku 26 je zobrazen graf úspěšnosti rozpoznání celé SPZ z upraveného obrazu pomocí YOLO a Tesseract OCR v závislosti na velikosti okolí SPZ. Jsou zde zobrazeny hodnoty úspěšností rozpoznání pro všechny testovací sady a také z nich vypočtená průměrná hodnota. Ta je nejvyšší při okolí SPZ 0 % a je to 28.61 %. To znamená cca o 26 % lepší výsledek, než ten, kterého je dosaženo použitím samotného Tesseract OCR.

Tabulka 6: Úspěšnost rozpoznání SPZ pomocí YOLO a Tesseract OCR

Testovací sada	Okolí SPZ	Neupravený obraz		Upravený obraz	
		Celá SPZ	Levenshteinova vzdálenost	Celá SPZ	Levenshteinova vzdálenost
A	0%	24.51%	47.61%	28.43%	53.18%
	10%	21.57%	42.39%	19.61%	44.89%
	20%	18.63%	40.68%	23.53%	45.91%
	50%	13.73%	36.70%	14.71%	38.18%
	70%	9.80%	28.30%	12.75%	37.16%
	100%	7.84%	29.55%	10.78%	29.32%
	150%	9.80%	29.32%	5.88%	27.61%
	200%	4.90%	22.61%	2.94%	22.50%
B	0%	23.20%	49.57%	28.18%	55.51%
	10%	20.44%	46.22%	24.86%	53.37%
	20%	16.57%	42.69%	23.20%	51.03%
	50%	14.36%	34.40%	9.94%	37.74%
	70%	13.26%	32.00%	12.50%	35.14%
	100%	8.29%	30.66%	11.60%	32.73%
	150%	5.52%	24.85%	4.97%	23.91%
	200%	3.87%	21.91%	3.87%	20.71%
C	0%	27.69%	48.61%	29.23%	51.04%
	10%	36.92%	50.00%	35.38%	53.30%
	20%	18.46%	40.10%	30.77%	50.17%
	50%	10.77%	32.64%	16.92%	40.80%
	70%	7.69%	25.17%	12.31%	30.21%
	100%	9.23%	31.60%	18.46%	37.50%
	150%	6.15%	27.43%	6.15%	24.48%
	200%	6.15%	23.26%	3.08%	21.88%

Tabulka 7: Průměrné časy rozpoznání SPZ pomocí YOLO a Tesseract OCR

Testovací sada	YOLO	Okolí SPZ	Neupravený obraz		Upravený obraz	
			Samostatně	Celkem (s YOLO)	Samostatně	Celkem (s YOLO)
A	273.00 ms	0%	197.06 ms	470.06 ms	232.19 ms	505.19 ms
		10%	244.26 ms	517.26 ms	282.08 ms	555.08 ms
		20%	251.79 ms	524.79 ms	296.94 ms	569.94 ms
		50%	321.23 ms	594.23 ms	429.80 ms	702.80 ms
		70%	383.82 ms	656.82 ms	513.00 ms	786.00 ms
		100%	445.26 ms	718.26 ms	623.86 ms	896.86 ms
		150%	602.45 ms	875.45 ms	896.52 ms	1169.52 ms
		200%	789.47 ms	1062.47 ms	1195.91 ms	1468.91 ms
B	268.60 ms	0%	193.00 ms	461.60 ms	225.41 ms	494.01 ms
		10%	203.78 ms	472.38 ms	257.64 ms	526.24 ms
		20%	224.86 ms	493.46 ms	282.32 ms	550.92 ms
		50%	284.81 ms	553.41 ms	392.45 ms	661.05 ms
		70%	344.93 ms	613.53 ms	521.72 ms	790.32 ms
		100%	443.92 ms	712.52 ms	696.69 ms	965.29 ms
		150%	647.14 ms	915.74 ms	972.47 ms	1241.07 ms
		200%	847.23 ms	1115.83 ms	1348.17 ms	1616.77 ms
C	305.12 ms	0%	228.60 ms	533.72 ms	276.22 ms	581.34 ms
		10%	257.87 ms	562.99 ms	328.93 ms	634.05 ms
		20%	279.49 ms	584.61 ms	364.10 ms	669.22 ms
		50%	383.59 ms	688.71 ms	515.43 ms	820.55 ms
		70%	444.36 ms	749.48 ms	599.23 ms	904.35 ms
		100%	638.72 ms	943.84 ms	834.51 ms	1139.63 ms
		150%	886.32 ms	1191.44 ms	1178.03 ms	1483.15 ms
		200%	1151.75 ms	1456.87 ms	1568.86 ms	1873.98 ms



Obrázek 26: Úspěšnost rozpoznání celé SPZ z upraveného obrazu pomocí YOLO a Tesseract OCR v závislosti na okolí SPZ s vyznačenou nejlepší průměrnou hodnotou

2.4.2 Kombinace YOLO a OpenALPR

Hodnoty úspěšnosti rozpoznání textů SPZ pomocí kombinace YOLO a OpenALPR při různých velikostech okolí SPZ jsou zobrazeny v tabulce 8. V případě úspěšnosti rozpoznání celé SPZ se opět hodnotí v jakém rozmezí seznamu kandidátů byla správná SPZ nalezena a v případě Levenshteinovy vzdálenosti rozpoznaného řetězce SPZ od skutečného se opět hodnotí SPZ na 1. místě tohoto seznamu (viz kapitola 2.2.2). Průměrné časy rozpoznání jedné SPZ v závislosti na okolí jsou v tabulce 9.

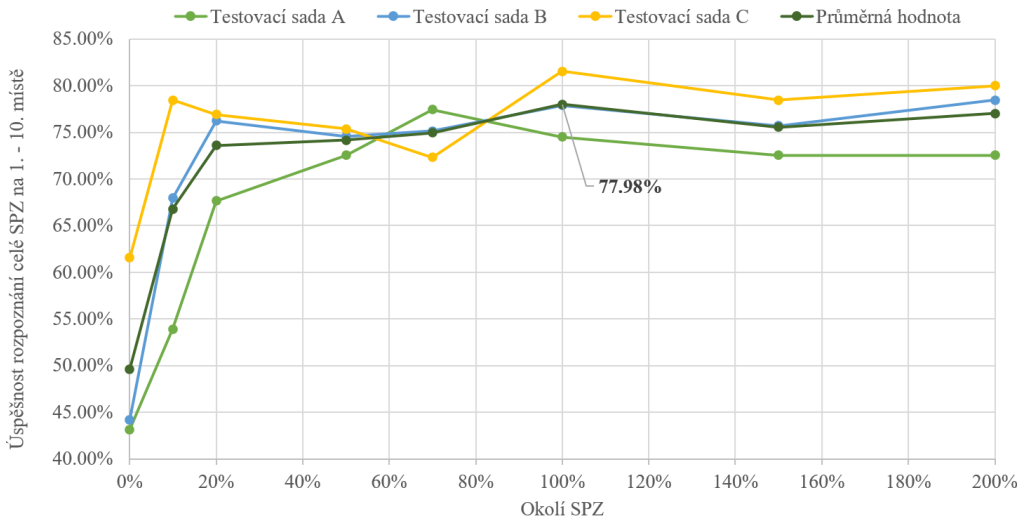
Graf na obrázku 27 demonstruje závislost úspěšnosti rozpoznání celé SPZ pomocí YOLO a OpenALPR na velikosti okolí. Nejlepšího průměrného výsledku 77.98 % je v tomto případě dosaženo s okolím SPZ 100 %. V porovnání s průměrnou hodnotou úspěšnosti rozpoznání pomocí samotného OpenALPR se jedná o cca 6 % zlepšení.

Tabulka 8: Úspěšnost rozpoznání SPZ pomocí YOLO a OpenALPR

Testovací sada	Okolí SPZ	Celá SPZ				Levenshteinova vzdálenost
		1. místo	1. - 3. místo	1. - 5. místo	1. - 10. místo	1. místo
A	0%	23.53%	36.27%	39.22%	43.14%	89.46%
	10%	39.22%	49.02%	52.94%	53.92%	90.94%
	20%	48.04%	62.75%	63.73%	67.65%	92.15%
	50%	50.98%	69.61%	71.57%	72.55%	92.53%
	70%	56.86%	73.53%	76.47%	77.45%	93.30%
	100%	49.02%	67.65%	72.55%	74.51%	91.52%
	150%	49.02%	64.71%	69.61%	72.55%	91.52%
	200%	49.02%	60.78%	68.63%	72.55%	90.51%
B	0%	32.04%	40.33%	43.09%	44.20%	90.38%
	10%	53.04%	64.64%	67.40%	67.96%	93.71%
	20%	59.12%	70.72%	75.14%	76.24%	94.31%
	50%	60.22%	70.72%	72.93%	74.59%	94.40%
	70%	59.12%	72.93%	74.03%	75.14%	94.12%
	100%	60.77%	70.72%	76.80%	77.90%	94.28%
	150%	60.77%	71.27%	74.03%	75.69%	94.23%
	200%	60.22%	73.48%	77.35%	78.45%	94.14%
C	0%	46.15%	55.38%	56.92%	61.54%	91.67%
	10%	50.77%	67.69%	75.38%	78.46%	91.07%
	20%	49.23%	69.23%	73.85%	76.92%	90.16%
	50%	55.38%	69.23%	72.31%	75.38%	91.60%
	70%	53.85%	66.15%	69.23%	72.31%	90.32%
	100%	56.92%	72.31%	80.00%	81.54%	90.82%
	150%	49.23%	64.62%	72.31%	78.46%	89.48%
	200%	56.92%	73.85%	75.38%	80.00%	91.67%

Tabulka 9: Průměrné časy rozpoznání SPZ pomocí YOLO a OpenALPR

Testovací sada	YOLO	Okolí SPZ	Samostatně	Celkem (s YOLO)
A	273.00 ms	0%	209.62 ms	482.62 ms
		10%	218.62 ms	491.62 ms
		20%	235.46 ms	508.46 ms
		50%	234.34 ms	507.34 ms
		70%	249.83 ms	522.83 ms
		100%	250.16 ms	523.16 ms
		150%	274.02 ms	547.02 ms
		200%	299.02 ms	572.02 ms
B	268.60 ms	0%	203.18 ms	471.78 ms
		10%	209.58 ms	478.18 ms
		20%	242.91 ms	511.51 ms
		50%	229.56 ms	498.16 ms
		70%	227.99 ms	496.59 ms
		100%	244.37 ms	512.97 ms
		150%	271.31 ms	539.91 ms
		200%	308.66 ms	577.26 ms
C	305.12 ms	0%	233.44 ms	538.56 ms
		10%	239.51 ms	544.63 ms
		20%	242.82 ms	547.94 ms
		50%	262.81 ms	567.93 ms
		70%	266.36 ms	571.48 ms
		100%	284.33 ms	589.45 ms
		150%	333.89 ms	639.01 ms
		200%	348.22 ms	653.34 ms



Obrázek 27: Úspěšnost rozpoznání celé SPZ na 1. - 10. místě v seznamu kandidátů pomocí YOLO a OpenALPR v závislosti na okolí SPZ s vyznačenou nejlepší průměrnou hodnotou

2.4.3 Kombinace Faster R-CNN a Tesseract OCR

V tabulce 10 jsou zobrazeny hodnoty úspěšností rozpoznání textů SPZ pomocí kombinace Faster R-CNN a Tesseract OCR při různých hodnotách vyříznutého okolí SPZ. Vyhodnocení probíhá stejným způsobem, jako při použití kombinace YOLO a Tesseract OCR (viz kapitola 2.4.1). V tabulce 11 jsou průměrné časy rozpoznání jedné SPZ v závisloti na okolí.

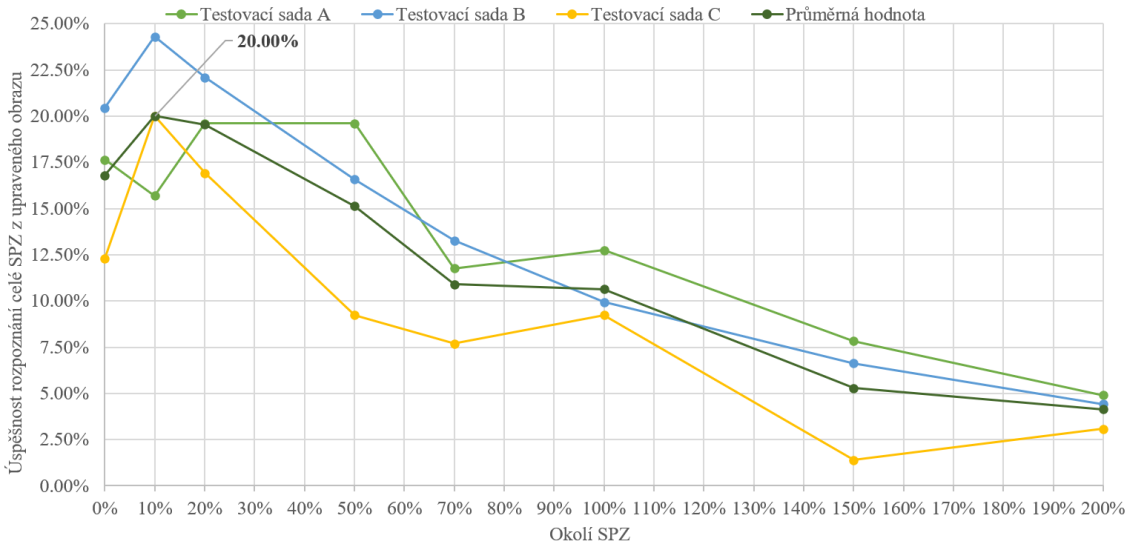
Graf na obrázku 28 zobrazuje úspěšnosti rozpoznání celé SPZ z upraveného obrazu pomocí této kombinace. Nejvyšší průměrná hodnota úspěšnosti rozpoznání je při okolí SPZ 10 % a je to 20 %. Tento výsledek znamená pouze o cca 18 % lepší výsledek, než v případě použití samotného Tesseract OCR, takže kombinace YOLO a Tesseract OCR vychází v porovnání s touto kombinací lépe.

Tabulka 10: Úspěšnost rozpoznání SPZ pomocí Faster R-CNN a Tesseract OCR

Testovací sada	Okolí SPZ	Neupravený obraz		Upravený obraz	
		Celá SPZ	Levenshteinova vzdálenost	Celá SPZ	Levenshteinova vzdálenost
A	0%	18.63%	39.53%	17.65%	41.76%
	10%	15.69%	38.76%	15.69%	37.60%
	20%	16.67%	36.24%	19.61%	36.82%
	50%	14.71%	33.82%	19.61%	34.69%
	70%	5.88%	27.23%	11.76%	30.81%
	100%	5.88%	26.94%	12.75%	32.46%
	150%	3.92%	21.51%	7.84%	26.55%
	200%	3.92%	23.06%	4.90%	24.81%
B	0%	24.31%	41.68%	20.44%	42.20%
	10%	16.02%	37.51%	24.31%	41.20%
	20%	14.92%	32.68%	22.10%	37.99%
	50%	12.71%	30.05%	16.57%	32.53%
	70%	10.50%	26.55%	13.26%	30.29%
	100%	9.39%	24.93%	9.94%	26.23%
	150%	4.42%	20.96%	6.63%	24.12%
	200%	4.42%	20.62%	4.42%	20.33%
C	0%	23.08%	36.28%	12.31%	34.38%
	10%	15.38%	32.47%	20.00%	37.33%
	20%	10.77%	30.21%	16.92%	35.94%
	50%	7.69%	27.43%	9.23%	28.82%
	70%	6.15%	22.05%	7.69%	24.65%
	100%	4.62%	26.74%	9.23%	30.56%
	150%	4.62%	21.70%	1.40%	20.49%
	200%	0.00%	17.01%	3.08%	20.66%

Tabulka 11: Průměrné časy rozpoznání SPZ pomocí Faster R-CNN a Tesseract OCR

Testovací sada	Faster R-CNN	Okolí SPZ	Neupravený obraz		Upravený obraz	
			Samostatně	Celkem (s Faster R-CNN)	Samostatně	Celkem (s Faster R-CNN)
A	261.03 ms	0%	178.92 ms	439.95 ms	196.24 ms	457.27 ms
		10%	201.47 ms	462.50 ms	213.72 ms	474.75 ms
		20%	238.23 ms	499.26 ms	230.39 ms	491.42 ms
		50%	293.95 ms	554.98 ms	283.58 ms	544.61 ms
		70%	326.58 ms	587.61 ms	344.20 ms	605.23 ms
		100%	411.92 ms	672.95 ms	429.56 ms	690.59 ms
		150%	544.12 ms	805.15 ms	668.14 ms	929.17 ms
		200%	723.23 ms	984.26 ms	1096.73 ms	1357.76 ms
B	237.66 ms	0%	225.51 ms	463.17 ms	228.64 ms	466.30 ms
		10%	231.67 ms	469.33 ms	245.30 ms	482.96 ms
		20%	225.83 ms	463.49 ms	267.86 ms	505.52 ms
		50%	275.06 ms	512.72 ms	357.38 ms	595.04 ms
		70%	339.50 ms	577.16 ms	432.51 ms	670.17 ms
		100%	420.35 ms	658.01 ms	572.74 ms	810.40 ms
		150%	603.04 ms	840.70 ms	883.00 ms	1120.66 ms
		200%	818.52 ms	1056.18 ms	1231.35 ms	1469.01 ms
C	237.05 ms	0%	205.15 ms	442.20 ms	196.92 ms	433.97 ms
		10%	236.79 ms	473.84 ms	214.16 ms	451.21 ms
		20%	233.35 ms	470.40 ms	247.18 ms	484.23 ms
		50%	271.52 ms	508.57 ms	333.59 ms	570.64 ms
		70%	342.84 ms	579.89 ms	332.57 ms	569.62 ms
		100%	454.96 ms	692.01 ms	490.00 ms	727.05 ms
		150%	668.97 ms	906.02 ms	710.77 ms	947.82 ms
		200%	764.11 ms	1001.16 ms	968.83 ms	1205.88 ms



Obrázek 28: Úspěšnost rozpoznání celé SPZ z upraveného obrazu pomocí Faster R-CNN a Tesseract OCR v závislosti na okolí SPZ s vyznačenou nejlepší průměrnou hodnotou

2.4.4 Kombinace Faster R-CNN a OpenALPR

Poslední kombinací algoritmů použitou pro účely rozpoznávání textů SPZ je kombinace Faster R-CNN a OpenALPR. V tabulce 12 jsou zobrazeny hodnoty úspěšností rozpoznání v závislosti na okolí SPZ a v tabulce 13 pak průměrné časy rozpoznání jedné SPZ pomocí této kombinace. Způsob vyhodnocení úspěšnosti rozpoznání je stejný jako v případě kombinace YOLO a OpenALPR (viz kapitola 2.4.2).

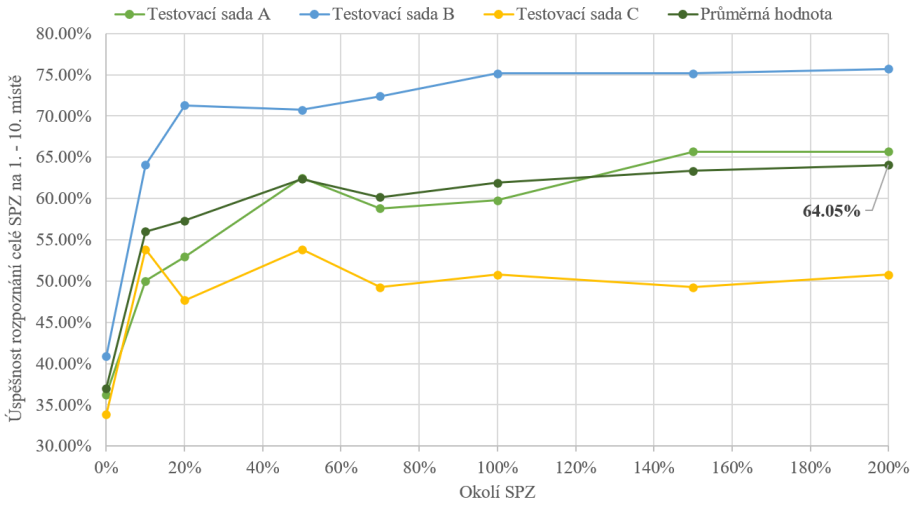
Na obrázku 29 je zobrazen průběh úspěšnosti rozpoznání SPZ v závislosti na okolí SPZ. Nejvyšší průměrnou hodnotou je v případě této kombinace 64.05 %, což nepřekračuje nejvyšší průměrnou hodnotu úspěšnosti rozpoznání kombinace YOLO a OpenALPR, která si tedy vede v porovnání s touto kombinací lépe.

Tabulka 12: Úspěšnost rozpoznání SPZ pomocí Faster R-CNN a OpenALPR

Testovací sada	Okolí SPZ	Celá SPZ				Levenshteinova vzdálenost
		1. místo	1. - 3. místo	1. - 5. místo	1. - 10. místo	1. místo
A	0%	23.53%	33.33%	35.29%	36.27%	90.99%
	10%	30.39%	45.10%	46.08%	50.00%	90.28%
	20%	39.22%	46.08%	50.00%	52.94%	92.38%
	50%	43.14%	54.90%	59.80%	62.50%	92.25%
	70%	34.31%	51.96%	54.90%	58.82%	89.38%
	100%	41.18%	57.84%	57.84%	59.80%	90.58%
	150%	36.27%	55.88%	61.76%	65.69%	91.05%
	200%	45.10%	58.82%	62.75%	65.69%	90.54%
B	0%	32.04%	38.67%	39.78%	40.88%	94.39%
	10%	49.72%	61.33%	61.88%	64.09%	92.78%
	20%	56.35%	65.75%	68.51%	71.27%	93.93%
	50%	56.35%	66.30%	67.96%	70.72%	93.76%
	70%	57.46%	67.96%	70.17%	72.38%	93.93%
	100%	59.12%	69.61%	72.38%	75.14%	94.17%
	150%	61.33%	71.82%	72.93%	75.14%	93.92%
	200%	62.98%	70.72%	73.48%	75.69%	93.34%
C	0%	18.46%	23.08%	26.15%	33.85%	87.07%
	10%	32.31%	46.15%	50.77%	53.85%	89.38%
	20%	35.38%	43.08%	46.15%	47.69%	90.77%
	50%	40.00%	49.23%	50.77%	53.85%	93.02%
	70%	29.23%	40.00%	46.15%	49.23%	90.24%
	100%	30.77%	43.08%	46.15%	50.77%	89.58%
	150%	29.23%	41.54%	46.15%	49.23%	88.72%
	200%	33.85%	47.69%	49.23%	50.77%	90.12%

Tabulka 13: Průměrné časy rozpoznání SPZ pomocí Faster R-CNN a OpenALPR

Testovací sada	Faster R-CNN	Okolí SPZ	Samostatně	Celkem (s Faster R-CNN)
A	261.03 ms	0%	235.95 ms	496.98 ms
		10%	245.69 ms	506.72 ms
		20%	246.08 ms	507.11 ms
		50%	268.40 ms	529.43 ms
		70%	263.40 ms	524.43 ms
		100%	277.16 ms	538.19 ms
		150%	300.98 ms	562.01 ms
		200%	320.59 ms	581.62 ms
B	237.66 ms	0%	269.34 ms	507.00 ms
		10%	293.10 ms	530.76 ms
		20%	280.44 ms	518.10 ms
		50%	292.87 ms	530.53 ms
		70%	295.67 ms	533.33 ms
		100%	313.17 ms	550.83 ms
		150%	333.33 ms	570.99 ms
		200%	374.45 ms	612.11 ms
C	237.05 ms	0%	253.08 ms	490.13 ms
		10%	240.84 ms	477.89 ms
		20%	229.38 ms	466.43 ms
		50%	246.98 ms	484.03 ms
		70%	240.76 ms	477.81 ms
		100%	255.79 ms	492.84 ms
		150%	275.79 ms	512.84 ms
		200%	315.36 ms	552.41 ms



Obrázek 29: Úspěšnost rozpoznání celé SPZ na 1. - 10. místě v seznamu kandidátů pomocí Faster R-CNN a OpenALPR v závislosti na okolí SPZ s vyznačenou nejlepší průměrnou hodnotou

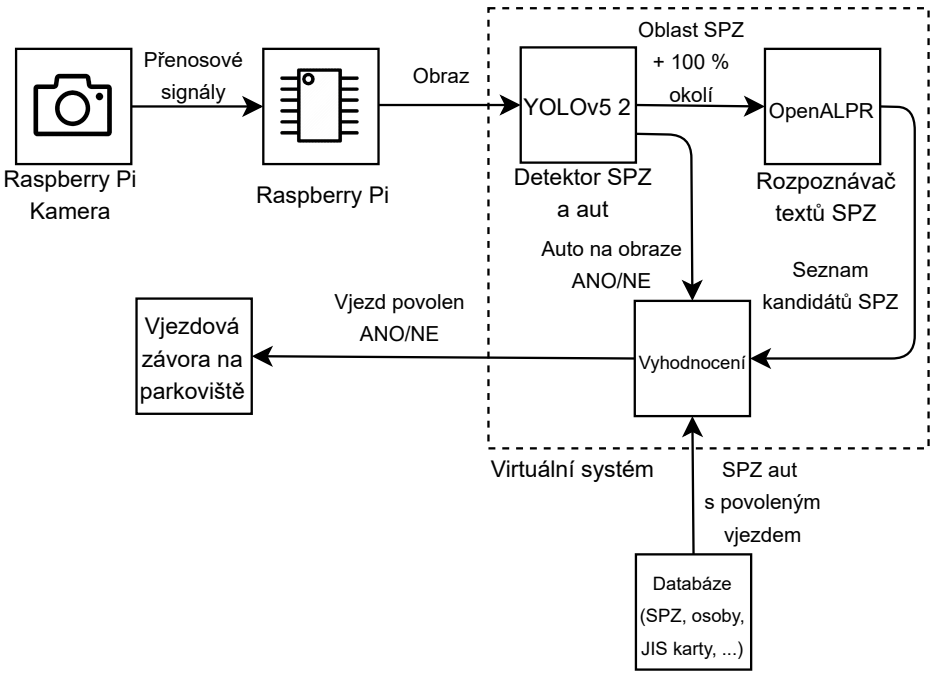
2.5 Diskuze výsledků a jejich možné využití v praxi

Na základě výsledků jednotlivých kombinací algoritmů pro účely rozpoznávání textů SPZ prezentovaných v kapitole 2.4 se jako nejvhodnější volba pro účely detekce a rozpoznání SPZ jeví použití kombinace YOLO a OpenALPR, kde bylo dosaženo nejvyšší průměrné hodnoty úspěšnosti rozpoznání 77.98 %. Všechny kombinace algoritmů byly vyzkoušeny na datech z testovací sady A, B a C (viz tabulka 2), které na jednotlivých obrazech obsahují SPZ naklopené pod různými úhly. Z toho důvodu je hodnocení celkem přísné. V praxi budou SPZ snímány stále na stejném místě pod vodorovným úhlem, takže jsou očekávány lepší výsledky.

Úloha detekce a rozpoznávání SPZ v praxi je demonstrována v tabulce nacházející se v příloze. Je vybráno deset obrazů z volně dostupné testovací sady z projektu detekce a rozpoznávání SPZ výzkumného týmu záhřebské univerzity (viz `http://www.zemris.fer.hr/projects/LicensePlates/english/`). V každém obraze je nejprve pomocí modelu YOLOv5 2 nalezena oblast SPZ se 100 % okolím. Z této oblasti je poté pomocí OpenALPR nalezeno vždy pět kandidátů na tuto SPZ. V osmi z deseti případů je správná SPZ nalezena na 1. místě v seznamu kandidátů a v tabulce je vždy zvýrazněna tučně. Pro každý obraz je zde také uveden čas potřebný pro detekci

oblasti SPZ pomocí YOLOv5 2, rozpoznání textu SPZ pomocí OpenALPR a celkový čas zpracování jednoho obrazu.

Na obrázku 30 je zobrazeno blokové schéma jednoduchého parkovacího systému s možným využitím například v univerzitních podzemních garážích. Jedná se o příklad uplatnění zjištěných výsledků a na jejich základě vybrané nejlepší kombinace algoritmů. Ty jsou za účelem urychlení procesu v tomto příkladě implementovány ve virtuálním systému provozovaném na výkonnějším hardwaru, než je mikropočítač Raspberry Pi. Ten je v příkladě použit jen pro účely komunikace s kamerou snímající obraz přijíždějícího auta a pro zaslání tohoto obrazu do virtuálního systému. Tam je provedena detekce oblasti SPZ a auta v obraze a následné rozpoznání textu SPZ v podobě možných kandidátů. Ty jsou poté porovnávány s SPZ aut, které mají dle databáze povolený vjezd na dané parkoviště. Pokud je nalezena shoda s jedním z kandidátů na SPZ a v obraze je kromě SPZ detekováno také auto, tak systém otevře vjezdovou závoru na parkoviště.



Obrázek 30: Blokové schéma parkovacího systému - příklad praktického využití YOLO a OpenALPR pro účely detekce a rozpoznávání SPZ

3 Závěr

Cílem této bakalářské práce bylo nastudovat problematiku počítačového vidění pro účely detekce a rozpoznávání textů SPZ, provést rešerši problematiky rozpoznávání SPZ a implementovat dostupný model. Následně navrhnout model pro detekci objektů SPZ, natrénovat ho a otestovat na vlastních datech. V teoretické části byly krátce představeny již dostupné fungující systémy pro detekci a rozpoznávání SPZ. Následoval úvod do dvou, pro tuto problematiku stěžejních, odvětví počítačového vidění. Prvním z nich byla detekce objektů a druhým z nich optické rozpoznávání znaků. Kromě základních principů každého odvětví zde bylo popsáno i několik algoritmů pro řešení této problematiky.

V úvodu praktické části byl popsán proces vytváření vlastních trénovacích a testovacích datasetů. Následovala implementace dvou algoritmů pro optické rozpoznávání znaků. Prvním z nich byl pro tuto problematiku obecný algoritmus Teseract OCR a druhým specializovaný algoritmus pro rozpoznávání textů SPZ OpenALPR. Oba z nich byly otestovány na vytvořených testovacích sadách a lepších výsledků dosáhl algoritmus OpenALPR. Pro zlepšení výsledků těchto algoritmů byly pomocí vytvořených trénovacích datasetů natrénovány tři modely pro detekci oblasti SPZ v obraze, konkrétně dva modely YOLOv5 a jeden model Faster R-CNN. Schopnost těchto modelů detekovat oblast SPZ byla vyhodnocena na jedné z testovacích sad. Nejlepší dva modely z hlediska úspěšnosti detekce oblasti SPZ, jeden z modelů YOLOv5 a model Faster R-CNN, byly poté použity v kombinaci s oběma z algoritmů pro rozpoznávání textů SPZ.

Nejlepšího výsledku, průměrné úspěšnosti rozpoznání SPZ 77.98 %, dosáhla kombinace natrénovaného detektoru objektů SPZ YOLOv5 s knihovnou pro rozpoznávání textů SPZ OpenALPR. V praxi by však úspěšnost měla být vyšší, neboť zde budou SPZ snímány pod vodorovným úhlem. Tato situace je demonstrována v tabulce, která se nachází v příloze. Na úplném závěru praktické části je popsán příklad možného využití výsledků této bakalářské práce pro účely jednoduchého parkovacího systému.

Výsledek průměrné úspěšnosti rozpoznání by mohl být zlepšen, pokud by byla zvýšena úspěšnost detekce oblasti SPZ. Toho by se dosáhlo větším množstvím trénovacích dat obsahujícím zejména více SPZ pod různými úhly. Úspěšnost detekce textů SPZ by mohla být zvýšena přidáním algoritmu pro vyrovnání naklopené SPZ do vodorovné polohy.

Literatura

[1] *Object Detection Guide*. Fritz AI, 2021. https://www.fritz.ai/object-detection/.

[2] *Carmen® ANPR software*. Adaptive Recognition, 2021. https://adaptiverecognition.com/products/carmen-anpr-software/.

[3] *Detectron2*. Facebook Research, 2021. https://github.com/facebookresearch/detectron2.

[4] *Automatické rozpoznání státních poznávacích značek GPP LPR*. Green Center, 2021. https://www.green.cz/kamerovy-system-11.

[5] *Precision and recall*. Wikipedia, 2021. https://en.wikipedia.org/wiki/Precision_and_recall.

[6] *Levenshtein distance*. Wikipedia, 2021. https://en.wikipedia.org/wiki/Levenshtein_distance.

[7] Raluca Marina Sferle a Elisa Valentina Moisi. *Automatic Number Plate Recognition for a Smart Service Auto*. University of Oradea, 2019.

[8] Adriano Mendes Gil, Cícero Ferreira Fernandes Costa Filho a Marly Guimarães Fernandes Costa. *Handwritten Digit Recognition Using SVM Binary Classifiers and Unbalanced Decision Trees*. Instituto Nokia de Tecnologia, Manaus, 2014.

[9] Ahmed Fawzy Gad. *An Introduction to Optical Character Recognition for Beginners*. Towards Data Science, 2020. https://towardsdatascience.com/an-introduction-to-optical-character-recognition-for-beginners-14268c99d60.

[10] Ahmed Fawzy Gad. *Evaluating Object Detection Models Using Mean Average Precision (mAP)*. PaperspaceBlog, 2020. https://blog.paperspace.com/mean-average-precision.

[11] Ross Girshick. *Fast R-CNN*. In Proc. IEEE Intl. Conf. on computer visionia, s. 1440–1448, 2015.

[12] Matthew Hill. *OpenALPR Design*. GitHub, 2014. https://github.com/openalpr/openalpr/wiki/OpenALPR-Design.

[13] I. M. D. R. Mudiarta, I. M. D. S. Atmaja, I. K. Suharsana, I. W. G. S. Antara, I. W. P. Bharaditya, G. A. Suandirat a G. Indrawan. *Balinese character recognition on mobile application based on tesseract open source OCR engine*. Dept. of Computer Science Universitas Pendidikan Ganesha Singaraja, 2020.

[14] Joseph Redmon, Santosh Divvalay, Ross Girshick a Ali Farhadiy. *You Only Look Once: Unified, Real-Time Object Detection*. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[15] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn a Andrew Zisserman. *The PASCAL Visual Object Classes (VOC) Challenge*. University of California, s. 1–3, 2009.

[16] N. O' Mahony, S. Campbell, A. Carvalho, S. Harapanahalli, G. V. Hernandez, L. Krpalkova, D. Riordan a J. Walsh. *Deep Learning vs. Traditional Computer Vision*. IMaR Technology Gateway, Institute of Technology Tralee, s. 4, 2019.

[17] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov a Sergey Zagoruyko. *End-to-End Object Detection with Transformers*. 2020.

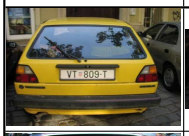
[18] Ross Girshick, Jeff Donahue a Trevor Darrell Jitendra Malik. *Rich feature hierarchies for accurate object detection and semantic segmentation*. In Proc. IEEE Conf. on computer vision and pattern recognition (CVPR), s. 580–587, 2014.

[19] Shaoqing Ren, Kaiming He, Ross Girshick a Jian Sun. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. In Advances in neural information processing systems, s. 91–99, 2015.

[20] Venkatesh Wadawadagi. *Metrics to Use to Evaluate Deep Learning Object Detectors*. KDnuggets, 2020. `https://www.kdnuggets.com/2020/08/metrics-evaluate-deep-learning-object-detectors.html`.

Příloha

Následující tabulka demonstruje detekci a rozpoznávání SPZ pomocí YOLOv5 a OpenALPR v praxi, kde jsou SPZ snímány pod vodorovným úhlem.

Obraz	YOLOv5 2		OpenALPR		Celkový čas detekce a rozpoznání SPZ
	Detekovaná oblast SPZ se 100 % okolím	Čas detekce oblasti SPZ	Seznam kandidátů s hodnotami důvěryhodností	Čas rozpoznání textu SPZ	
		670 ms	**1. VT809T na 88.54 %** 2. VT8O9T na 85.42 % 3. VT8Q9T na 81.54 % 4. VT8D9T na 81.32 % 5. V809T na 79.26 %	119.54 ms	789.54 ms
		770 ms	1. KI3GC na 88.01 % **2. KI43GC na 83.53 %** 3. KI8GC na 83.43 % 4. KIBGC na 82.89 % 5. KISGC na 81.46 %	41.27 ms	811.27 ms
		710 ms	**1. 325J521 na 92.91 %** 2. 3Z5J521 na 84.74 % 3. 32SJ521 na 83.14 % 4. 325J5Z1 na 82.68 % 5. 325JS21 na 82.63 %	29.27 ms	739.27 ms
		800 ms	**1. ZG8297I na 92.95 %** 2. ZG82S7I na 85.68 % 3. 2G8297I na 83.68 % 4. Z68297I na 83.28 % 5. ZG82B7I na 83.01 %	32.91 ms	832.91 ms
		790 ms	1. VZ09O9MD na 93.83 % 2. VZO9O9MD na 91.35 % **3. VZ0909MD na 91.31 %** 4. VZO909MD na 88.83 % 5. VZD9O9MD na 88.71 %	42.30 ms	832.30 ms
		840 ms	**1. HGAS1802 na 92.59 %** 2. HGAS18D2 na 86.52 % 3. HGAS18O2 na 86.34 % 4. HGAS18Q2 na 85.8 % 5. HGAS18G2 na 85.38 %	34.91 ms	874.91 ms
		820 ms	**1. ZG4513R na 89.8 %** 2. ZG513R na 82.45 % 3. ZG451BR na 81.66 % 4. ZG451SR na 80.49 % 5. ZG451R na 79.76 %	33.27 ms	853.27 ms
		730 ms	**1. ZG4269AC na 89.96 %** 2. ZG269AC na 84.45 % 3. ZG42G9AC na 83.2 % 4. ZG42B9AC na 82.77 % 5. ZG42S9AC na 82.04 %	35.65 ms	765.65 ms
		830 ms	**1. EDAL551 na 92.06 %** 2. JEDAL551 na 89.41 % 3. 8EDAL551 na 88.8 % 4. E0AL551 na 83.83 % 5. EOAL551 na 83.78 %	38.84 ms	868.84 ms
		780 ms	**1. BV593KX na 88.83 %** 2. HBV593KX na 88.62 % 3. BV59SKX na 83.79 % 4. BV59KX na 83.76 % 5. HBV59SKX na 83.58 %	39.72 ms	819.72 ms