Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

BAKALÁŘSKÁ PRÁCE

PLZEŇ, 2020

Jakub Straka

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(a) samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí

V Plzni dne 31. května 2020 .....

Anotace

Tato práce se zabývá detekcí objektů v obraze pomocí metod založených na neuronových sítích. V první části jsou popsány datasety, které jsou běžně používané při detekci objektů. Dále jsou představeny některé metody pro řešení tohoto problému a jejich porovnání. V další části jsou některé z představených metod natrénovány na specifickém datasetu. V poslední části jsou porovnány výsledky natrénovaných modelů.

Klíčová slova

Konvoluční neuronové sítě, Detekce objektů, Počítačové vidění, SSD, Mask R-CNN

Annotation

This thesis is focused on object detection in an image using neural network. In the first part are described common datasets used in object detection task. Object detection methods are described and compared in the second part. In third part some of the methods are trained on a specific dataset. Last part compares obtained results.

Keywords

Convolutional neural network, Object detection, Computer vision, SSD, Mask R-CNN

Obsah

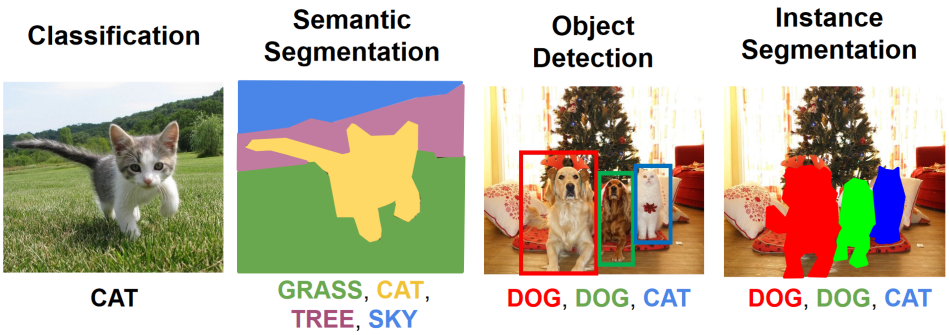
Seznam obrázků

Seznam tabulek

1 Úvod

Od narození vnímáme okolní svět zrakem a učíme se rozpoznávat různé objekty. Časem nám nedělá problém od sebe objekty rozlišit. Dokonce dokážeme poznat objekty jedné instance i přesto, že jsme neviděli všechny, které existují. Například stůl, může mít různý tvar, být různě vysoký, mít různý počet nohou. Pokud jsme v životě nějaký stůl viděli, nejspíše bychom dokázali poznat stůl, který jsme ještě nikdy neviděli. To nicméně neplatí pro počítače. Určit co se na obrázku nachází, případně kde, je pro počítače velmi výpočetně náročné. Až se zvýšením výpočetního výkonu a vývojem hlubokých neuronových sítí bylo možné tyto úlohy řešit s větší efektivitou.

Ještě před neuronovými sítěmi, se používali metody nyní nazývané tradiční metody detekce. V roce 2001 dosáhl P. Viola a M. Jones [19] jako první detekce obličejů v reálném čase. Příkladem další tradiční metody je HOG deskriptor, který publikoval N.Dalal a B. Triggs [3] v roce 2005. Pokrok tradičních metod detekce během let zpomaloval a svého vrcholu dosáhl v letech 2010 - 2012. V roce 2012 Alex Krizhevsky, Ilya Sutskever a Geoffrey E. Hinton publikovali model založený na konvoluční neuronové síti [10], který dosahoval lepších výsledků, než dosavadní metody. Architektura sítě byla nazvána AlexNet a obsahuje 8 vrstev. Od té doby došlo k velkému rozvoji v oblasti strojového vidění. Drtivá většina metod je založená na konvolučních neuronových sítích s velkým počtem vrstev.



Obrázek 1: Ukázka některých úloh počítačového vidění [12]

Cílem strojového vidění je získat informaci o objektech které se nacházejí v obrazu nebo videu, tak jak to dokážou lidé. Pro všechny úlohy, o kterých budeme dále mluvit předpokládáme, že vstupem je digitální obraz a množina tříd, které mohou být přiřazeny objektům v obrazu.

První úlohou strojového vidění, která může být považována za základní, je klasifikace. Při klasifikace chceme obrázek označit jednou z možných tříd. Neurčujeme žádnou další

informaci o objektu v obraze a předpokládáme, že se v obrazu nachází pouze jeden objekt jedné třídy. Příkladem klasifikace je dataset MNIST [11] pro klasifikaci ručně psaných čísel.

Další úlohou je segmentace. Při segmentaci chceme označit co nejpřesněji každý pixel, který danému objektu náleží. Tato úloha může být ještě rozdělena na segmentaci instancí a sémantickou segmentaci. Při segmentaci instancí, chceme označit každý objekt zvlášť, i v případě, že náleží stejné třídě. Při sémantické segmentaci budou objekty náležící jedné třídě označeny stejně. Zpravidla také označujeme všechny pixely v obrazu, i pozadí.

Možná nejvíce důležitou úlohou, které je v poslední době věnována velká pozornost je detekce objektů. Cílem úlohy je najít všechny výskyty všech objektů z množiny tříd a označit je rámečkem (bounding box). Detekce objektů nachází využití například v autonomním řízení aut, medicíně, průmyslu a dalších. Všechny z představených úloh jsou ilustrovány v obrázku 1.

Detekce objektů je užitečná v širokém rozsahu oborů. V této práci se budeme zabývat některými metodami detekce objektů založených na neuronových sítích a jejich porovnáním. Detekci budeme provádět na datasetu CoralClef 2020, který je publikován v rámci druhého ročníku soutěže ImageCLEF. Soutěž je zaměřena na detekci korálů.

2 Data pro trénování neuronové sítě

2.1 Datasety

Pro trénování jakékoliv neuronové sítě jsou potřeba data a obvykle platí čím více tím lépe. Během let vzniklo velké množství datasetů, které jsou obvykle rozděleny do trénovací, validační a testovací množiny.

Mezi nejznámější a nejpoužívanější datasety v oboru detekce objektů patří: Pascal VOC [4] (Visual Object Classes) 2005-2012, ILSVRC [17] (Large Scale Visual Recognition Challenge) 2010-2017, Microsoft COCO [13] (Common Objects in Context). Datasety se liší v počtu tříd, počtu obrázků a jejich velikosti, ale i úlohách, které je možné s nimi řešit. Kromě dat pro detekci objektů, některé obsahují i data pro segmentaci, nebo detekci pózy osob.

2.1.1 ILSVRC

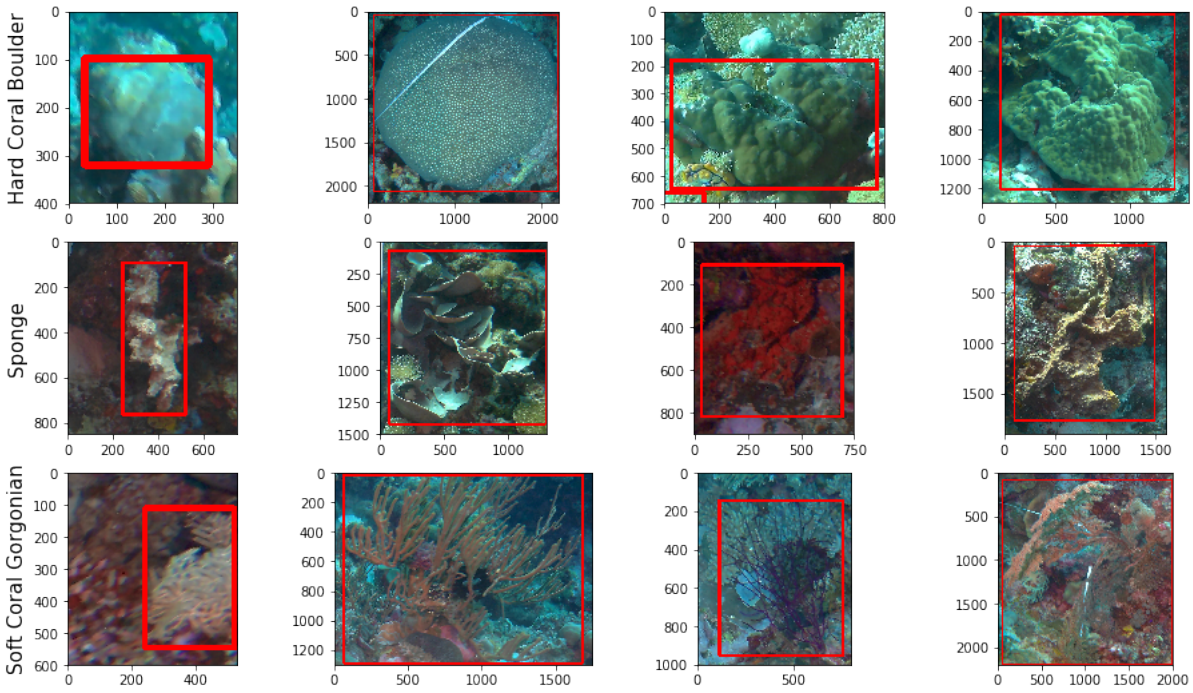
Pravděpodobně nejrozsáhlejší z datasetů pro detekci objektů. Soutěž probýhala během let 2010 až 2017. Dataset obsahuje přes 14 milionů hierarchicky organizovaných obrázků. Obrázky jsou rozděleny do přibližně 22 tisíc pod-tříd, které vycházejí z 27 hlavních tříd. Každá pod-třída obsahuje průměrně 500 obrázků. Přes 1 milion obrázků obsahuje anotace pro rámečky.

2.1.2 Pascal VOC

Jeden z nejpopulárnějších datasetů v počátcích počítačového vidění. Soutěž probíhala během let 2005 až 2012. Dataset obsahuje data pro detekci, segmentaci, detekci částí těla osob nebo detekce akce. Objekty jsou klasifikovány do dvaceti tříd ve čtyřech kategoriích(osoby, zvířata, vozidla a domácí vybavení). Verze z roku 2007 obsahuje přibližně 5 tisíc trénovacích obrázků a 12 tisíc objektů a verze z roku 2012 11 tisíc trénovacích obrázků a 27 tisíc. Nejčastěji používané verze datasetu jsou z let 2007 a 2012. Které se liší v počtu obrázků a některých vedlejších úlohách.

2.1.3 MS COCO

MS COCO je jedním z nejnáročnějších dostupných datasetů nyní. Často jsou modely hodnoceny a porovnávány právě na tomto datasetu. Dataset je určen pro detekci a segmentaci objektů v jejich přirozeném kontextu. Obsahuje 80 tříd a přibližně 7 instancí objektů na obrázek. Pro porovnání Pascal VOC 2007 obsahuje 20 tříd a přibližně 3 instance objektů na obrázek.



Obrázek 2: V každé řádce je ukázka několika instancí jedné třídy.

2.1.4 CoralClef 2020

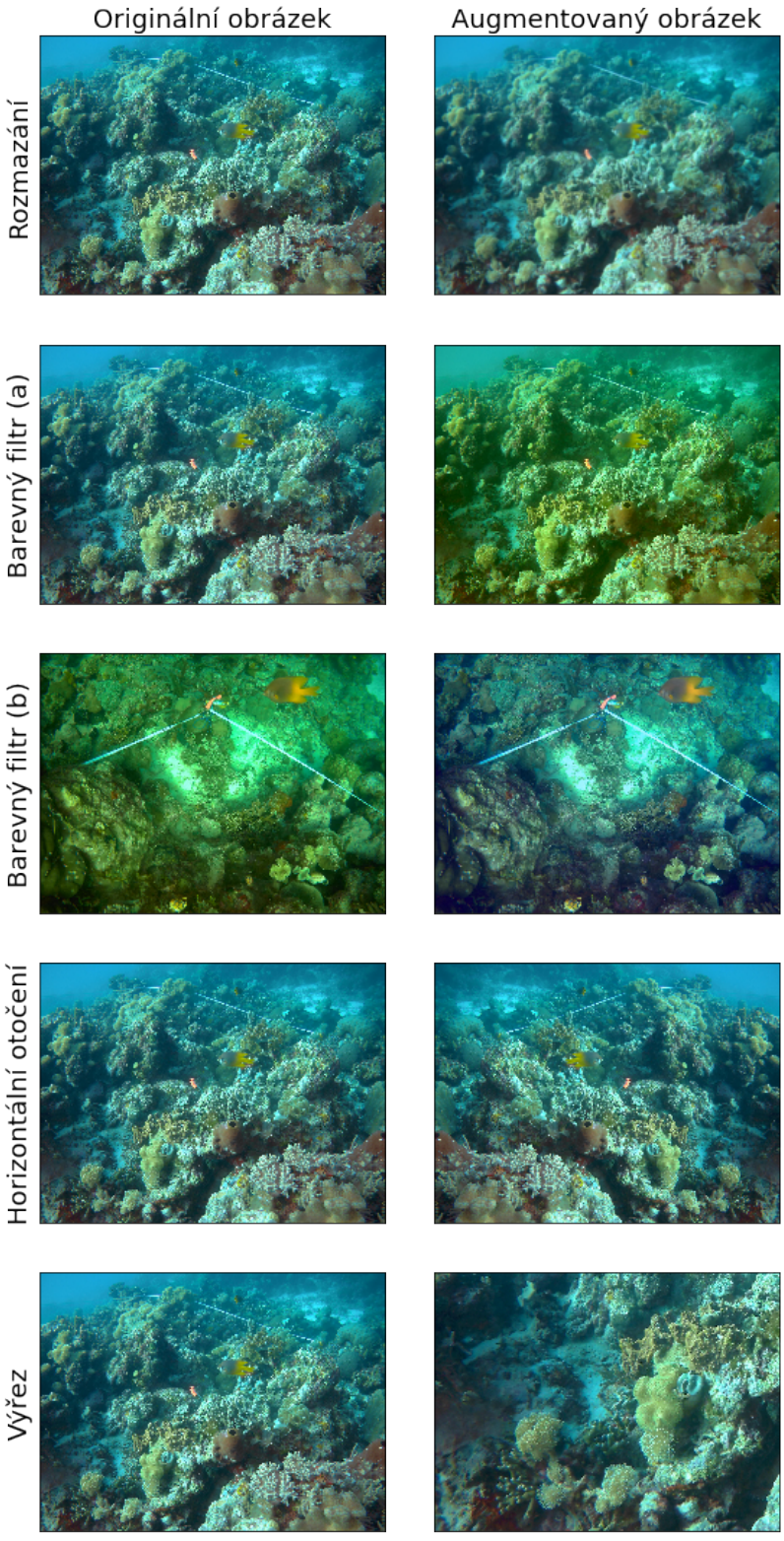
V rámci konference CLEF 2020 byla publikována soutěž ImageCLEF 2020 Coral. Soutěž je zaměřená na detekci korálových útesů. Se změnou klimatu v posledních letech přichází nebezpečí ztráty korálových útesů a ekosystému, který podporují. Protože obrázky těchto korálů jsou komplexní a pro lidi složité anotovat, nabízí se využití automatické detekce. A zjednodušit tak jejich monitorování. Data pro tuto úlohu pochází z kolekce obrázků korálových útesů z celého světa, která vzniká v rámci projektu monitorování korálových útesů na univerzitě v Essexu. Poskytnutá trénovací data obsahují 440 obrázků a anotace pro 13 tříd. V rámci soutěže je možné řešit úlohu detekce a segmentace. Úloha přináší několik výzev. Obrázky obsahují velké množství objektů, průměrně 28. Koráli stejné třídy mohou být různě veliké, mohou mít různý tvar, barvu a strukturu. Ukázka rozmanitosti v rámci jedné třídy je na obrázku 2. Problematická je také kvalita některých obrázků a anotací. Část obrázků je rozmazaná a rámečky nejsou vždy přesně umístěné. Testovací množina obsahuje 400 obrázků ze čtyř různých míst.

- stejné místo jako trénovací množina
- podobné místo jako trénovací množina
- geograficky podobné místo jako trénovací množina
- geograficky odlišné místo od trénovací množiny

2.2 Augmentace

Při trénování neuronové sítě se snažíme síť naučit obecné vlastnosti jednotlivých tříd. Z tohoto důvodu je potřeba, aby trénovací množina obsahovala co nejvíce různých případů, ve kterých se instance jednotlivých tříd mohou reálně vyskytovat. Například, pokud bychom měli dataset, ve kterém jsou pouze kočky a psi, přičemž všechny kočky na všech obrázcích by byly otočené jedním směrem a psi druhým, je velmi pravděpodobné, že by se síť tuto vlastnost naučila a objekty by rozpoznávala podle této vlastnosti. Stejný problém může nastat například s osvětlením. Tento problém se snažíme minimalizovat. Jedním řešením je získat více dat, ale to není vždy možné. Proto se pro rozšíření datasetu používají augmentace, při kterých se z obrázků které již máme, vygenerují další. Vygenerované obrázky by měli být konzistentní s realitou a ostatními obrázky v datasetu. Například, mějme obrázek psa, který je otočený tak že se dívá doprava, další obrázek můžeme vygenerovat horizontálním otočením originálního obrázku, tak získáme obrázek psa, který se dívá doleva. Tato augmentace dává smysl, protože psa můžeme vyfotit jakkoliv otočeného. V tomto případě by nedávalo smysl provést vertikální otočení obrázku, kdy by pes skončil vzhůru nohama. Při použití nevhodných augmentací může dojít k zhoršení výsledků. Kromě vertikálního a horizontálního otočení jsou běžné augmentace jako je otočení o určitý úhel, výřez části obrázku, použití filtru (rozostření, šum, ...), změna kontrastu barev, změna jasu a další. Pro vygenerování obrázku může být použito více augmentací najednou.

Dataset CoralClef 2020, neobsahuje mnoho obrázků, proto bylo vhodné augmentace provést. První z augmentací, které byli použity je rozmazání. V trénovacím datasetu se nachází několik rozmazaných fotek a lze předpokládat, že tomu tak bude i v testovací množině a při reálném nasazení algoritmu. Další augmentace je použití barevného filtru. Obrázky v datasetu byly pořízeny při dvou různých typech osvětlení a část fotek má zelený odstín a část modrý. Další augmentace jsou horizontální otočení a výřez části obrázku. Augmentace jsou zobrazeny na obrázku 3.



Obrázek 3: Ukázka augmentací provedených pro dataset CoralCLef 2020.

3 Detekce objektů

Metody založené na neuronových sítích se často rozdělují do dvou skupin. První skupina se označuje dvoufázové detektory. Model se skládá ze dvou hlavních fází, v první fázi vygeneruje regiony, které potencionálně obsahují objekt a extrahuje z každého vektor příznaků, v druhé fázi je každý region klasifikován. Druhá skupina označovaná jednofázové detektory. Nemají fázi predikce regionů, obvykle berou v úvahu všechny pozice v obrázku a snaží se každý region klasifikovat. Detekce probíhá během jedno průchodu obrázku sítí. Dvoufázové metody jsou často přesnější, ale pomalé. Jednofázové metody jsou velmi rychlé a vhodnější pro použití v reálném čase.

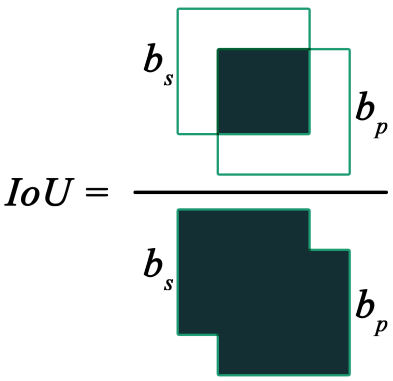
3.1 Míry

3.1.1 Intersection over Union

Intersection over Union (IoU), někdy označováno jako Jaccard index, je míra používaná pro vyhodnocení přesnosti predikovaného rámečku. Předpokládáme, že při trénování sítě máme pro každý vstupní obrázek správné souřadnice rámečků pro všechny objekty, které se v něm nacházejí. IoU je definováno jako poměr mezi sjednocením ploch predikovaného a skutečného rámečku a jejich průnikem:

$$IoU(b_p, b_s) = \frac{|b_p \cap b_s|}{|b_p \cup b_s|} \tag{1}$$

Kde b_p je predikovaný a b_s je skutečný rámeček. Hodnota bude vždy mezi 1 a 0, čím blíže jedné tím je predikce lepší a blíží se velikosti a poloze skutečného rámečku. Pokud je hodnota IoU vetší než zvolená mezní hodnota, často 0.5, je rámeček považován za správně predikovaný.



Obrázek 4: Grafické znázornění IoU

3.1.2 mean Average Precision

Mean Average Precision (mAP), je míra často používaná k hodnocení přesnosti výsledků modelů pro detekci objektů. Obvykle je vypočítáno AP zvlášť pro každou třídu a poté je vypočten průměr AP přes všechny třídy. Pro výpočet AP je nejdříve potřeba vypočítat precision a recall.

Precision udává procento správně detekovaných rámečků, ze všech detekovaných. Jinými slovy, udává s jakou pravděpodobností bude detekovaný rámeček správně.

$$presicion = \frac{TP}{TP + FP} = \frac{TP}{\text{všechny detekované rámečky}}$$

Recall udává procento objektů, které jsou správně detekovány. Pokud by například byl $recall = 0.7$, znamenalo by to, že jsme detekovali 70% správných rámečů.

$$recall = \frac{TP}{TP + FN} = \frac{TP}{\text{všechny správné rámečky}}$$

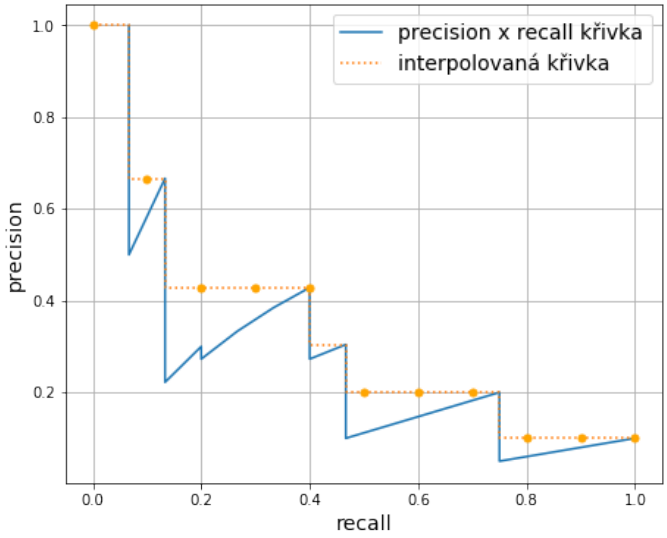
kde:

- TP (true positive) - Predikované rámečky, který má IoU > mezní hodnota se skutečným rámečkem.
- FP (false positive) - Predikované rámečky, který nemá IoU > mezní hodnota se skutečným rámečkem. Nebo predikujeme více rámečků pro jeden objekt.
- FN (false negative) - Nepredikujeme rámeček vůbec, nebo pro rámeček predikujeme špatnou třídu.

Model by mohl být poté porovnán s jiným modelem pomocí křivky, která vznikne ze závislosti precision na recall. Čím déle zůstane hodnota recall vysoká se zvyšující se hodnotou precision, tím je model lepší. Tyto křivky bývají často "klikaté"a těžko se mezi sebou porovnávají, proto se z křivky raději vypočítává AP, které charakterizuje celou křivku jedním číslem. Ukázka křivky je na obrázku 5. AP získáme jako plochu pod křivkou. Pro výpočet se používají dva přístupy.

První přístup, je definován jako průměr precision z jedenácti rovnoměrně rozmístěných bodů na křivce. A může být vypočteno podle vztahu:

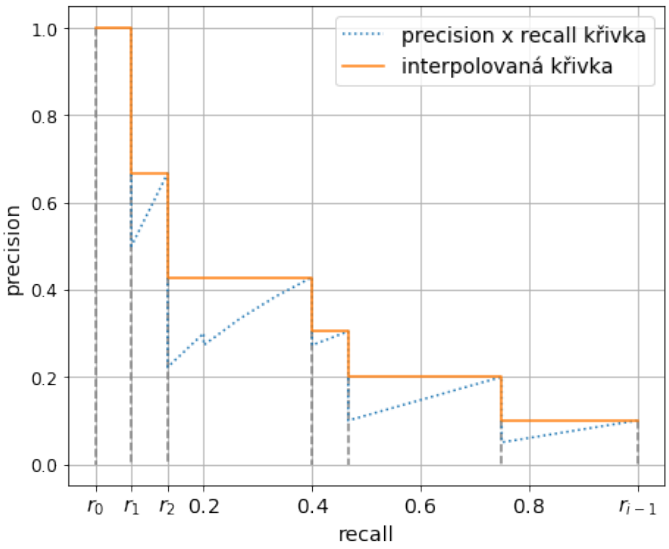
$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1\}} p_{interp}(r) \tag{2}$$



Obrázek 5: Křivka závislosti precision na recall.

Pro zjednodušení výpočtu je křivka interpolována a to tak, že pro hodnoty recall r najdeme novou hodnotu recall $\widetilde{r}$ tak aby platilo $\widetilde{r} \geq r$ a hodnota precision na nové úrovni byla maximální. Ukázka interpolované křivky je na obrázku 5. Interpolovanou hodnotu je vypočtena podle následujícího vztahu:

$$p_{interp}(r) = \max_{\widetilde{r} \geq r} p(\widetilde{r}) \tag{3}$$



Obrázek 6: Výpočet plochy pod interpolovanou křivkou.

Druhý přístup počítá s celou plochou pod interpolovanou křivkou a je definován vztahem:

$$AP = \sum_{n=0}^{i-1} (r_{n+1} - r_n) p_{interp}(r_{n+1}) \tag{4}$$

Kde $r_0, r_1, \ldots, r_{i-1}$ je úroveň recall ve které je poprvé interpolována hodnota precision, graficky jsou tyto úrovně zobrazeny na obrázku 6. Hodnota i je počet interpolovaných úrovní precision. Interpolovaná hodnota je vypočtena obdobně jako v předchozím případě:

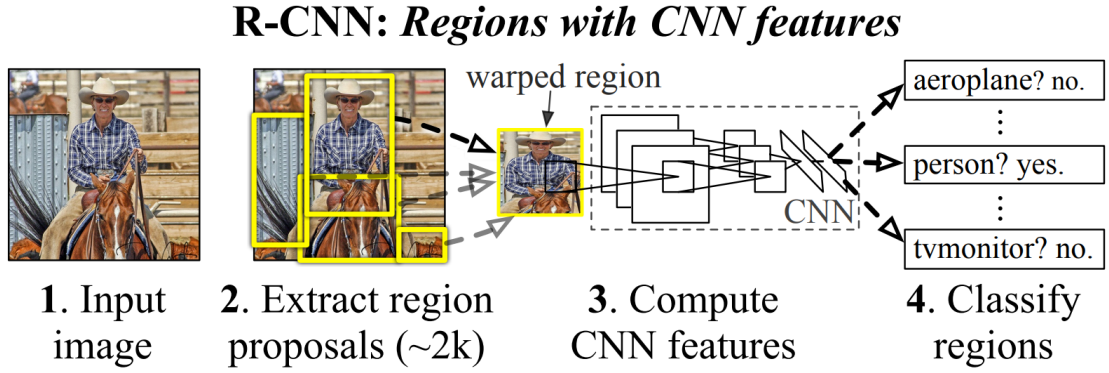
$$p_{interp}(r_{n+1}) = \underset{\widetilde{r} \geq r_{n+1}}{max} \; p(\widetilde{r}) \tag{5}$$

3.2 Přehled algoritmů

V této podkapitole uvádím nejdůležitější detektory založené na neuronových sítích. Jak již bylo řečeno, detektory lze rozdělit do dvou skupin - jednofázové, a dvoufázové. R-CNN byl první dvoufázový detektor založený na neuronových sítích [7]. Práce byla publikována roku 2014 týmem z Kalifornské univerzity v Berkeley. Členové týmu byly Ross Girshicka, Jeffa Donahue a Trevor Darrel. Hlavní nevýhodou R-CNN modelu je dlouhá doba detekce, řádově desítky sekund, proto v nadcházejících letech došlo k několika vylepšením. První model, který na R-CNN navazoval, přišel v roce 2015, publikoval ho Ross Girshicka a nazýval se Fast R-CNN [6]. Dalším vylepšením byl model Faster R-CNN [16] publikovaný Shaoqing Ren ad. Mezi významné jednofázové přístupy pak patří YOLO [15] publikované roku 2015 a SSD [14] publikované o rok později.

3.2.1 R-CNN

R-CNN model [7] se skládá ze tří základních částí. V první fázi je z obrázku vygenerováno přibližně 2000 regionů pomocí selective search [18], které potenciálně obsahují objekty. U každého regionu musí být upravena velikost, aby ho bylo možné vložit do neuronové sítě. V originální práci použili architekturu AlexNet [10] se sedmi vrstvami a vstupní velikostí obrázku 227x227px. Pro každý region je výstupem ze sítě vektor příznaků o délce 4096. V posledním kroku se pomocí lineárních SVM klasifikátorů rozhodne, zda v regionu objekt je a případně jaký. Na obrázku 7 je vývojový diagram z původní práce. V porovnání s ostatními metodami publikovanými do té doby, si R-CNN vedlo velmi dobře. V soutěže ILSVRC2013 [17], byl nejlepší výsledek 22.6% mAP, kdežto R-CNN dosáhlo při stejných podmínkách mAP 31.4%. Na datasetu PASCAL VOC 2007 [4] dosáhlo výsledku mAP 66.0%.



Obrázek 7: Vývojový diagram RCNN [7], skládající se ze 3 částí: (1) návrh regionů, (2) výpočet příznaků pro každý region a (3) klasifikace pomocí SVM

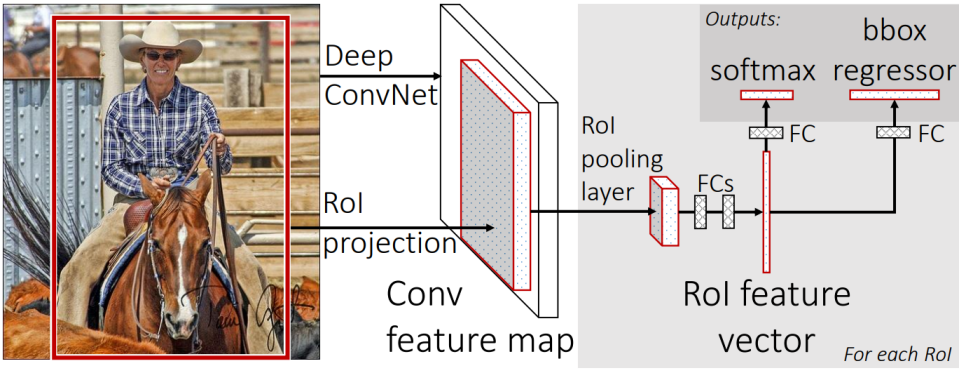
Selective search je algoritmus navržený pro návrh oblastí v obrázku, které potencionálně obsahují objekt. Algoritmus funguje na principu shlukování. Autoři algoritmus navrhli tak, aby splňoval tři podmínky: (1) V jednom obrázku se mohou nacházet různě velké objekty a proto by všechny velikosti měli být brány v úvahu. (2) Objekty mohou sdílet stejnou barvu s okolím nebo stejnou texturu a hrany objektů může být těžké odlišit, proto při slučování regionů je bráno v potaz více kritérií. (3) Algoritmus by měl proběhnout v rozumném čase. Algoritmus začíná rozdělením obrázku do malých regionů. Pro každý region je spočtena podobnost se sousedy a dva nejpodobnější regiony jsou spojeny. Tento proces probíhá dokud nejsou spojeny všechny regiony. Princip shlukování zajišťuje segmentaci objektů různé velikosti, malé objekty vznikají v počátku a velké ke konci průběhu algoritmu. Podobnost dvou regionů je vypočtena podle podobnosti barev, textury, velikosti regionů (upřednostňuje spojení malých regionů) a je upřednostňováno spojení regionů, které spolu sousedí velkou plochou.

R-CNN se potýká s několika problémy. Pro každý region je zvlášť vypočten vektor příznaků, v obrázku se ale velké množství regionů překrývá, to znamená, že pro některé oblasti v obrázku dochází k výpočtu několikrát. Toto nesdílení výpočetního výkonu působilo velmi pomalé trénování i testování. Další problém způsobuje rozdělení do tří separátních modulů, z tohoto důvodu nebylo možné model optimalizovat způsobem end-to-end (od začátku do konce) a najít optimální globální nastavení je náročné. Poslední z hlavních problémů je použití selective search v první části modelu, u kterého nedochází k žádnému trénování a může docházet ke generování špatných kandidátů.

3.2.2 Fast R-CNN

Fast R-CNN [6] přináší velké zlepšení v oblasti rychlosti a také zlepšuje přesnost detekce. Vstupem je obrázek a soubor regionů, které jsou generovány algoritmem selective search. V prvním kroku je celý obrázek vložen do konvoluční neuronové sítě a je vygenerována

mapa příznaků. Následně je každý region zmenšen do mapy příznaků s fixní velikostí, k tomu slouží speciálně pro tento účel navržená RoI pooling vrstva. Mapa příznaků je vložena do sekvence plně propojených vrstev (fully-connected), které se nakonec rozdělí do dvou větví. Jedna větev produkuje pravděpodobnost pro každou třídu a pozadí pomocí softmax funkce. Výstupem druhé vrstvy jsou čtyři souřadnice rámečku pro každou třídu. Architektura Fast R-CNN je na obrázku 8.



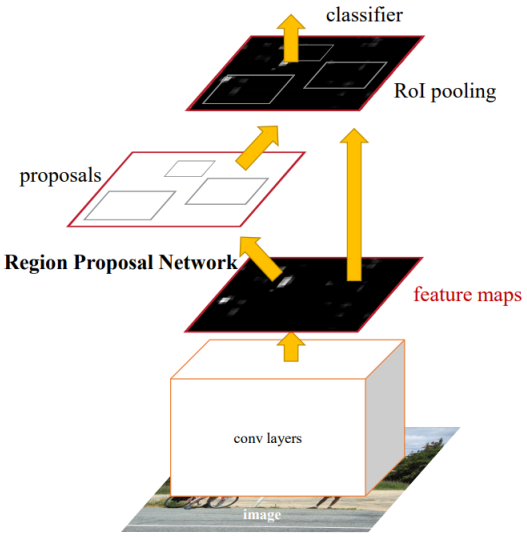
Obrázek 8: Architektura Fast RCNN [7]

Celý model je možné trénovat end-to-end a není potřeba trénovat jednotlivé části samostatně, jako tomu je u R-CNN. V oblasti přesnosti dosáhlo Fast R-CNN na datasetu PASCAL VOC 2007 70.0%. Problém nicméně stále zůstává v oblasti návrhu regionů. R-CNN i Fast R-CNN používají selective search, který je časově náročný a není možné ho trénovat a zlepšit tím návrh regionů.

3.2.3 Faster R-CNN

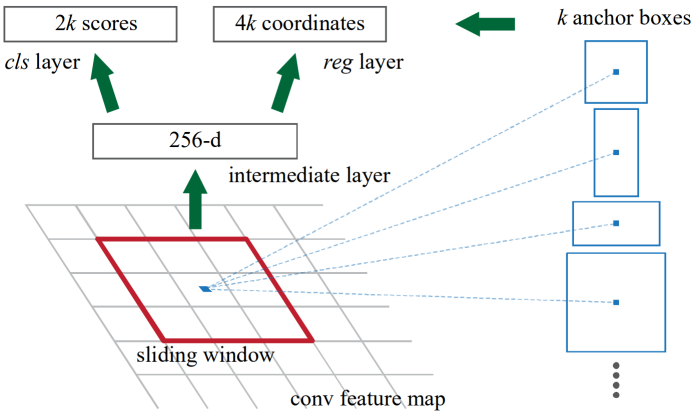
Faster R-CNN [16] odstraňuje poslední z hlavních problémů R-CNN a Fast R-CNN. Již není potřeba pro generování regionů používat externí algoritmus jako je selective search. Přináší až desetinásobné zlepšení času detekce oproti Fast R-CNN. Vstupem je obrázek. V prvním kroku je obrázek vložen do konvoluční neuronové sítě a je vygenerována mapa příznaků, tak jako tomu je u Fast R-CNN. V dalším kroku je malou neuronovou sítí nazývanou "Region Proposal Network"(RPN) vygenerována množina regionů. Dále je postup stejný jako u Fast R-CNN, u každého regionu je změněna velikost průchodem přes RoI pooling vrstvu a dvě výstupní vrstvy určí pravděpodobnost výskytu každého objektu v daném regionu a čtyři hodnoty upravující polohu rámečku. Architektura Faster R-CNN je na obrázku 9.

Vstupem do RPN je mapa příznaků. Výstupem je množina navržených rámečků a pravděpodobnost, s kterou obsahují objekt. Pro RPN je navrženo k fixně velkých rámečků (anchors) s různým poměrem stran. V původní práci použily 3 různé velikosti a 3 různé poměry stran, kombinací všech získáme $k = 9$. Přes mapu příznaků



Obrázek 9: Architektura Faster RCNN [16]

je posouváno okénko o velikosti $n \times n$, obvykle $n = 3$. Pro každou pozici je vygenerován vektor příznaků, který je vložen do dvou paralelních plně propojených vrstev. Pro každou pozici posuvného okénka jsou výstupem první výstupní vrstvy 2 hodnoty pro každé fixní okénko, tedy $2k$ hodnot. Hodnoty určují pravděpodobnost s jakou je na dané pozici objekt nebo pozadí. Druhá výstupní vrstva generuje 4 hodnoty pro každé okénko, tedy $4k$ hodnot. Tyto hodnoty upravují velikost okénka. Síť je implementována $n \times n$ konvoluční vrstvou následovanou dvěma paralelními 1×1 konvolučními vrstvami. Ilustrace RPN pro jednu pozici posuvného okénka je na obrázku 10.

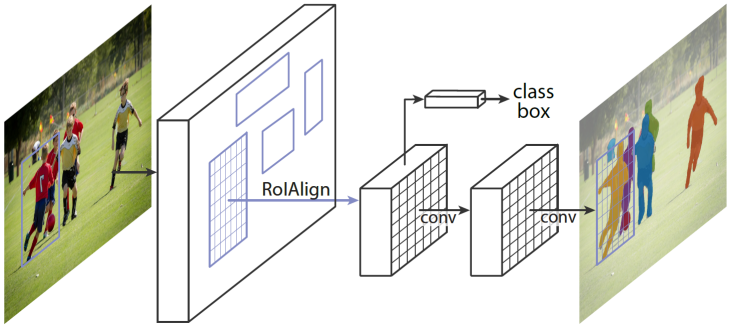


Obrázek 10: Region Proposal Network (RPN) [16]

Model přináší zlepšení v přesnosti detekce výsledek na datasetu PASCAL VOC 2007 mAP 78.8%. Hlavní zlepšení je v oblasti času detekce, díky použití RPN pro návrh regionů. U Fast R-CNN dosahuje detekce 0.5 FPS, u Faster R-CNN až 7 FPS.

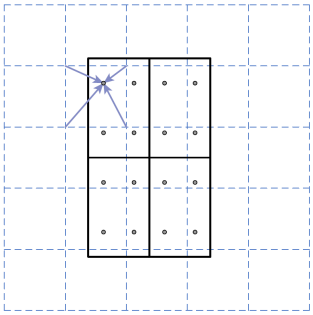
3.2.4 Mask R-CNN

Všechny dosud představené modely predikovali pouze rámeček pro každý objekt. Mask RCNN [8] navíc přidává funkcionalitu segmentace a pro každou instanci predikuje masku. Mask R-CNN navazuje na Faster R-CNN a větve pro predikci rámečků a skóre pro jednotlivé třídy rozšiřuje o větev pro predikci masek. Architektura sítě je na obrázku 11. Větev pro predikci masek je malá "Fully Convolutional Networks"(FCN). FCN generuje binární masku nezávisle pro každou třídu, třída je určena podle větve predikující skóre pro jednotlivé třídy. Další změnou oproti Faster R-CNN, která vede k zlepšení segmentace je záměna RoI pooling vrstvy za RoIAlign vrstvu. RoI pooling vrstva slouží ke zmenšení navržených regionů na fixní velkost, nicméně při tomto procesu dochází kvantizací ke ztrátě informace. To pro detekci rámečků až tak nevadí, ale při segmentaci, kdy chceme určit, zda každý pixel patří do třídy nebo ne, to může působit problémy a snížit přesnost. RoIAlign vrstva má stejnou funkci, ale řeší tento problém.



Obrázek 11: Architektura Mask R-CNN [8]

Mapa příznaků je k-krát menší než vstupní obrázek. Pro navržený region se souřadnicemi x je potřeba provést x/k, aby bylo možné umístit region na mapu příznaků a vybrat příslušné pixely. Nové souřadnice nemusejí vyjít jako celé číslo. Takto zmenšený region je umístěn přes mapu příznaků a rozdělen do mřížky $n \times n$. V každé buňce mřížky jsou rovnoměrně rozmístěny 4 body. Pro každý bod je bilineární interpolací dopočítána hodnota z mapy příznaků. Hodnota pro každou buňku je určena jako maximum nebo průměr z hodnot bodů, které se v ní nacházejí, viz obrázek 12.



Obrázek 12: Čárkovanou čarou je vyznačena mapa příznaků. Plnou černou čarou zmenšený region, rozdělený do mřížky 2×2. Pro každý bod v mřížce je dopočtena hodnota z mapy příznaků. [8]

3.2.5 YOLO (You Only Look Once)

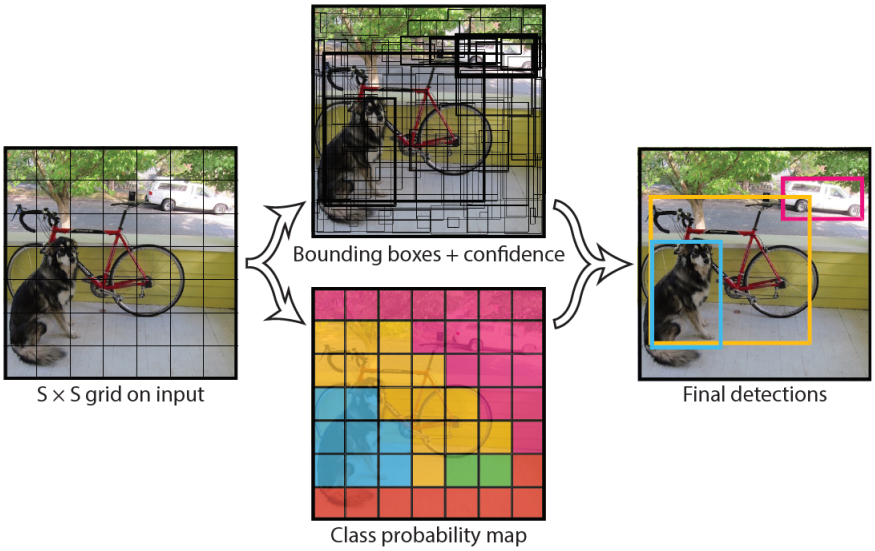
První jednofázová metoda, představil jí J. Redmon ad. [15] roku 2015 po Faster R-CNN. Hlavním přínosem je detekce v reálném čase. Celý model se skládá pouze z jedné konvoluční neuronové sítě, obrázek touto sítí prochází celý a pouze jednou. To, a nižší počet predikovaných rámečků než u Faster R-CNN je hlavním důvodem rychlé detekce.

Systém rozdělí vstupní obrázek do mřížky $S \times S$. Pokud se střed objektu v obraze nachází v buňce, je tato buňka zodpovědná za jeho detekci. Každá buňka může predikovat až B rámečků a hodnotu spolehlivosti s jakou model věří, že se v rámečku objekt nachází a jak přesně je rámeček umístěn. Pokud se v buňce objekt nenachází, hodnota spolehlivosti bude nula, jinak chceme, aby hodnota spolehlivosti byla rovna IoU mezi predikovaným a skutečným rámečkem. Kromě hodnoty spolehlivosti jsou pro každý rámeček predikovány 4 další hodnoty. Dvě hodnoty (x, y), které určují střed rámečku relativně k buňce, a (w, h) šířka a výška rámečku relativně k celému obrázku, všechny hodnoty se pohybují v rozmezí 0 a 1. Jako poslední predikuje model C pravděpodobností pro každou třídu. Tento postup je graficky znázorněn na obrázku 13. Pro každou buňku je predikována jedna množina pravděpodobností tříd, nezávisle na počtu rámečků B. Pro každou buňku je tedy výstupem vektor ve tvaru:

$$y = \begin{bmatrix} P_1 & x_1 & y_1 & w_1 & h_1 & \cdots & P_B & x_B & y_B & w_B & h_B & c_1 & \cdots & c_C \end{bmatrix}$$

Celkový výstup sítě bude tenzor o dimenzi $S \times S \times (B * 5 + C)$. Každá buňka může pro jeden objekt generovat více rámečků, v posledním kroku je pro každý objekt zachován pouze jeden rámeček, při trénování sítě právě ten, který má nejvyšší IoU se skutečným rámečkem, při testování pak ten s nejvyšší hodnotou spolehlivosti, stejně tak jsou odstraněny všechny rámečky, které mají nižší hodnotu spolehlivosti než je zvolený práh.

Model byl implementován neuronovou sítí s 24 konvolučními vrstvami následovanými dvěma plně propojenými vrstvami. Architektura sítě je zobrazena na obrázku 14. Tento



Obrázek 13: YOLO diagram [15]. (1) rozdělení obrázku do mřížky, (2) predikce rámečků a tříd pro každou buňku a (3) odstranění rámečků s malou hodnotou spolehlivosti.

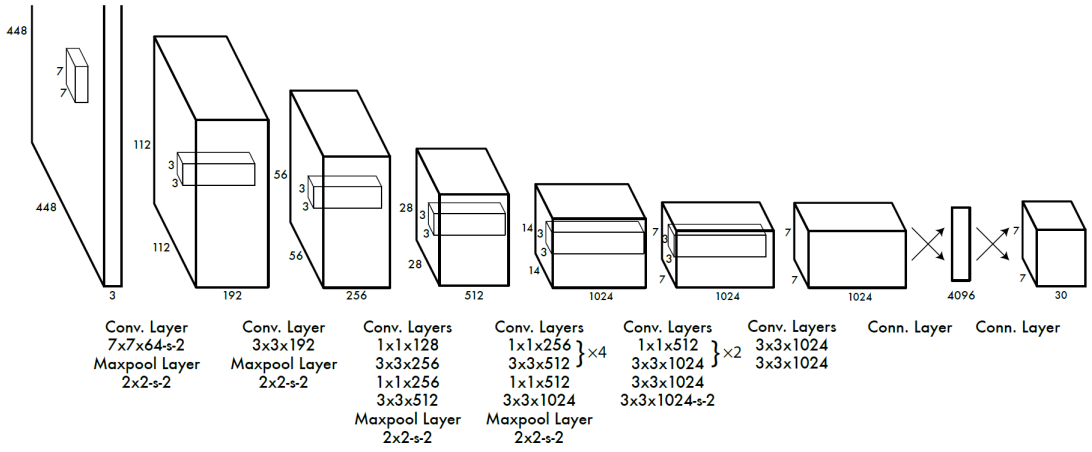
model je schopný detekovat s rychlostí 45 FPS a mAP 63.4% na datasetu PASCAL VOC 2007. S tímto modelem byla publikována i rychlejší verze YOLO s 9 konvolučními vrstvami a nižším počtem filtrů v jednotlivých vrstvách. Dosahoval rychlosti 155 FPS a mAP 52.7%.

YOLO predikuje pro každou buňku pouze jednu třídu, i přesto, že může v každé buňce predikovat více rámečků, ty se v průběhu trénování můžou specializovat na predikci různě velkých objektů a zlepšit tak přesnost. Z toho důvodu má tento model problém detekovat malé objekty, které se nacházejí blízko sebe, jako jsou například davy lidí. Model má také problémy s detekcí objektů s velikostí nebo poměrem stran, které nikdy předtím neviděl.

3.2.6 SSD (Single Shot MultiBox Detector)

SSD se objevil krátce po YOLO, je to další z jednofázových detektorů, představil ho W. Liu ad. [14]. YOLO predikuje rámečky pouze z poslední vrstvy, to působí problém s detekcí malých objektů, nebo objektů s neobvyklým poměrem stran. Hlavním přínosem SSD je predikce na více vrstvách s různou velikostí.

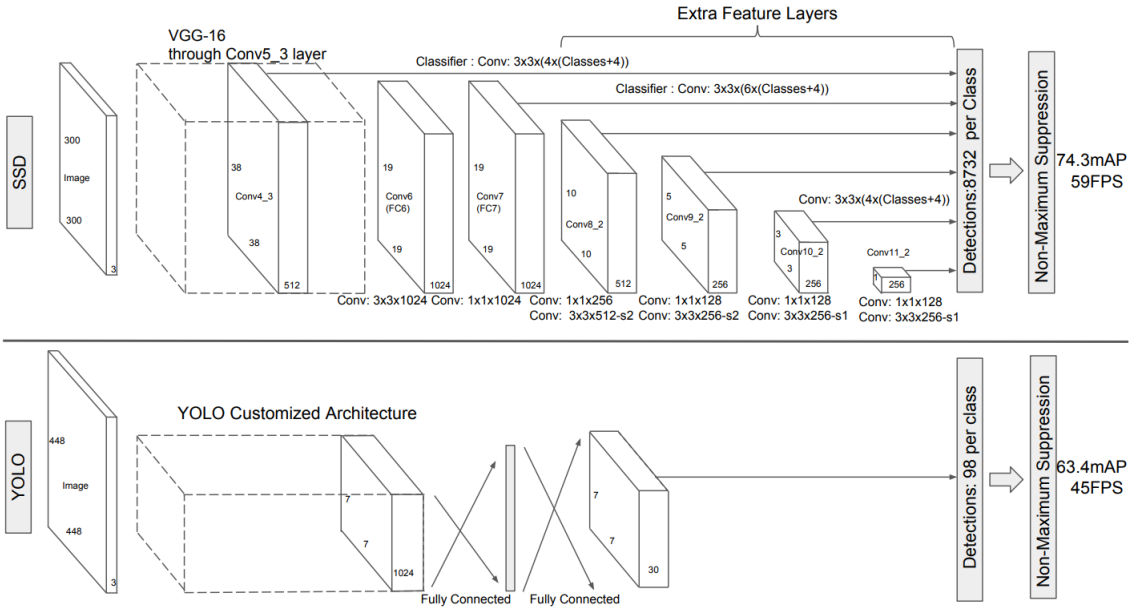
Celý model se skládá pouze z jedné sítě a obrázek jí prochází pouze jednou, tak jako tomu je u YOLO. Síť je možné rozdělit na dvě části, první základní síť, na kterou navazuje pomocná struktura složená z několika konvolučních vrstev. Porovnání architektury YOLO a SSD je na obrázku 15. Základní síť se stará pouze o extrakci mapy příznaků. Pro pomocnou strukturu, která následuje, je předem zvoleno několik fixních



Obrázek 14: Architektura YOLO, síť byla trénována na datasetu PASCAL VOC, s mřížkou 7×7 a s dvěma rámečky pro každou buňku. [15]

rámečků s různou velikostí a poměrem stran. Pomocná struktura se skládá z několika konvolučních vrstev, které postupně zmenšují mapu příznaků. K predikci rámečků požívá SSD malé konvoluční filtry (3×3), které jsou aplikovány na každou pozici ve všech mapách příznaků. Filtry predikují pro každou pozici c pravděpodobností pro výskyt jednotlivých tříd a 4 hodnoty pro každý z k předdefinovaných rámečků, které upravují jeho velikost a polohu. Každá vrstva tedy aplikuje $(c + 4)k$ konvolučních filtrů pro každou pozici. Velké mapy příznaků na začátku struktury se starají o detekci malých objektů, kdežto malé na konci detekují velké objekty.

SSD bylo publikováno ve dvou variantách, odlišujících se ve velikosti vstupního obrázku. SSD300 vyžaduje velikost vstupního obrázku 300×300 a SSD512 vyžadující velikost 512×512. SSD512 dosahuje samozřejmě lepších výsledků. U malých objektů může dojít průchodem sítí ke ztrátě informace. V porovnání s YOLO, je SSD rychlejší i přesnější. Při použití stejné základní sítě dosahuje SSD300 na datasetu PASCAL VOC 2007 mAP 74.3% a rychlosti 46FPS, SSD512 mAP 76.8% a rychlosti 19FPS a YOLO mAP 66.4% a rychlosti 21FPS. SSD300 je jediný model dosahující detekce v reálném čase s mAP nad 70%. Autoři SSD udávají, že až 80% času je spotřebováno při průchodu základní sítí. Při použitím rychlejší sítě by tedy mohlo reálného času dosáhnout i SSD512.



Obrázek 15: Architektura SSD a YOLO, porovnání architektur dvou jedno-fázových modelů, publikováno týmem, který navrhl SSD [14]

Metoda	mAP [%]	FPS	Rozlišení vstupu
R-CNN	66.0		
Fast R-CNN	70.0	0.5	
Faster R-CNN (VGG16)	78.8	7	
YOLO (VGG16)	66.4	21	448×448
SSD300 (VGG16)	74.3	46	300×300
SSD512 (VGG16)	76.8	19	512×512

Tabulka 1: Porovnání výsledků jednotlivých sítí na datasetu Pascal VOC 2007. [14]

4 Aplikace detekce objektů

V této části se budeme zabývat detekcí objektů pomocí některých z představených metod a jejich porovnání. Detekci budeme provádět na datasetu CoralClef2020. Na začátku jsme metody detekce rozdělili do dvou skupin, jedno-fázové a dvou-fázové. Z každé skupiny použijeme jednu z metod. Protože dataset obsahuje i data pro segmentaci, z dvou-fázových metod budeme používat Mask R-CNN. Dataset obsahuje velké množství objektů v jednotlivých obrázcích proto je YOLO pro tuto úlohu nevhodné a SSD bude vhodnější.

4.1 Implementace algoritmů

V rámci této práce jsem využil již hotovou implementaci vybraných modelů. Tyto implementace byly vybrány na základě velké podobnosti s originálnímy modely. Pro Mask R-CNN využijeme [1] (https://github.com/matterport/Mask_RCNN) a pro SSD [5] (https://github.com/pierluigiferrari/ssd_keras). Oba modely jsou implementovány ve frameworku Keras s backendem TensorFlow. Implementace SSD udává výsledky na datasetu Pascal VOC 2007, které jsou o trochu lepší než u originální implementace, stejně tomu tak je s rychlostí. Implementace Mask R-CNN bohužel nepublikuje žádné výsledky. Pro ověření funkčnosti byl model trénován po deset epoch na datasetu Pascal VOC 2007, při použití předtrénovaných vah na MS COCO a dosáhl výsledku 61.7%.

4.2 Rozdělení dat

Aby bylo možné model otestovat. Je potřeba rozdělit data na trénovací a validační množinu. Testovací množina je autory poskytována zvlášť a obsahuje 400 obrázků. Při trénování může dojít k přetrénování modelu, to znamená, že se síť nebude učit obecné vlastnosti objektů, ale konkrétní rozmístění objektů v trénovací množině. Aby bylo možné zjistit, zda nedochází k přetrénování modelu, budeme počítat během trénování chybu nejen pro trénovací, ale i pro validační množinu. Pokud bude chyba obou množin klesat podobně, je vše v pořádku, pokud chyba trénovací množiny klesá o dost rychleji než chyba validačního setu, dochází pravděpodobně k přetrénování. Validační množina by měla obsahovat instance jednotlivých tříd ve stejném rozložení jako tomu je v trénovací množině a měla by obsahovat 10-20% z celkového počtu obrázků. Proto je potřeba rozdělit obrázky podel instancí tříd, které obsahují. Zastoupení jednotlivých instancí v obou množinách je v tabulce 2. Trénovací množina obsahuje 371 obrázků a validační množina 69, to je přibližně 15.7% z celkového počtu, průměrný počet instancí je 16.4%.

Třída	Celkový počet instancí	Instance v trénovací množině [%]	Instance ve validační množině [%]
Hard coral branching	1181	76 (893)	24 (288)
Hard coral submassive	198	87 (172)	13 (26)
Hard coral boulder	1642	83 (1364)	17 (278)
Hard coral encrusting	946	86 (816)	14 (130)
Hard coral table	21	86 (18)	14 (3)
Hard coral foliose	177	86 (153)	14 (24)
Hard coral mushroom	233	80 (179)	20 (44)
Soft coral	5663	79 (4459)	21 (1204)
Soft coral gorgonian	90	88 (79)	12 (11)
Sponge	1691	90 (1514)	10 (177)
Sponge barrel	139	77 (107)	23 (32)
Fire coral millepora	19	84 (16)	16 (3)
Algae macro or leaves	92	85 (78)	15 (14)

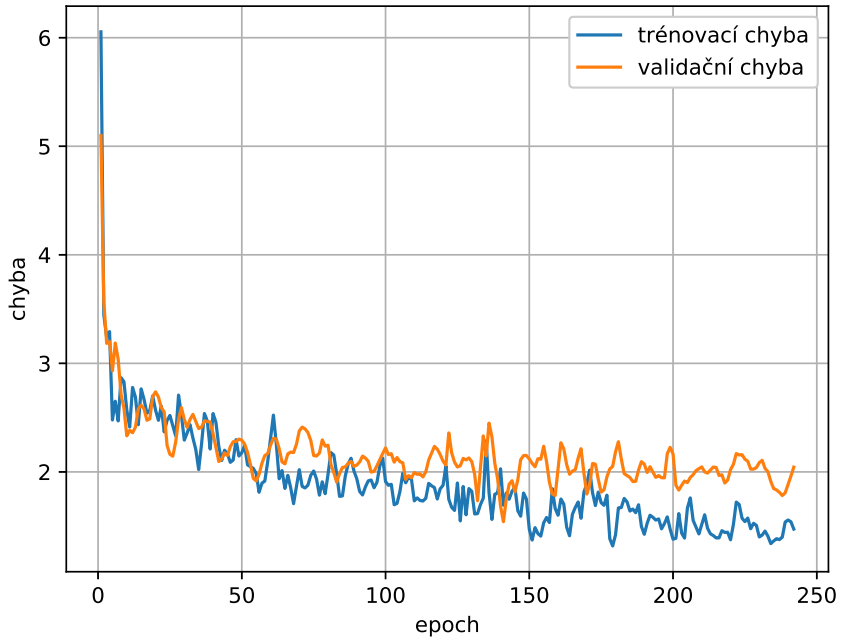
Tabulka 2: Rozdělení dat do trénovací a validační množiny.

4.3 Trénování modelů

V této části se budeme zabývat trénováním modelů. Pro SSD i Mask R-CNN použijeme předtrénované váhy na datasetu MS COCO pro urychlení trénování. Při trénování se síť může naučit rozpoznávat různé tvary, textury a podobně. Počáteční vrstvy se zpravidla naučí detekovat nízko-úrovňové vlastnosti objektů, jako jsou hrany, křivky nebo barevné přechody, vrstvy dále v síti se pak mohou naučit detekovat komplexnější objekty. MS COCO obsahuje rozmanité třídy, proto je pravděpodobné, že se síť dokázala naučit obecné vlastnosti objektů, které je možné aplikovat na velkou škálu jiných objektů. Těchto vlastností bychom chtěli využít a síť pouze dotrénovat na datasetu CoralClef 2020.

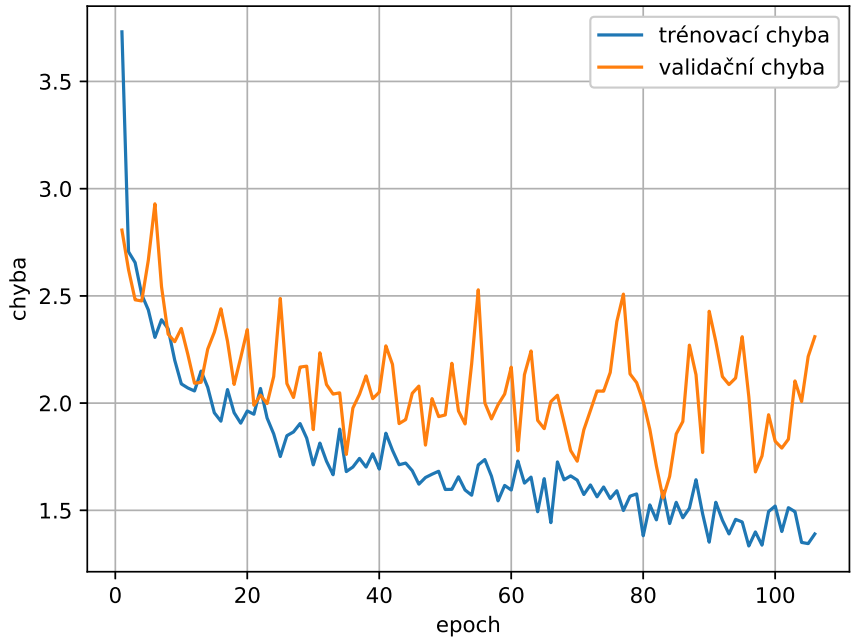
4.3.1 Trénování Mask R-CNN

Model budeme trénovat se základní sítí ResNet-101 [9] se vstupní velikostí $1024 \times 1024px$. Aby bylo později možné určit, jaký vliv mají na trénování sítě augmentace, byl jako první model trénovat na datasetu bez augmentací. Síť byla trénována s optimizerem SGD a konstantou učení $lr = 0.00001$. Konstanta učení je zde poměrně nízká, při vyšší hodnotě docházelo k přetrénování. A to proto, že jsme použili již přetrénovanou síť. Průběh chyby je zobrazen v obrázku 16. Od epochy 150 docházelo jen k malému zlepšení validační chyby a model se začal přetrénovávat.



Obrázek 16: Průběh chyby trénování Mask R-CNN na neaugmentovaných datech

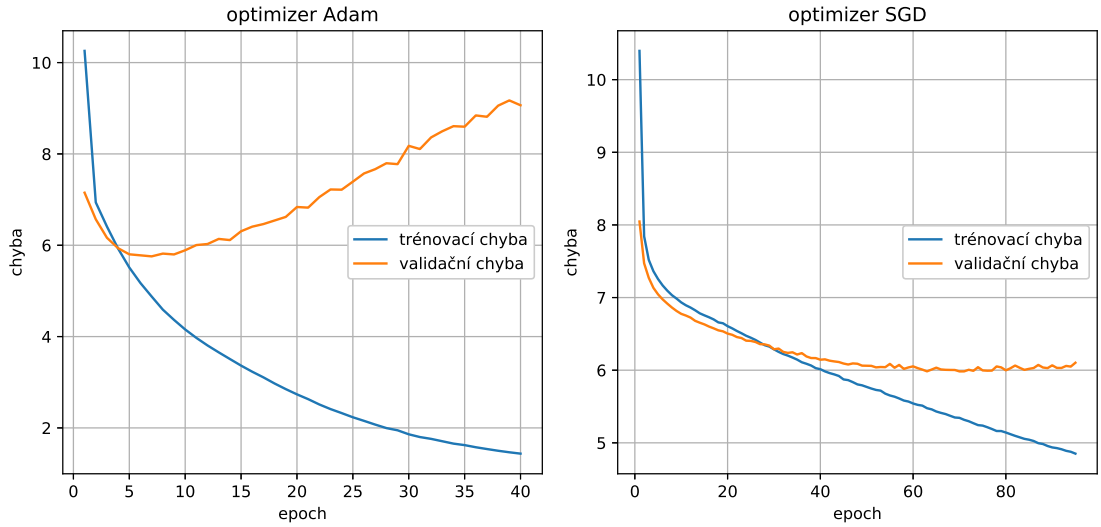
Protože nemusejí augmentace vždy výsledek trénování zlepšit, byly další modely trénovány na různě augmentovaných datech. Nejlepších výsledků bylo dosaženo při trénování s augmentacemi: barevný filtr, horizontální otočení a rozmazání. Po augmentování obsahovala trénovací množina 2968 obrázků a validační 552. Síť byla trénována jako v předchozím případě s optimizerem SGD a konstantou učení $lr = 0.00001$. Průběh chyby na obrázku 17, je méně stabilní než v předchozím případě, nicméně výsledná chyba je nižší. Nejlepší výsledek mAP pro validační množinu 10.18%. Výsledky pro jednotlivé třídy jsou v tabulce 3. U tříd jejichž instance jsou v datasetu zastoupeny v nízkém počtu jsou výsledky špatné a síť se tyto objekty detekovat nenaučila.



Obrázek 17: Průběh chyby trénování Mask R-CNN na augmentovaných datech

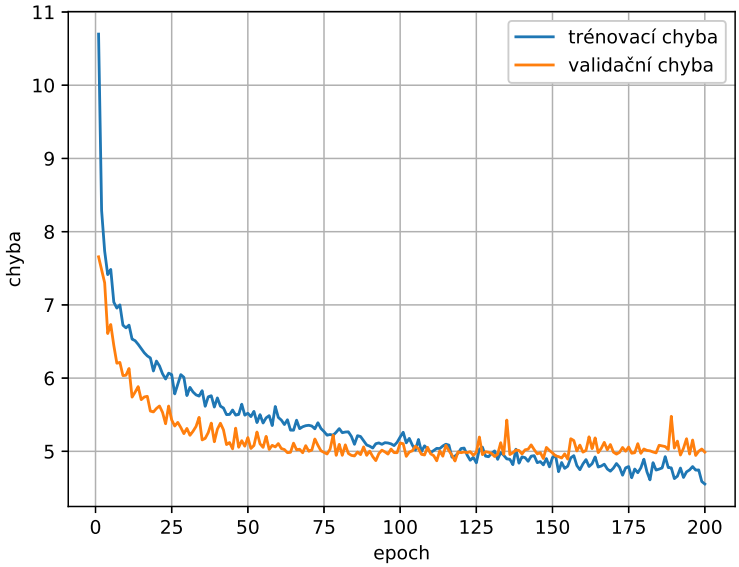
4.3.2 Trénování SSD

Model budeme trénovat se základní sítí VGG-16 se vstupní velikostí obrázku $512 \times 512px$. Při trénování s neaugmentovanými daty s optimizerem Adam a konstantou učení $lr = 0.00001$ docházelo k přetrénování modelu na trénovací data a validační chyba divergovala. Tento problém mohl být způsobený nízkým počtem obrázků, proto jsme pokračovali s trénováním s augmentovanými daty. Po augmentacích bylo v trénovací množině 3710 obrázků a ve validační 690. Byly použity augmentace: barevný filtr, horizontální otočení, výřez a rozmazání. Nicméně problém přetrvával a k přetrénování stále docházelo. V obrázku 18 jsou průběhy chyb pro trénování modelu s optimizerem Adam a SGD. Změna optimizeru měla pouze vliv na dobu, kdy došlo k přetrénování. Nejnižší hodnota validační chyby při trénování s SGD byla vyšší než nejnižší chyba při trénování s optimizerem Adam. S optimizerem SGD dosáhl model mAP 3.78% a s Adam mAP 7.74%.

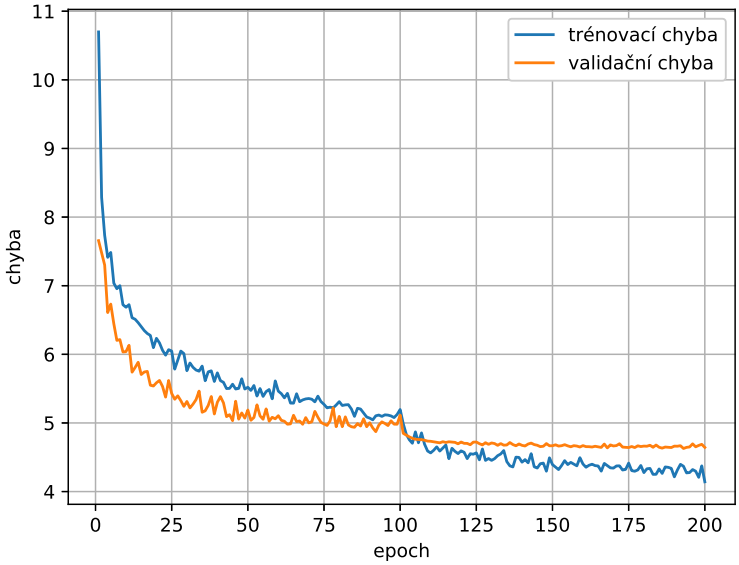


Obrázek 18: Průběh chyby trénování SSD na augmentovaných datech pro dva různé optimizery.

I po augmentacích není 3710 obrázků stále mnoho pro trénování a další augmentace by mohly pomoci, nicméně implementace SSD, kterou používáme, poskytuje možnost generování augmentací během trénování. Stejné augmentace byly použity v originální práci a jsou podobné augmentacím, které jsme doposud používali. Výhodou generování augmentací během trénování je, že při každém průchodu, jsou síti předloženy jinak augmentované obrázky. Nevýhodou je pomalé trénování. Model jsme trénovali s augmentacemi generovanými během trénování na originálních obrázcích s optimizerem Adam a počáteční konstantou učení $lr = 0.0001$. Průběh chyby je na obrázku 19. Obě chyby konvergují podobně rychle. Od epochy 100 klesá pouze trénovací chyba a validační chyba zůstává na hodnotě 5. Model jsme trénovali znovu od epochy 100 s nižší konstantou učení $lr = 0.00001$. Validační chyba po této změně dále klesala, průběh je na obrázku 20. Hodnota mAP pro validační množinu je 14.94%. Hodnota mAP pro jednotlivé třídy jsou v tabulce 3. Model má problém s detekcí instancí tříd, které se v datasetu objevují jen zřídka, jako hard coral submassive nebo soft coral gorganian. Výjimkou je třída sponge barrel, pro kterou je v datasetu málo výskytů, nicméně má ze všech tříd nejlepší výsledek. To je pravděpodobně způsobeno velkou odlišností objektu od ostatních tříd.



Obrázek 19: Průběh chyby trénování SSD na datech augmentovaných během trénování. Počáteční konstanta učení $lr = 0.0001$.



Obrázek 20: Průběh chyby trénování SSD na datech augmentovaných během trénování. S konstantou učení $lr = 0.0001$ do epochy 100 a s $lr = 0.00001$ dále.

Třida	Mask R-CNN	SSD
Hard coral branching	19.56	23.13
Hard coral submassive	0.00	0.00
Hard coral boulder	25.76	33.29
Hard coral encrusting	2.35	5.48
Hard coral tablet	0.00	0.00
Hard coral foliose	8.33	14.24
Hard coral mushroom	5.91	15.91
Soft coral	44.97	34.41
Soft coral gorgonian	0.00	0.00
Sponge	3.52	5.55
Sponge barrel	21.98	62.17
Fire coral millepora	0.00	0.00
Algae macro or leaves	0.00	0.00
mAP	10.18	14.94

Tabulka 3: Nejlepší dosažené výsledky mAP pro jednotlivé třídy pro validační množinu CoralClef 2020.

4.4 Porovnání výsledků

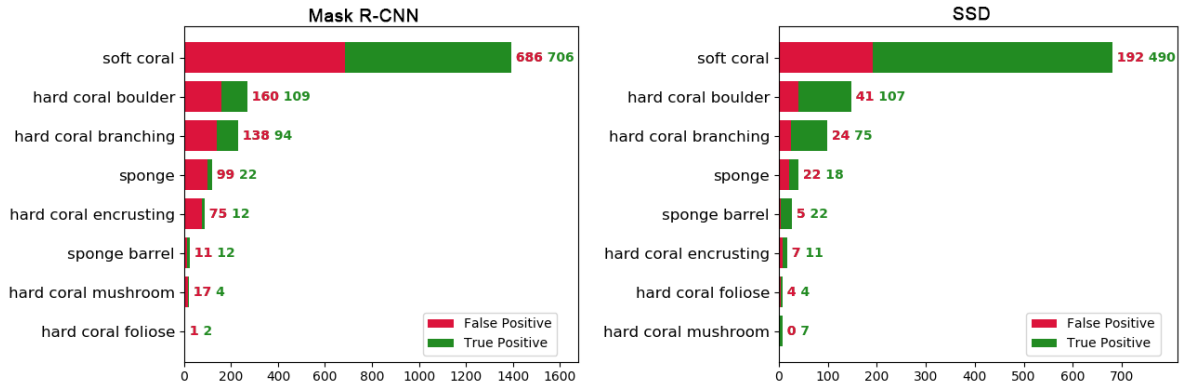
Porovnání času Rychlost detekce pro oba modely je v tabulce 4. Výsledky byly dosaženy na grafické kartě GeForce GTX 1060 6GB. SSD si v porovnání s originální prací vede velmi dobře. V originální práci použili pro detekci výkonnější grafickou kartu Titan X a dosáhly výsledku 19 FPS (tabulka 1). Jak lze očekávat Mask R-CNN je výrazně pomalejší než SSD.

Metoda	FPS	Velikost batch	Velikost vstupu
Mask R-CNN	1	1	1024 x 1024
SSD	14	1	512 x 512

Tabulka 4: Porovnání rychlosti obou metod.

Metoda	Úloha	validační mAP	testovací mAP
Mask R-CNN	lokalizace	10.18	24.3
Mask R-CNN	segmentace	10.29	30.4
SSD	**lokalizace**	**14.94**	**49.0**

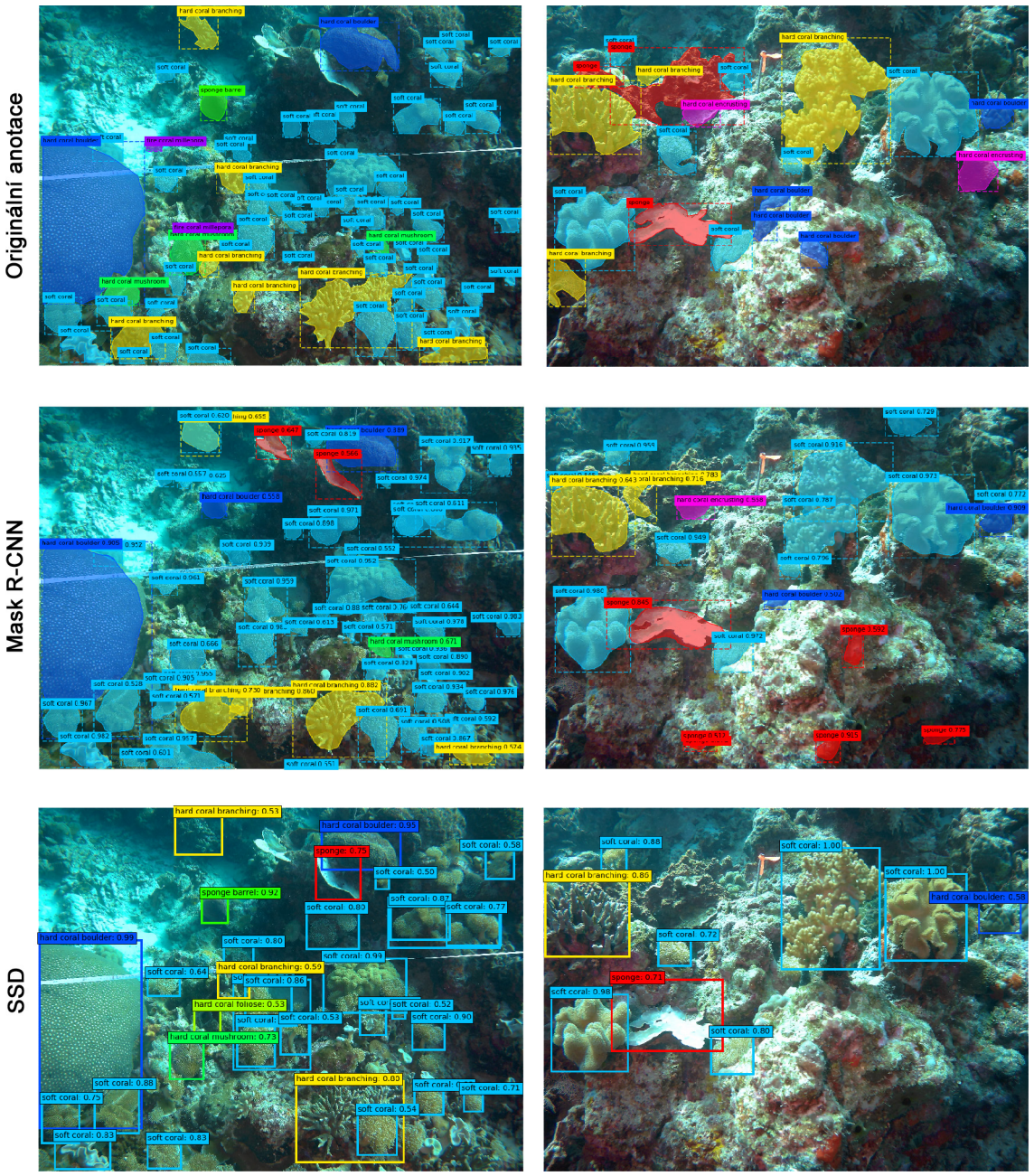
Tabulka 5: Porovnání výsledků obou modelů na validační množině s 69 obrázky a testovací množině s 400 obrázky. V obou případech s IoU = 0.5.



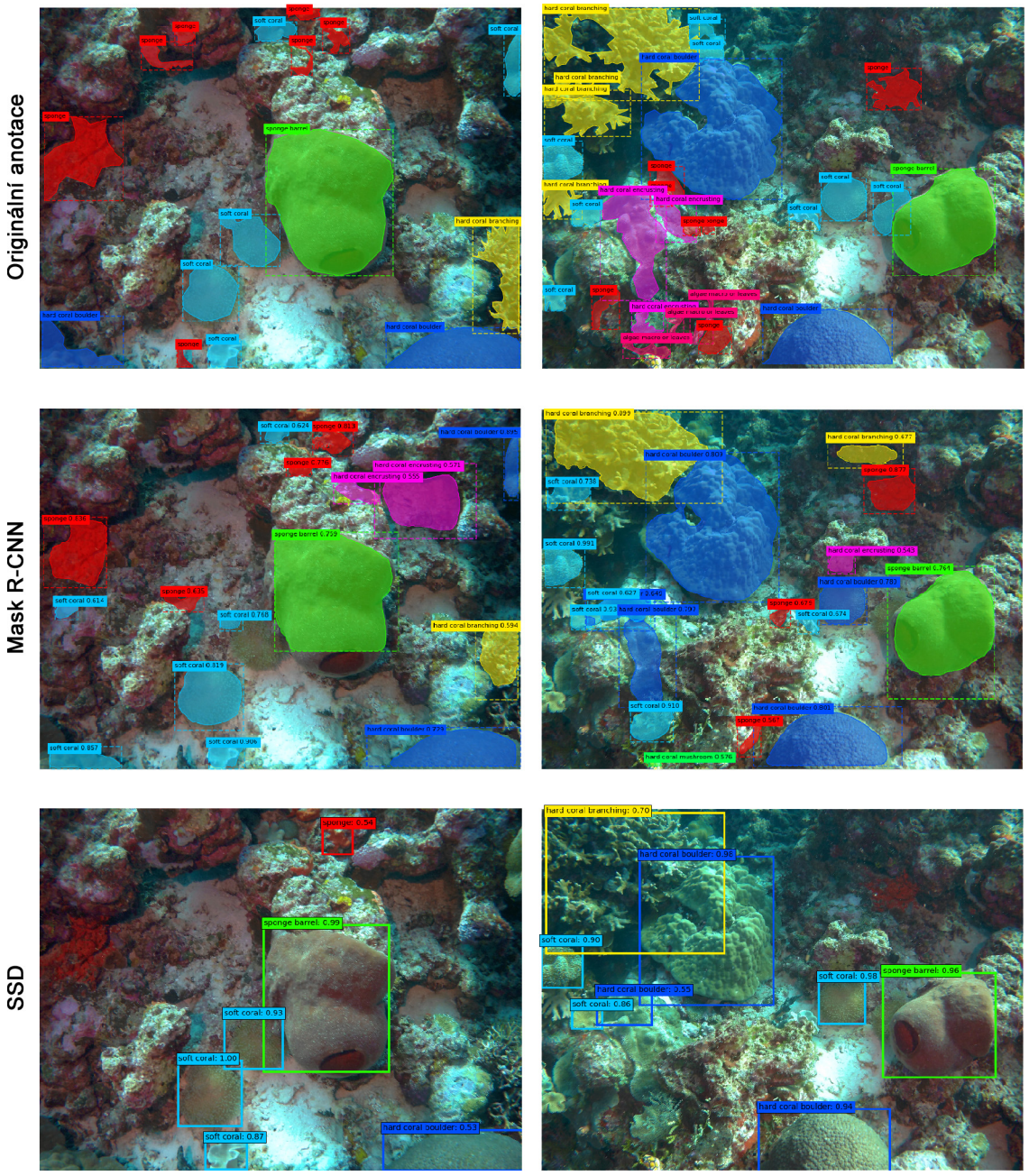
Obrázek 21: Porovnání detekovaných rámečků.

Na obrázcích 22, 23 je několik výsledků pro obrázky z validační množiny. Z obrázků se může zdát, že Mask R-CNN detekuje více rámečků a výsledky by tedy měli být lepší. A i tomu tak je, na obrázku 21 je zobrazeno počet detekovaných objektů. Mask R-CNN detekoval celkově 2148 z toho je pouze 44.7% true positive, tedy správně detekováno. SSD detekovalo 1029 rámečků a z toho je 71.3% true positive. Pokud by jsme model hodnotili pouze podle počtu správně detekovaných rámečků, byl by lepší Mask R-CNN. Lze říct, že Mask R-CNN detekuje více rámečků ale méně kvalitně, kdežto SSD málo rámečků, ale kvalitně. Proto se při porovnání obrázků může zdát, že Mask R-CNN funguje lépe, ale často se stává, že model detekuje objekt, kde nic není, nebo pro jeden objekt detekuje více rámečků.

Výsledky pro testovací množinu jsou automaticky vypočteny autory soutěže, a proto máme o výsledcích jen částečné informace. Výsledky pro oba modely jsou v tabulce 5.V předchozím ročníku CoralClef 2019 [2] byl nejlepší dosažený výsledek 24.27 mAP, prakticky stejně jako dosáhlo Mask R-CNN, SSD dosahuje až dvojnásobek. Takto dobré výsledky mohou být způsobeny velkým počtem instancí tříd, které se modely naučili dobře detekovat v testovací množině. Průměrný recall pro Mask R-CNN je 0.113 a pro SSD 0.08, hodnoty jsou pravděpodobně nízké, protože pro některé třídy model nic nedetekoval. Hodnoty naznačují, že by Mask R-CNN mělo dosáhnou lepších výsledků, je ale možné, že došlo ke stejnému problému jako u validační množiny a model detekoval velké množství špatných rámečků (false positive) a bude mít nízké precision. Úlohu segmentace jsme prováděli pouze pro Mask R-CNN a není možné jí porovnat s jiným trénovaným modelem, pouze s výsledky z minulého roku. Nejlepší výsledek pro segmentaci z roku 2019 byl 4.2% mAP. Nutno dodat, že řešení úlohy segmentace se zúčastnil pouze jeden tým. Datum finálního vyhodnocení soutěže je bohužel až po odevzdání této práce, proto není možné připojit konečné výsledky.



Obrázek 22: Porovnání výsledků SSD a Mask R-CNN, pro obrázky z validační množiny.



Obrázek 23: Porovnání výsledků SSD a Mask R-CNN, pro obrázky z validační množiny.

5 Závěr

Cílem této bakalářské práce bylo seznámit se s metodami detekce objektů v obraze a vybrané modely otestovat. V úvodu jsme krátce představili několik úloh počítačového vidění, mezi které patří úloha detekce objektů, která je předmětem této práce. V další kapitole byly představeny důležité datasety, které jsou často používány v úloze detekce objektů a možnost jejich rozšíření pomocí augmentací. V této části byl také představen relativně malý dataset CoralClef 2020, který byl publikován v rámci probíhající soutěže. Dataset obsahuje data pro lokalizaci a segmentaci korálů. Tento dataset byl dále použit pro trénování a testování vybraných modelů.

V třetí části byly popsány modely pro detekci objektů, přičemž byly rozděleny na jednofázové a dvoufázové. U dvoufázových modelů byl popsán vývoj od R-CNN, prvního dvoufázového modelu s rychlostí detekce jednoho snímku za několik vteřin až k Mask R-CNN, modelu který dokáže kromě detekce rámečků provádět i segmentaci a to pro několik snímků za vteřinu. Hlavní výhodou jednofázových modelů je jejich rychlost. Nejen při detekci, ale i při jejich trénování. Byly popsány modely YOLO a SSD. SSD dokáže provádět detekci v reálném čase a přesností výsledků se vyrovná dvoufázovým modelům. Pro detekci na vybraném datasetu byl vybrán z každé skupiny jeden model, pro dvoufázové Mask R-CNN a pro jednofázové SSD.

V poslední části byl popsán postup trénování a porovnání dosažených výsledků jednotlivých modelů. V obou případech mělo na trénování pozitivní vliv použití augmentací. Nejlepší výsledek pro SSD byl mAP 14.94% a pro Mask R-CNN mAP 10.18% pro validační množinu. Oba modely měly problém naučit se detekovat objekty tříd, které se v datasetu nacházely v malém počtu. Výsledky pro testovací množinu byly u obou modelů lepší, SSD dosáhlo mAP 49.0% a Mask R-CNN mAP 24.3% v úloze lokalizace. Úloha segmentace byla vedlejší, protože výsledek nebylo možné porovnat, vzhledem k tomu, že řešení této úlohy umožňuje pouze metoda Mask R-CNN. Výsledek pro testovací množinu pro úlohu segmentace byl mAP 30.4 %.

Ke zlepšení výsledků by pravděpodobně pomohlo větší množství kvalitních obrázků. Především obrázků, které obsahují objekty tříd, které jsou v datasetu zastoupeny v malém počtu.

Appendix

A. Použité kódy

- Mask_RCNN - použitá implementace Mask R-CNN
- SSD - použitá implementace SSD
- mAP - kód použitý pro evaluaci
- Ostatní - Ostatní použité kódy (rozdělení dat, augmentace)

Složky s SSD a Mask R-CNN obsahují kód pro trénování, testování a evaluaci modelu. Všechny kódy se nacházejí v repozitáři na adrese: https://github.com/strakaj/CoralClef2020

B. Natrénované modely

- SSD - ssd512_coralclef2020_epoch-200.h5
- Mask R-CNN - mask_rcnn_coralclef2020_epoch-083.h5

Natrénované modely na datasetu CoralClef 2020.
https://drive.google.com/drive/folders/1JsdcgucPERQv3P-X4xb4LrnCmQbViaNe?usp=sharing

Reference

[1] Waleed Abdulla. Mask r-cnn for object detection and instance segmentation on keras and tensorflow. https://github.com/matterport/Mask_RCNN, 2017.

[2] Jon Chamberlain, Antonio Campello, Jessica P. Wright, Louis G. Clift, Adrian Clark, and Alba García Seco de Herrera. Overview of ImageCLEFcoral 2019 task. In *CLEF2019 Working Notes*, volume 2380 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019.

[3] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

[5] Pierluigi Ferrari. Ssd: Single-shot multibox detector implementation in keras. https://github.com/pierluigiferrari/ssd_keras, 2018.

[6] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[7] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[11] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[12] Fei-Fe Li, Justin Johnson, and Serena Yeung. Lecture 12: Detection and segmentation. http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture12.pdf, 2019.

[13] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014.

[14] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. *CoRR*, abs/1512.02325, 2015.

[15] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[16] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[17] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[18] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[19] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. IEEE, 2001.