VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ANALÝZA ROZLOŽENÍ TEXTU V HISTORICKÝCH DOKUMENTECH

TEXT LAYOUT ANALYSIS IN HISTORICAL DOCUMENTS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. BIANCA PALACKOVÁ**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. OLDŘICH KODYM**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2020/2021

Zadání diplomové práce

Studentka: **Palacková Bianca, Bc.**
Program: Informační technologie
Obor: Počítačové vidění
Název: **Analýza rozložení textu v historických dokumentech**
Text Layout Analysis in Historical Documents
Kategorie: Zpracování obrazu
Zadání:

1. Proveďte literární rešerši metod detekce textových regionů v dokumentech se složitým rozložením.
2. Identifikujte nebo připravte vhodný dataset pro experimenty s detekcí textových regionů.
3. Vyberte nebo navrhněte vhodnou metodu pro řešení této úlohy.
4. Implementujte navrženou metodu metodu a proveďte experimenty nad datovou sadou.
5. Porovnejte dosažené výsledky a diskutujte možnosti budoucího vývoje.
6. Vytvořte stručný plakát nebo video prezentující vaši práci, její cíle a výsledky.

Literatura:

- Galal M. Binmakhashen and Sabri A. Mahmoud. "Document Layout Analysis: A Comprehensive Survey". ACM Comput. Surv. (2020)
- C. Clausner, A. Antonacopoulos, S. Pletschacher. "ICDAR2019 Competition on Recognition of Documents with Complex Layouts - RDCL2019." ICDAR (2019)

Při obhajobě semestrální části projektu je požadováno:

- Body 1 až 3.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/
Vedoucí práce: **Kodym Oldřich, Ing.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 19. května 2021
Datum schválení: 30. října 2020

Abstrakt

Cieľom tejto diplomovej práce je navrhnúť a implementovať algoritmus na analýzu rozloženia textu v historických dokumentoch. Pri riešení tohto problému bola využitá neurónová sieť, konkrétne architektúra Faster-RCNN. Na trénovanie a otestovanie algortimu bol využitý dataset so 6 135 obrázkami dobových novín. V rámci práce boli natrénované 4 modely neurónových sietí: model na detekciu slov, nadpisov, textových regiónov a model detekujúci slová na základe ich polohy v riadku. Výstupy z týchto sietí boli vhodne spracované, s cieľom detekovať rozloženie textu na vstupnom obrázku. Na evaluáciu bola použitá upravená metrika F-score, na základe ktorej algoritmus dosiahol presnosť takmer 80 %.

Abstract

The goal of this thesis is to design and implement algorithm for text layout analysis in historical documents. Neural network was used to solve this problem, specifically architecture Faster-RCNN. Dataset of 6 135 images with historical newspaper was used for training and testing. For purpose of the thesis four models of neural networks were trained: model for detection of words, headings, text regions and model for words detection based on position in line. Outputs from these models were processed in order to determine text layout in input image. A modified F-score metric was used for the evaluation. Based on this metric, the algorithm reached an accuracy almost 80 %.

Kľúčové slová

analýza rozloženia textu v dokumentoch, neurónové siete, Faster-RCNN, Python, spracovanie obrazu

Keywords

document layout analysis, neural networks, Faster-RCNN, Python, image processing

Citácia

PALACKOVÁ, Bianca. *Analýza rozložení textu v historických dokumentech.* Brno, 2021. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Oldřich Kodym

Analýza rozložení textu v historických dokumentech

Prehlásenie

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně pod vedením pana Ing. Oldřicha Kodyma. Uvedla jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpala.

.....
Bianca Palacková
17. mája 2021

Poďakovanie

Týmto by som sa chcela poďakovať vedúcemu mojej práce Ing. Oldřichovi Kodymovi za je odborné rady, podporu a správne nasmerovanie pri riešení tejto práce kedykoľvek to bolo potrebné. Ďakujem taktiež organizácii MetaCentrum za poskytnutie výpočetných zdrojov potrebných pri trénovaní neurónových sietí. V neposlednom rade by som sa chcela poďakovať priateľovi, rodine a kamarátom za podporu pri štúdiu a písaní diplomovej práce.

Obsah

Kapitola 1

Úvod

Digitalizácia je globálny trend, ktorý sa nevyhol ani knihovníctvu. Digitalizácia v knihovníctve je transformácia kníh a dokumentov do počítačových súborov. Týmto spôsobom sa spracúvajú najmä historické a archívne knihy, staré periodiká a iné dokumenty, ktorých časté fyzické používanie by mohlo prispieť k ich poškodeniu. Konverzia takýchto dokumentov do ich digitálnej podoby prináša niekoľko podstatných výhod:

- vyššia efektívnosť práce s dokumentom v zmysle možnosti vyhľadávania textových reťazcov alebo vytvárania kópii,
- rýchlejší a pohodlnejší prístup k dokumentom, ktoré si je možné jednoducho prehliadať aj z domu,
- viacnásobné využívanie jedného dokumentu v tom istom čase,
- dlhšia životnosť fyzických dokumentov.

Analýza layoutu dokumentu, ktorej sa v tejto práci venujem, je dôležitým krokom pri digitalizácii a je to predspracovanie dokumentov pred ďalšími procesmi, ktoré z dokumentu extrahujú informácie. Je to krok zodpovedný za detekovanie a anotáciu geometrickej štruktúry dokumentu. Jedná sa o textové bloky, nadpisy, obrázky, tabuľky a ďalšie. Cieľom analýzy layoutu je zjednodušiť následné fázy analýzy dokumentu, ako sú OCR, porozumenie textu alebo obrázkov, určenie poradia čítania dokumentu atď. Takáto analýza dokumentu má niekoľko dôležitých aplikácii, spomenúť môžeme napríklad vyhľadávanie dokumentov podľa obsahu, automatická kategorizácia dokumentov alebo vyhľadávanie textových reťazcov v dokumente.

Postup analýzy layoutu pozostáva z niekoľkých fáz, ktoré sa môžu medzi metódami líšiť, v závislosti od rozloženia dokumentov a cieľu konečnej analýzy. V súčastnosti ešte nie je vyvinutý univerzálny algortimus na analýzu dokumentov, ktorý vyhovuje všetkým typom rozložení alebo ktorý spĺňa všetky ciele analýzy.

V tejto práci sa zaoberám detekovaním textových regiónov a nadpisov v obrázkoch dokumentov s rozličným rozdelením layoutu. Nezaoberám sa detekciou ďalších komponentov dokumentu ako sú napríklad obrázky, tabuľky alebo ornamenty. Pri detekcii regiónov a nadpisov som sa zamerala na neurónové siete. Ide o technológiu, ktorá je v súčastnosti široko používaná a dosahuje veľmi dobré výsledky.

Ciele tejto práce teda sú:

- Preštudovať problematiku detekcie textových regiónov v historických dokumentoch so zložitým rozdelím.

- Navrhnúť a implementovať algoritmus na detekciu textových regiónov.
- Program vhodným spôsobom otestovať a zhrnúť dosiahnuté výsledky.

Práca je rozdelená do ôsmich kapitol, pričom kapitola 2 vysvetľuje pojem analýza layoutu a zhrňuje existujúce metódy k detekcii textových regiónov. Kapitola 3 zhrňuje teoretické znalosti z oblasti neurónových sietí a porovnáva súčasné detekčné siete. V kapitole 4 sú popísané existujúce dátové sady, ktoré obsahujú obrázky dokumentov. Kapitola 5 popisuje výsledný návrh riešenia. Presný popis experimentov a ich implementácia je popísaná v kapitole 6. Kapitola 7 popisuje dosiahnuté výsledky.

Kapitola 2

Analýza layoutu dokumentu

2.1 Čo je to analýza layoutu

Analýza layoutu dokumentu je proces rozdelenia a kategorizácie homogénnych regiónov obrázku dokumentu. Obrázky dokumentov sú často získané digitalizáciou fyzických dokumentov a to skenerom alebo digitálnym fotoaparátom. Mnoho dokumentov, napríklad noviny, časopisy, brožúry a historické texty obsahujú veľmi komplexné a zložité usporiadanie layoutu a to z dôvodu rozličného umiestnenia obrázkov, titulkov, odsekov, tabuliek, zložitého pozadia, formátovania umeleckého textu atď. Príklady takýchto dokumentov je možné vidieť na obrázku 2.1. [36]

Človek používa rôzne podnety, ako napríklad kontext, konvencie a informácie o jazyku, spolu s komplexným procesom uvažovania na pochopenie obsahu dokumentu. Zatiaľ čo analýzu dokumentov s jednoduchou štruktúrou môžeme považovať za vyriešenú, komplexné rozloženia ako môžeme nájsť v novinách alebo technických článkoch predstavujú ešte stále výzvu. Tento problém je veľmi aktuálny vzhľadom na rastúci počet digitalizácie dokumentov knižnicami po celom svete. Aj napriek pokroku v posledných rokoch v tejto oblasti ostáva automatická analýza dokumentov so zložitým usporiadaním náročnou a otvorenou úlohou.

Obr. 2.1: Dokumenty so zložitým layoutom. Prevzaté z [38] a [21]

Automatická analýza rozloženia dokumentu je dôležitým krokom v kognitívnych procesoch, ktoré extrahujú informácie z obrázkov dokumentu, ide napríklad o určenie textových a netextových regiónov pred tým ako na nich bude aplikované OCR (Optical Character Re-

cognition), usporiadanie textových častí podľa poradia čítania, porozumenie grafom a obrázkom, extrakcia štruktúrovaných údajov z tabuliek a iné.

Obsah dokumentov môže mať rôznu štruktúru. Profesor Koichi Kise ich rozdelil do nasledujúcich kategórii: pravidelné, Manhattan, non-Manhattan, viacstĺpcový Manhattan, horizontálne a diagonálne prekrývajúce [29]. Túto kategorizáciu je možné rozšíriť na historické rukopisy s ľubovoľným usporiadaním.

Pravidelné usporiadanie (obr. 2.2a) je charakterizované jedným obdĺžnikovitým regiónom textu. Dokumenty typu Manhattan (obr. 2.2b) majú viacero oddelených odstavcov, znovu zarovanných do obdĺžnikového tvaru. Takéto rozloženie môžeme nájsť napríklad v technickým článkoch a magazínoch. Non-Manhattan rozloženie (obr. 2.2c) sú typické odstavcami, ktoré nemajú odĺžníkové zarovnanie. Prekrývajúce sa rozloženia (obr. 2.2f) majú niektoré prvky dokumentu, napríklad texty, ktoré prekrývajú ďalšie prvky dokumentu. V skutočnosti môžu existovať ďalšie typy rozdelenia textu, avšak týchto šesť považujeme za najbežnejšie. [7]

(a) (b) (c)

(d) (e) (f)

Obr. 2.2: Typy layoutov: (a) pravidelné, (b) Manhattan, (c) non-Manhattan, (d) viacstĺpcový Manhattan, (e) ľubovoľné usporiadanie (f) horizontálne a diagonálne prekrývajúce sa. [7]

2.2 Prehľad prístupov

Prístupy ku analýze layoutu môžeme zaradiť do 3 skupín: prístupy zhora nadol, prístupy zdola nahor a hybridné prístupy. **Prístupy zdola nahor** (bottom-up) začínajú analýzu layoutu od malých elementov dokumentu ako sú pixely alebo spojené komponenty. Homogénne elementy sú potom spájané do väčších celkov. Spájanie regiónov pokračuje pokiaľ sa nedosiahne preddefinovaných ukončujúcich podmienok. **Stratégie zhora nadol** (top-down) začínajú od veľkých regiónov dokumentu, ako je napríklad samotný dokument. Potom je tento veľký región rozdelený na menšie časti ako sú odseky na základe určitých pravidiel. Tento prístup pokračuje v rozdeľovaný pokiaľ už nie sú regióny na delenie alebo pokiaľ ne-

dosiahne ukončovacej podmienky. Kombinácia oboch týchto prístupov (zhora nadol a zdola nahor) vytvára takzvaný **hybridný prístup**. [7]

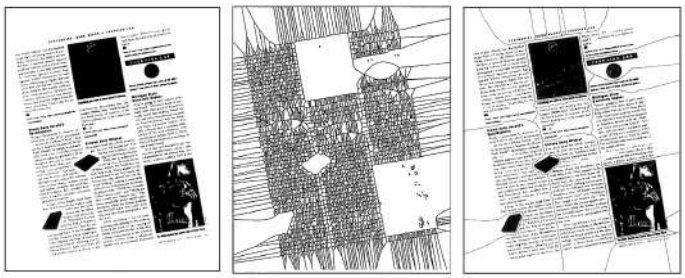
Pístupy zdola nahor

Prístupy pracujúce zdola nahor môžme ďalej rozdeliť podľa technológie ktorú využívajú. Konkrétne ide o analýzu spojených komponentov, textúr, analýzu založenú na učení, Voronoive diagramy a Delaunayho trianguláciu. [7]

Jeden z prvých úspešných bottom-up algoritmov založených na **analýze spojených komponentov** bol Docstrum [39]. Docstrum zhlukuje spojené komponenty algoritmom k-najbližších susedov. Algorimus bol úspešný na rôznych rozloženiach layoutu, bol však testovaný len na tlačených dokumentoch. Pri spracovaní historických ručne písaných dokumentoch boli využité lokálne príznaky spojených komponentov [8]. Tran et al. [51] navrhol metódu iteratívnej klasifikácie, ktorá delí spojené komponenty do štyroch tried, a to textu, obrázkov, oddelovače a šum.

Prístupy využívajúce **analýzu textúry** zvyčajne začínajú extrakciou rysov textúry priamo z pixelov obrázku. Potom sú tieto rysy použité na tvorbu homogénnych regiónov. Takáto analýza textúry môže byť kombinovaná s autokoreláciou [27] alebo Gaborovým filtrom [35].

Metódy využívajúce **analýzu layoutu založenú na učení** dosahujú dobrých výsledkov na rôznych rozloženiach dokumentu, vrátane tých komplexných. Väčšinou však tiež potrebujú určitý post-processing ako napríklad zhlukovanie alebo použitie morfologických operácii. Tieto prístupy je možné ďalej rozdeliť na plytké (non-deep), ktoré na extrakciu príznakov využívajú napríklad SIFT (Scale Invariant Feature Transform), a na hlboké (deep), ktoré na extrakciu príznakov využívajú konvolučnú neurónovú sieť. Grüning et al. [23] využili ARU-Net, čo je rozšírenie siete U-Net, na segmentáciu obrázku. Na lokalizáciu riadkov potom využili zhlukovanie nad výstupom siete ARU-Net. Chen et al. [9] využili taktiež segmentáciu obrázku, ich sieť však bola oveľa jednoduchšia. Mala len jednu konvolučnú vrstvu so štyrmi konvolučnými jadrami a netypicky pre CNN, neobsahovala žiadnu pooling vrstvu. Ich výsledky však boli porovnateľné s inými metódami.



Obr. 2.3: Použitie Voronoi diagramu na nájdenie regiónov. [30]

Obr. 2.4: Delaunay triangulácia na centroidoch spojených komponentov. [53]

Segmentácia ľubovoľného rozloženia dokumentu je náročná úloha, pretože takéto rozloženia väčšinou nemajú žiadne konkrétne tvary. Riešením takého problému môže byť **Voro-**

noi diagram, ktorý dokáže definovať hranice okolo ľubovoľných regiónov. Kise [30] vo svojej práci hľadal Voronoi hrany diagramu, vytvoreného nad spojenými komponentami, ktoré oddeľujú jednotlivé regióny textu v obrázku.

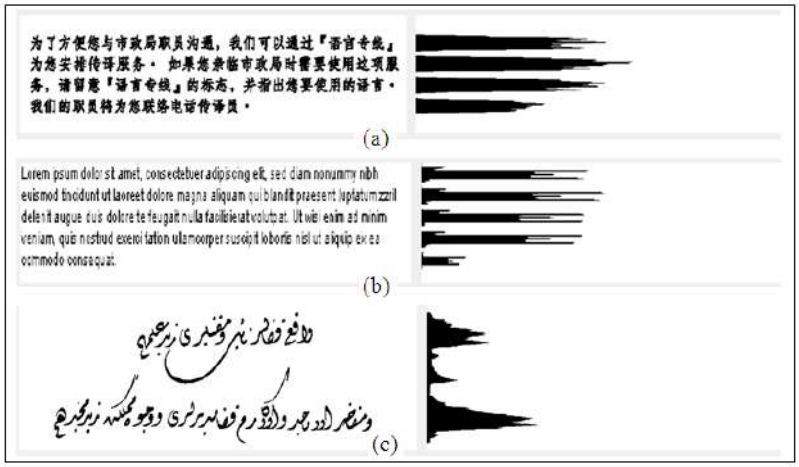
Delaunay triangulácia bola úspešne použitá napríklad Xiaom a Yanom [53], ktorí najprv našli centroidy spojených komponentov. Nad nimi potom zostrojili Delaunay triangluláciu. Pri hľadaní textových regiónov sa následne riadili pravidlami ako: trojuholníky v textových regiónoch majú kratšie hrany ako trojuholníky v pozadí, trojuholníky v textovej časti majú najkratšie hrany v smere riadku a dlhšie hrany spájajú dva susedné riadky a podobne. Platnosť týchto pravidiel je možné vidieť na obrázku 2.4.

Pístupy zhora nadol

Prístupy zhora nadol môžeme rozdeliť na štyri kategórie a to: prístupy využívajúce analýzu textúr, Run Length Smearing Algorithm (RLSA), projekciu profilu a analýzu medzier. [7]

Príkladom metód ktoré využívajú **analýzu textúr** je napríklad metóda od Saabni a El-Sana [48] na detekciu riadkov textu. V nej autori predpokladajú že textové regióny sú tmavšie ako pozadie. Pomocou dištančnej transformácie sú vygenerované energetické mapy, kde vysoké hodnoty označujú rozloženie riadkov textu. Charakteristiky obrázku môžu byť nájdené aj filtrovaním. Cohen et al. [12] využívajú šesť Laplacian of Gaussian (LoG) filtrov. Po filtrácii sú analyzované maximálne odozvy na nájdenie textových regiónov.

RLSA predpokladá na vstupe binárny obrázok. Jednotlivé pixely majú teda hodnoty 0 (pozadie) alebo 1 (popredie). Algoritmus zmení hodnotu 0 na hodnotu 1 ak počet pixelov s hodnotou 0 medzi dvomi pixelmi s hodnotou 1 je menej ako preddefinovaná hodnota. Prvý krát tento algoritmus predstavil Wahl et al. [52] na detekciu riadkov v dokumentoch s jednoduchým rozložením. Neskôr bol upravený Shiom a Govindarajuo [49], ktorí vykonávali scan v horizontálnom aj vertikálnom smere. RLSA je robustná a jednoduchá metóda na aplikáciu, avšak je veľmi citlivá na zvolený prah, ktorý musí byť dôkladne zvolený.



Obr. 2.5: Projekcie profilu textov v rôznych jazykoch. Obrázok prevzatý z [6]

Projekcia profilu sa rovnako ako RLSA vykonáva na binárnom obrázku. Ide o horizontálne alebo vertikálne sčítanie počtu pixelov, ktoré patria ku popredíu (obrázok 2.5). Nagy et al. na základe projekcie profilu vytvorili X-Y cut algoritmus, ktorý určuje miesta kde dokument rozdeliť na bloky. Tento algoritmus najviac vyhovuje dokumentom, ktoré

majú fixnú dĺžku medzier textových regiónov. X-Y cut sa dočkal aj niekoľkých modifikácii, napríklad v [24] bol vylepšený, tak aby pracoval na projekcii profilu bounding boxov.

Analýza medzier predpokladá že medzi jednotlivými textovými regiónmi sú biele miesta, ktoré ich oddeľujú. Tieto metódy vyhovujú dokumentom s jednoduchým rozložením so zreteľnými medzerami. Analýza medzier je využívaná napríklad v [5], kde sú v dokumente nájdené biele oblasti. Dokument je rozdelený na regióny, ktoré sú potom klasifikované na základe počtu oblastí.

Zhrnutie

Pri analýze layoutu dokumentov sú najčastejšie používané stratégie zdola nahor. Tieto prístupy väčšinou ako základný stavebný prvok využívajú spojené komponenty namiesto pixelov. Pozitívom týchto metód sú dobré výsledky aj na komplexnejších layoutoch, môžu však mať väčšie pamäťové a časové nároky. Podľa štúdie [7], kde bolo analyzovaných 79 prístupov, bolo 61% z nich zdola nahor, 33% zhora nadol a 6% hybridných. Prístupy zhora nadol sú vhodnejšie na použitie na štandartných rozloženiach layoutu, ako je napríklad Manhattan. [7]

V tejto práci experimentujem s obidvoma prístupmi, pričom obidve sú implementované pomocou CNN. Experimenty sú potom vyhodnotené a porovnané ich výhody a nevýhody. Z experimentov je potom vyskladaný finálny postup analýzy layoutu, ktorý bol vybratý s ohľadom na presnosť, ale aj čas výpočtu.

Kapitola 3

Detekcia pomocou neurónových sietí

Neurónové siete sú v súčasnosti hojne používané na pomerne širokú škálu problémov a vo vybraných úlohách dokážu priniesť výsledky s veľmi vysokou presnosťou. Využívajú sa napríklad v oblasti počítačového videnia, spracovania zvuku či textu.

Výskum umelých neurónových sietí započal už v 40. rokoch 20. storočia. Teória neurónových sietí vychádza z neurofyziologických poznatkov. Snaží sa vysvetliť správanie sa na princípe spracovania informácií v nervových bunkách. Niekedy sa umelé neurónové siete označujú aj ako modely mozgu bez mysle, keďže sa snažia pochopiť nervový systém, ale nezaoberajú sa psychikou. Informácie sa v nervovom systéme prenášajú vo forme zmien membránového potenciálu nervových buniek - neurónov. Práve biologické neuróny sa stali vzorom pre umelé (alebo taktiež formálne) neuróny. [32]

3.1 Formálny neurón

Formálny neurón je základnou jednotkou matematického modelu neurónovej siete získaný preformulovaním zjednodušenej funkcie fyziologického neurónu do matematického jazyka. Štruktúru formálneho neurónu popisuje obrázok 3.1. Neurón má n reálnych vstupov $x_1 ... x_n$, pričom každý z nich je ohodnotený synaptickou váhou $w_1 ... w_n$, ktoré udávajú dôležitosť daného vstupu. Vstup s väčšou synaptickou váhou ovplyvňuje neurón viac. Nastavovaním synaptických váh jednotlivých vstupov neurónov dochádza k učeniu siete. [55]

Vnútorný potenciál neurónu predstavuje vážená suma vstupných hodnôt. Matematicky môžeme vstup a jeho odpovedajúce váhy zapísať ako vektory $x = (x_1, x_2, ..., x_n)$ a $w = (w_1, w_2, ..., w_n)$. Vnútorný potenciál neurónu sa potom vypočíta ako ich skalárny súčin:

$$\xi = \sum_{i=1}^{n} w_i x_i \tag{3.1}$$

Výstup neurónu y sa vypočíta pomocou vnútorného potenciálu a aktivačnej funkcie σ:

$$y = \sigma(\xi) \tag{3.2}$$

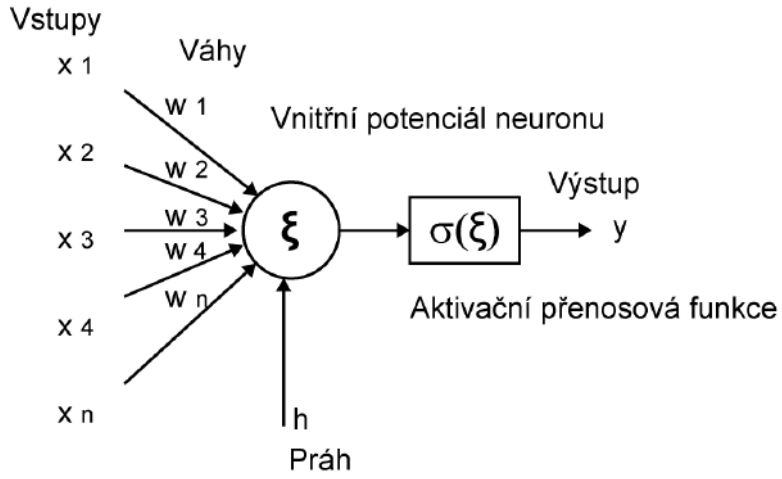
Aktivačná funkcia môže byť realizovaná ako porovnanie potenciálu neurónu s prahovou hodnotou (označovanou tiež ako bias). Ak je hodnota potenciálu vyššia ako prah, neurón vygeneruje signál, v opačnom prípade signál nevygeneruje. Najjednoduchším typom aktivačnej funkcie je tzv. ostrá nelinearita, ktorá má tvar:

$$\sigma(\xi) = \begin{cases} 1, & \text{ak } \xi \geq h \\ 0, & \text{ak } \xi < h \end{cases} \tag{3.3}$$

Formálnou úpravou je možné docieliť toho, že funkcia σ bude mať nulový prah a vlastný prah neurónu so záporným znamienkom budeme chápať ako váhu $w_0 = -h$ ďalšieho formálneho vstupu $x_0 = 1$ s konštantnou jednotkovou hodnotou. Matematická formulácia funkcie neurónu je potom daná vzťahom:

$$\sigma(\xi) = \begin{cases} 1, & \text{ak } \xi \geq 0 \\ 0, & \text{ak } \xi < 0 \end{cases}, \text{ kde } \xi = \sum_{i=0}^{n} w_i x_i \tag{3.4}$$

Takýto formálny neurón, ktorý doplňuje vstupný vektor o nultú zložku, ktorá má pevnú hodnotu 1 a ktorej váha je daná zápornou hodnotou prahu sa nazýva perceptron. [55]



Obr. 3.1: Formálny neurón. Obrázok prevzatý z [1].

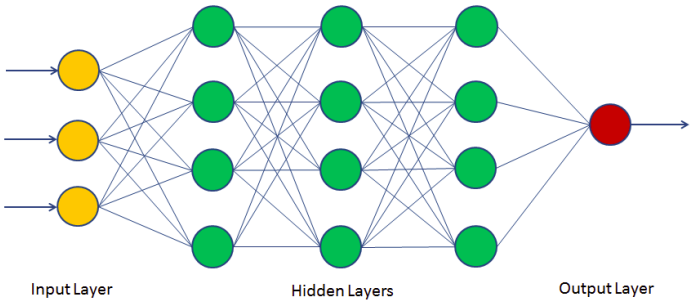
Ďalšími typmi aktivačných funkcii sú napríklad:

- hyperbolický tangens
$$\sigma(\xi) = \frac{1 - e^{-\xi}}{1 + e^{-\xi}} \tag{3.5}$$
- sigmoid
$$\sigma(\xi) = \frac{1}{1 + e^{-\xi}} \tag{3.6}$$
- ReLU
$$\sigma(\xi) = \begin{cases} 0, & \text{ak } \xi < 0 \\ \xi, & \text{ak } \xi \geq 0 \end{cases} \tag{3.7}$$

3.2 Neurónové siete

Neuróny sú zhlukované do vrstiev a ich prepájaním medzi vrstvami sa vytvárajú komplexnejšie modely, ktoré sa nazývajú umelé neurónové siete. To, aké a koľko vrstiev je použitých,

koľko neurónov obsahujú jednotlivé vrstvy a ako sú vrstvy poprepájané, sa nazýva architektúra siete. Typicky je možné vrstvy rozdeliť na vstupnú, skryté a výstupnú.

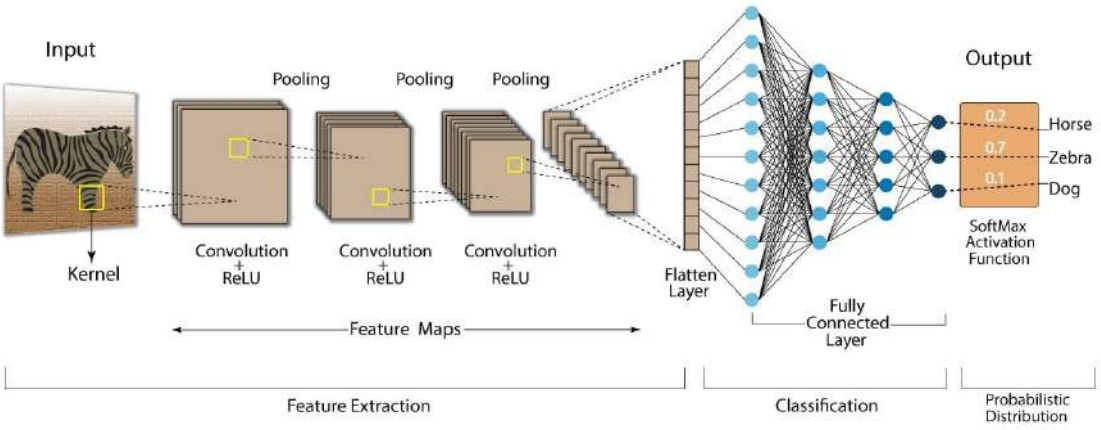


Obr. 3.2: Schéma jednoduchej doprednej siete s 3 skrytými vrstvami. [37]

Neurónové siete pracujú na princípe dopredného šírenia informácii. Vo vstupnej vrstve sa neurónom predajú vstupné dáta. Následne sa vypočítajú výstupné hodnoty všetkých neurónov z danej vrstvy. Tieto výstupy potom putujú do ďalšej vrstvy, kde sa proces opakuje. Jednotlivé vrstvy neurónov sú medzi sebou prepojené. Každé spojenie má svoju váhu. Proces učenia neurónovej siete potom spočíva v správnom nastavení týchto váh. Na začiatku sú všetky váhy nastavené náhodne. K učeniu potrebujeme vektory vstupných dát (trénovacia množina) a k nim očakávaný výstup. Pre viacvrstvovú doprednú sieť sa k učeniu najčastejšie používa algoritmus spätného šírenia chyby (back-propagation).

Základom metódy back-propagation je minimalizácia chyby neurónovej siete. Na vstup siete sa privedie vzorka z trénovacej množiny a vypočíta sa chyba medzi výstupom siete a požadovaným výstupom. Následne je chyba spätne šírená po sieti a upravujú sa váhy jednotlivých neurónov. Jeden priechod trénovacej množiny neurónovou sieťou sa označuje ako epocha.

3.3 Konvolučné neurónové siete



Obr. 3.3: Architektúra konvolučnej neurónovej siete. [42]

Konvolučné neurónové siete (ďalej ako CNN) sú jedným z typov viacvrstvových neurónových sietí. Začiatok vývoja CNN započali Hubel a Wisel už v roku 1959 [26] na základe výskumu

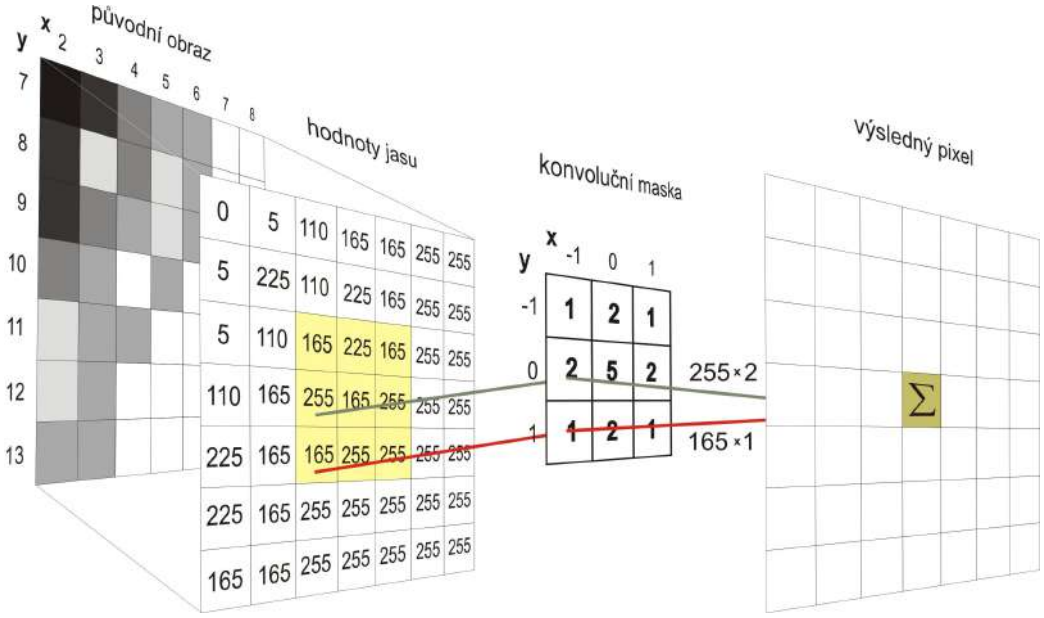
prepojení neurónov v zrakovej kôre mozgu živočíchov. Na základe inšpirácie tohto článku v roku 1980 Kunihiko Fukusima navrhol neocognitron [16]. Táto práca je považovaná za prvý teoretický model pre CNN. V roku 1990, LeCun vytvoril sieť LeNet-5 na rozpoznávanie písaných číslic. V tej dobe bolo však používanie a ďalší vývoj limitované výkonom vtedajšej techniky a nedostatkom dát. V dnešnej dobe už máme k dispozícii výkonné GPU stroje a tak v roku 2012 bola predstavená veľká hlboká CNN, nazývaná AlexNet [31], ktorá predviedla excelentný výkon. Úspech AlexNet tak započal cestu ku vynájdeniu rôznych CNN modelov a ich aplikovaniu v rôznych sférach počítačového videnia a spracovania jazyka. [18]

CNN obsahuje 3 hlavné typy vrstiev:

1. konvolučné vrstvy
2. pooling vrstvy
3. plne prepojené vrstvy.

Konvolučná vrstva

Konvolučná vrstva je prvým hlavným stavebným blokom v CNN. Jej úlohou je detekovať charakteristické črty vstupného obrazu, ako sú napríklad hrany alebo rohy. Táto vrstva aplikuje dané konvolučné jadro postupne na celý obraz. Prevádza pritom operáciu konvolúcie, ktorá odpovedajúce si hodnoty obrazu a jadra najprv vynásobí a potom všetky sčíta.

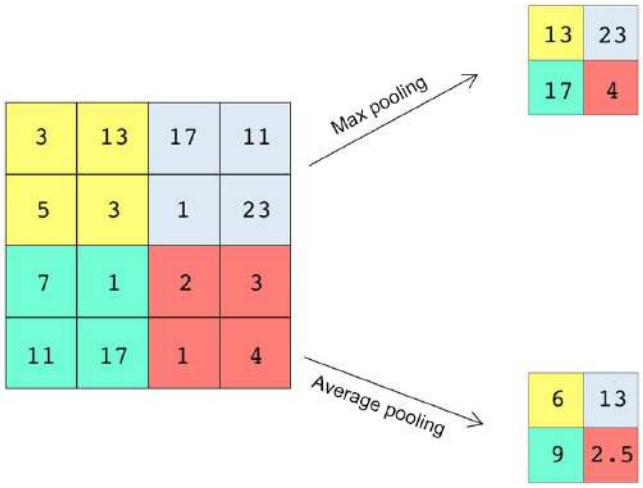


Obr. 3.4: Princíp dvojrozmernej konvolúcie. [22]

Veľkosť jadra je možné si zvoliť, napríklad jadro s veľkosťou 5×5×3. Toto jadro má výšku a šírku 5 a hĺbku 3. Každé jedno z jadier vytvára dvojrozmernú feature mapu. Počet máp postupne stúpa, no ich veľkosť sa zároveň zmenšuje. Voľba veľkosti masky taktiež vplýva na počet parametrov siete. Pred učením siete sú tieto parametre nastavené náhodne. Počas trénovania sa parametre vylepšujú tak, aby bolo jadro schopné extrahovať čo najhodnejšie črty obrázku.

Pooling vrstva

Pooling vrstva redukuje veľkosť dát a tým aj redukuje množstvo parametrov. Sama o sebe žiadne učiace parametre nemá. Najčastejšie sa táto vrstva používa s filtrom 2×2, ktorá vstupné dáta redukuje na štvrtinu. Najpoužívanejším typom poolingu je max pooling, ktorý z danej oblasti vyberá maximálnu hodnotu. Ďalším typom je average pooling, ktorý priemeruje hodnoty v oblasti. Princíp obidvoch popísaných pooling filtrov popisuje obrázok 3.5.



Obr. 3.5: Max a average pooling. [3]

Plne prepojená vrstva

V plne prepojených vrstvách je každý neurón z jednej vrstvy siete prepojený s každým neurónom nasledujúcej vrstvy. Plne prepojené vrstvy nasledujú po konvolučných a pooling vrstvách a starajú sa o klasifikáciu založenú na extrahovaných črtách obrázku. Zatiaľ čo konvolučné a pooling vrstvy využívajú ReLU aktivačné funkcie, plne prepojené vrstvy zvyčajne používaju softmax na vytvorenie pravdepodobnosti danej triedy v intervale od 0 do 1. [42]

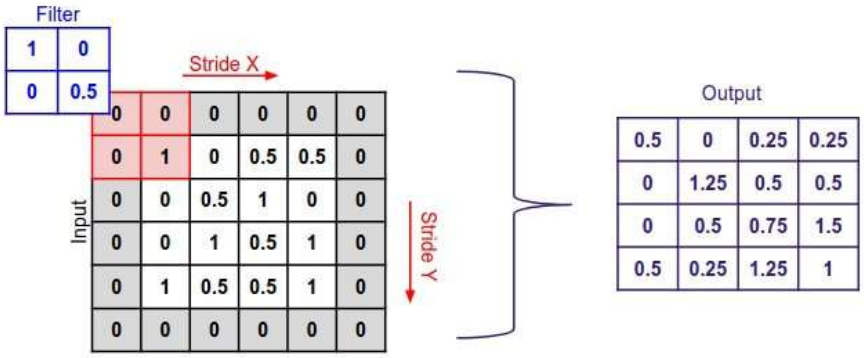
Ďalšie pojmy z oblasti konvolučných sietí

Batch normalizácia je ďalšou technikou na regularizáciu neurónovej siete. V praxi sa používa medzi konvolučnou a aktivačnou vrstvou. Jej úlohou je normalizovať extrémne hodnoty gradientu, ktoré sú príliš nízke alebo príliš vysoké a predísť tak zaseknutiu učenia siete. Vďaka batch normalizácii je možné použiť vyšší learning rate, čo zníži čas učenia siete. [42]

Dropout je technika používaná na regularizáciu, kde sa náhodne zvolené neuróny ignorujú behom trénovacieho procesu. Dropout hodnota je väčšinou nastavená na 0.5 a môže byť prispôsobená tak aby mala sieť čo najlepšie výsledky a trénovaciu rýchlosť. Časť vlastností sa tak zahodí a sieť sa bude učiť na komplexnejších dátach. [42]

Stride je vzdialenosť, alebo počet pixelov, o ktoré sa posúva jadro pri konvolúcii. Čím je toto číslo väčšie, tým menší je výstup, avšak hodnoty väčšie ako 2 sú používané len zriedkavo.

Padding je používaný nato, aby mal výstup rovnakú veľkosť ako vstup. Na základe jeho hodnoty sa okolo obrázku pridá rámček danej šírky s hodnotami pixelov 0.



Obr. 3.6: Konvolúcia, kde hodnoty padding a stride sú rovné 1. [40]

3.4 Súčasné detekčné siete

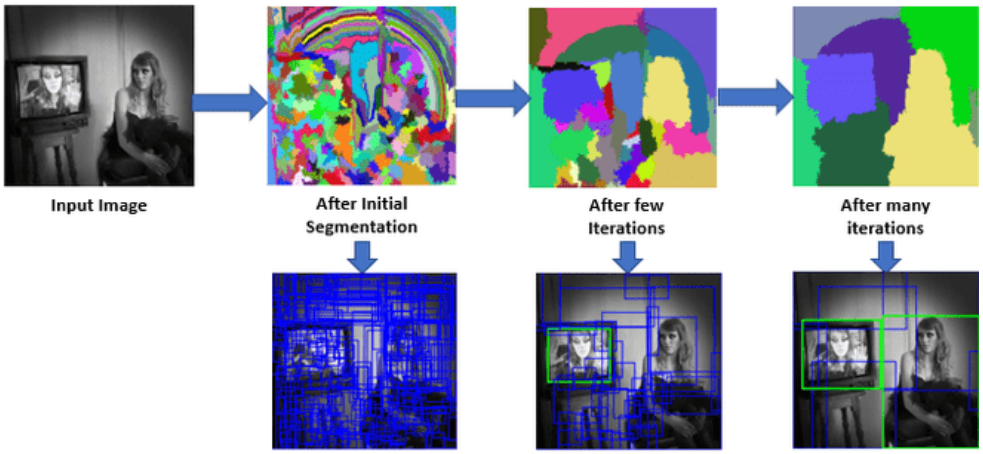
Cieľom detekčných sietí je lokalizovať objekty v obrázku, typicky špecifikovaním osovo zarovnaného ohraničujúceho rámu (angl. bounding box), ktorý je vycentrovaný na stred objektu a tento objekt zároveň aj klasifikovať. Lokalizácia objektu sa berie ako správna ak dostatočne prekrýva bounding box, ktorý bol pre daný objekt označený človekom (ground-truth). Medzi state-of-the-art prístupy patria architektúry YOLO, SSD a Faster-RCNN, ktoré budú podrobnejšie popísané v ďalšej časti.

Rodina modelov RCNN

Do skupiny metód RCNN (Region-based Convolutional Neural Network) patria RCNN, Fast-RCNN a Faster-RCNN, ktoré boli navrhnuté na detekciu objektov v obrázku.

RCNN bola navrhnutá v roku 2014 Rossom Girshickom na univerzite v Berkeley [20]. RCNN bola jednou z prvých architektúr, ktorá na detekciu využívala neurónové siete. Vtedajšie detekčné algoritmy boli totiž založené na SIFT, HOG (Histogram of Oriented Gradients) a Haarových príznakoch.
Štruktúru RCNN je možné rozdeliť na 3 časti: [20]

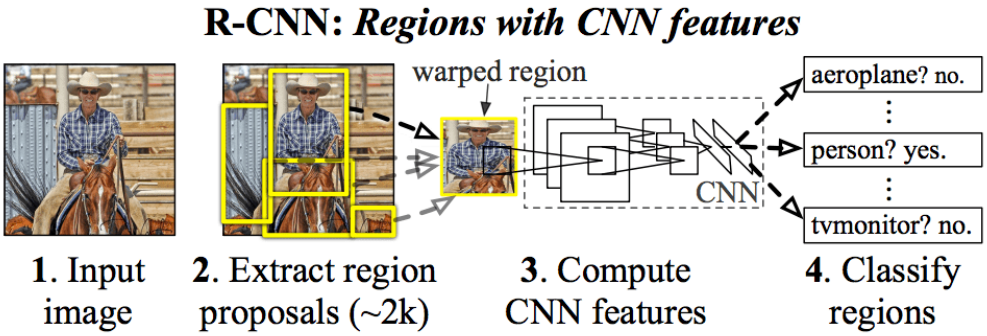
1. **Nájdenie kandidátnych regiónov** - v prvej časti sú z obrázku extrahované výseky, ktoré pravdepodobne obsahujú hľadaný objekt. Tieto regióny je možné hľadať viacerými spôsobmi, využitý však bol selective search. Selective search (obr. 3.7) najprv segmentuje obrázok na malé regióny, ktoré následne iteratívne zoskupuje podľa určitých kritérii. Segmentované oblasti nakoniec použije na vytvorenie približne 2000 kandidátskych regiónov.



Obr. 3.7: Algoritmus selective search. [2]

2. **Extrakcia príznakov** - na každom kandidátnom regióne z predchádzajúcej časti sa extrahujú príznaky pomocou konvolučnej siete. Použitá bola AlexNet, ktorej výstup je vektor o dĺžke 4096 prvkov pre každý región.

3. **Klasifikácia** - v poslednej časti vektory popisujúce výsek vstupujú do SVM klasifikátorov. Na každú triedu je natrénovaný jeden SVM klasifikátor, ktorý rozhodne či sa v danom výseku objekt nachádza.

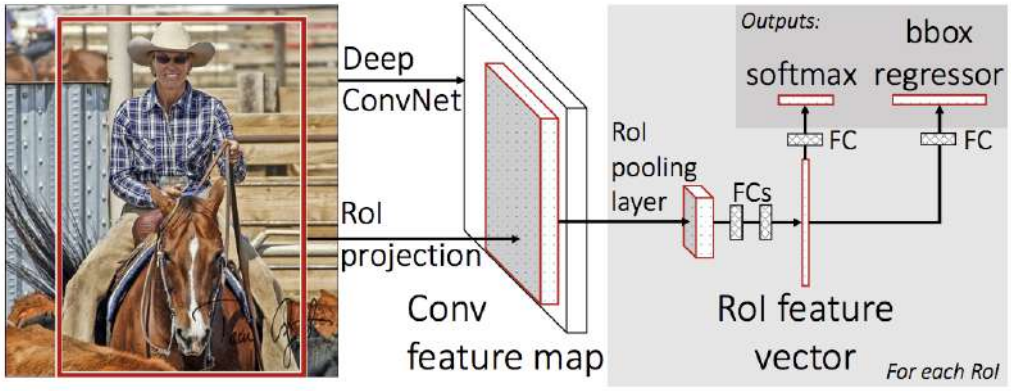


Obr. 3.8: Detekcia modelom RCNN: do RCNN vchádza (1) vstupný obrázok, z ktorého sú (2) extrahované kandidátne regióny. Z regiónov sú (3) extrahované príznaky pomocou CNN a (4) klasifikované pomocou SVM. [20]

Fast-RCNN bolo publikované v roku 2015 ako reakcia na nedostatky RCNN. RCNN síce dosahovalo vysokú presnosť, na druhej strane však malo veľkú priestorovú a časovú náročnosť.

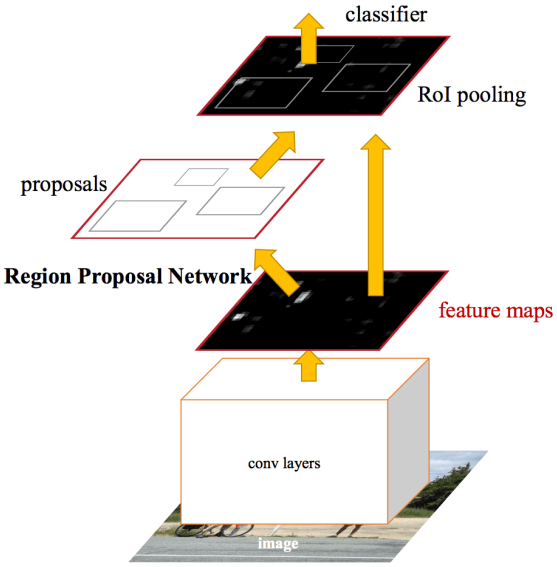
Fast-RCNN znovu využíva selective search na generovanie kandidátnych regiónov. RCNN postupne tieto regióny posielalo do CNN, ktorá pre každý región vygenerovala vektor príznakov. Keďže týchto regiónov bolo až 2000, veľká časť z nich sa prekrývala a teda dochádzalo k opakovaným výpočtom. Fast-RCNN preto do siete posiela výseky spolu s celým obrázkom a zdieľa výpočty konvolúcii výsekov. AlexNet je nahradená sieťou VGG-16, za ktorou sa nachádza vrstva Region of Interest Pooling Layer, ktorá extrahuje príznaky špecifické pre danú kandidátsku oblasť. Fast-RCNN taktiež spája 3 časti RCNN do jednej architektúry. SVM

klasifikátor je nahradený soft-max vrstvou. Sieť má 2 výstupné vektory na každý výrez, a to pravdepodobnosť výskytu objektu z vrstvy softmax a posun bounding boxu z regresoru. [19]



Obr. 3.9: Architektúra Fast-RCNN: Vstupný obrázok a kandidátske regióny (ROI) vstupujú do CNN. Každé ROI je prevedené do mapy príznakov fixnej veľkosti a namapované do vektoru príznakov vďaka dvom plne prepojeným vrstvám. [19]

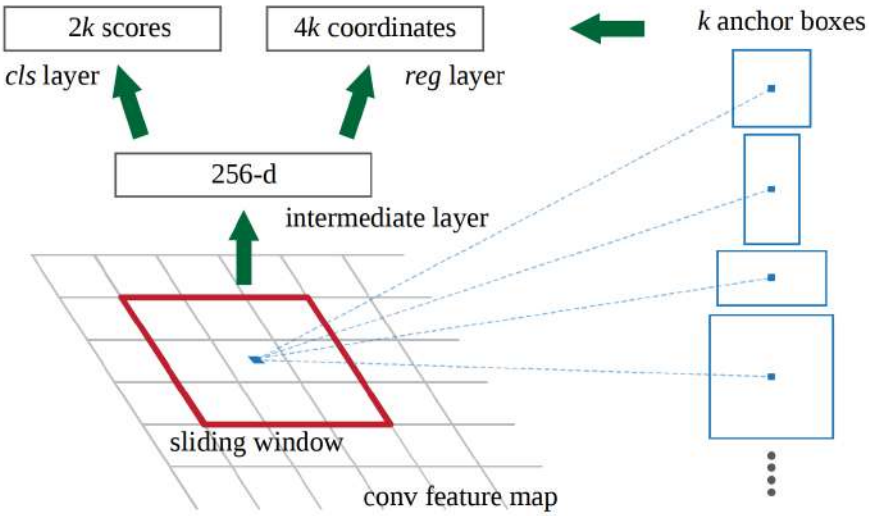
Faster-RCNN je posledným modelom z rodiny RCNN. Bol publikovaný v roku 2016 a ide o vylepšenie modelu Fast-RCNN. Najväčšou slabinou tohto modelu bolo generovanie kandidátnych regiónov pomocou algoritmu selective search, ktorý najviac spomaľoval celú detekciu. Namiesto použitia tohto algoritmu, autori prichádzajú s myšlienkou zdieľania konvolučnej siete aj na generovanie regiónov. [46]



Obr. 3.10: Architektúra Faster-RCNN: konvolučná neurónová sieť vytvorí mapu príznakov, ktorá je predaná RPN. Kandidátne regióny z RPN sú ďalej spracované ako v prípade Fast-RCNN. [46]

Konvolučná neurónová sieť má na vstupe obrázok a na výstupe mapu príznakov. Táto mapa je ďalej vstupom pre Region Proposal Network (RPN). Nad mapou príznakov je posúvané malé okno (autori pracovali s posuvným oknom o rozmere 3×3), ktoré je prevedené na vektor príznakov o veľkosti 256. Tento príznak vchádza do dvoch plne prepojených vrstiev, na obrázku 3.11 môžeme tieto vrstvy vidieť pod označením *cls* a *reg*, pričom *cls* je zodpovedá za určenie pravdepodobnosti výskytu objektu v okne a *reg* za pozíciu regiónu. Nad každej pozícii posuvného okna je predikovaných niekoľko kandidátnych regiónov, nazývaných anchor boxy. Nad každou pozíciou je vytvorených k anchor boxov, kde k je počet pomerov strán boxu vynásobené počtom rôznych veľkostí boxu. Anchor boxy môžeme upravovať na základe riešeného problému. V prípade detekcie slov sa očakáva že budú väčšie na šírku, zatiaľ čo napríklad u detekcie postáv, že budú väčšie na dĺžku. [46]

Vygenerované kandidátne regiónu sú ďalej spracované rovnako ako v prípade Fast-RCNN. Sú vstupom klasifikačnej vrstvy, ktorá určí pravdepodobnosť či sa vo výreze nachádza objekt a regresnej vrstvy, ktorá určí bounding box objektu.



Obr. 3.11: Region proposal network (RPN). [46]

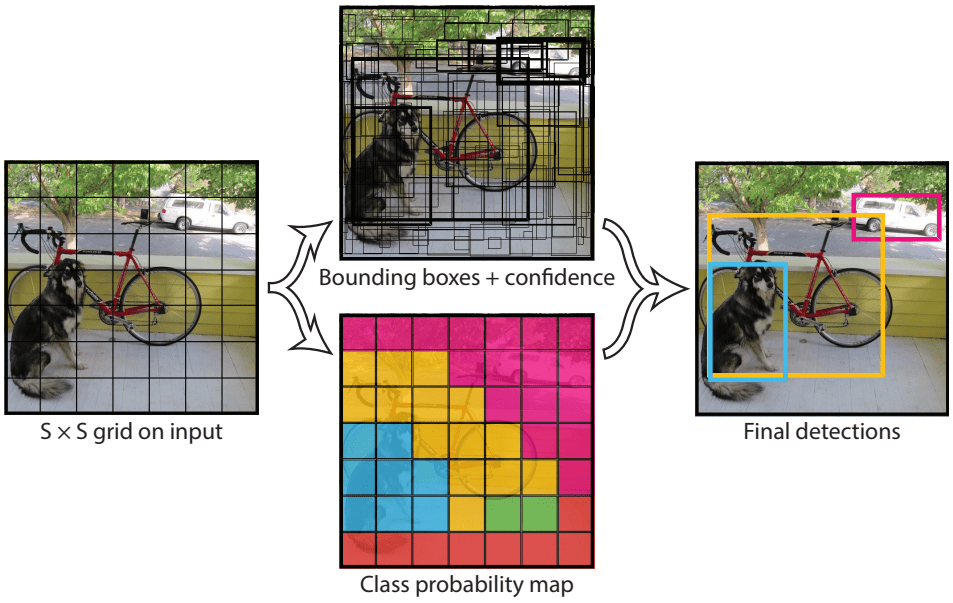
YOLO

Model YOLO (You Only Look Once) bol prvý krát popísaný v roku 2015 v článku [43], ktorého autorom je Joseph Redmon. Tento model je veľmi rýchly, dokáže detekovať objekty v reálnom čase, avšak dosahuje menšej presnosti ako SSD či Faster-RCNN.

Tento prístup zahŕňa len jednu neurónovú sieť, ktorá pracuje naraz so všetkými príznakmi z celého obrázku. Bounding boxy objektov naprieč všektými triedami sú predikované naraz, vďaka čomu je táto sieť veľmi rýchla.

Vstupný obrázok je rozdelený mriežkou s veľkosťou $S \times S$. Každá bunka mriežky je zodpovedná za predikciu objektu, ktorého stred do tejto bunky spadá. Každá bunka predikuje B bounding boxov (bounding box pozostáva zo 4 hodnôt: x, y, w, h) a k nim prislúchajúce confidence skóre, ktoré nám hovorí o tom, na koľko si je sieť istá, že bounding box obsahuje objekt. Spolu s týmito hodnotami bunka taktiež predikuje C pravdepodobností označujúce zaradenie do jednotlivých tried. Pravdepodobnosti tried sa teda nepredikujú na jednotlivé bounding boxy ale na celé bunky mriežky. Výsledné predikcie sú teda zakódované v tenzore o veľkosti $S \times S \times (B * 5 + C)$. [43]

Ak si vezmeme ako príklad obrázok, ktorý rozdelíme na mriežku 7×7, pričom predikujeme 2 bounding boxy na bunku a máme 20 tried, dostaneme tenzor o veľkosti $7 \times 7 \times 30$. Počet bounding boxov pri tomto počte je $7 * 7 * 2 = 98$. Mapa pravdepodobností tried a bounding boxy s ich confidence skórami sú nakoniec skombinované do finálnych predikcii bounding boxov s označením do akej triedy patria. Túto predikciu zobrazuje obrázok 3.12.



Obr. 3.12: Postup predikcie modelom YOLO. [43]

Samotná neurónová sieť je inšpirovaná GoogLeNet modelom [50] na klasifikáciu obrázkov. Sieť sa skladá z 24 konvolučných vrstiev, ktoré sú nasledované 2 plne prepojenými vrstvami.

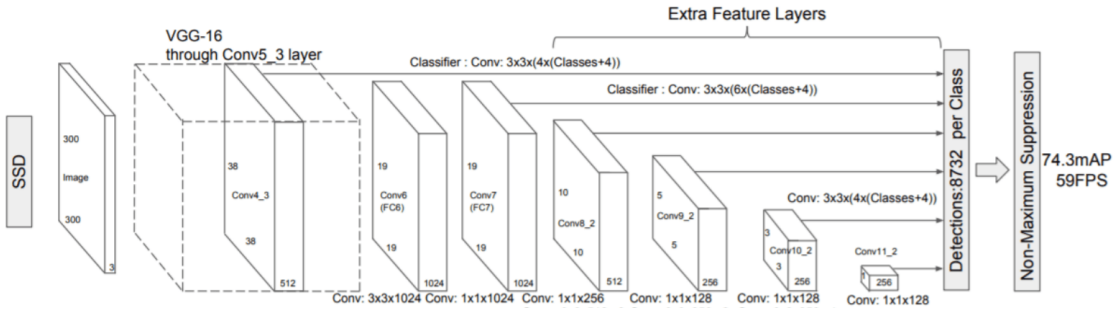
Model YOLO bol ďalej aktualizovaný v roku 2016 na YOLOv2 [44] (alebo aj YOLO 9000) a v roku 2018 na YOLOv3 [45] v snahe vylepšiť jeho výkon. Na modeli a pri trénovaní bolo urobených niekoľko zmien, ako napríklad použitie anchor boxov, batch normalizácie a práca s väčšími vstupnými obrázkami.

SSD

SSD (Single Shot Detector) vznikol v roku 2016 ako ďalší model schopný detekovať objekty v obrázku pomocou jednej neurónovej siete. Tento model bol popísaný v publikácii [34].

SSD je založený na doprednej konvolučnej sieti, ktorá produkuje fixný počet bounding boxov spolu so skóre, ktoré uvádza či sa daná trieda objektov v bounding boxe nachádza. Po konvolučnej sieti nasleduje non-maximum suppression s určitým prahom, ktoré potláča nemaximálne hodnoty a vytvára finálne predikcie.

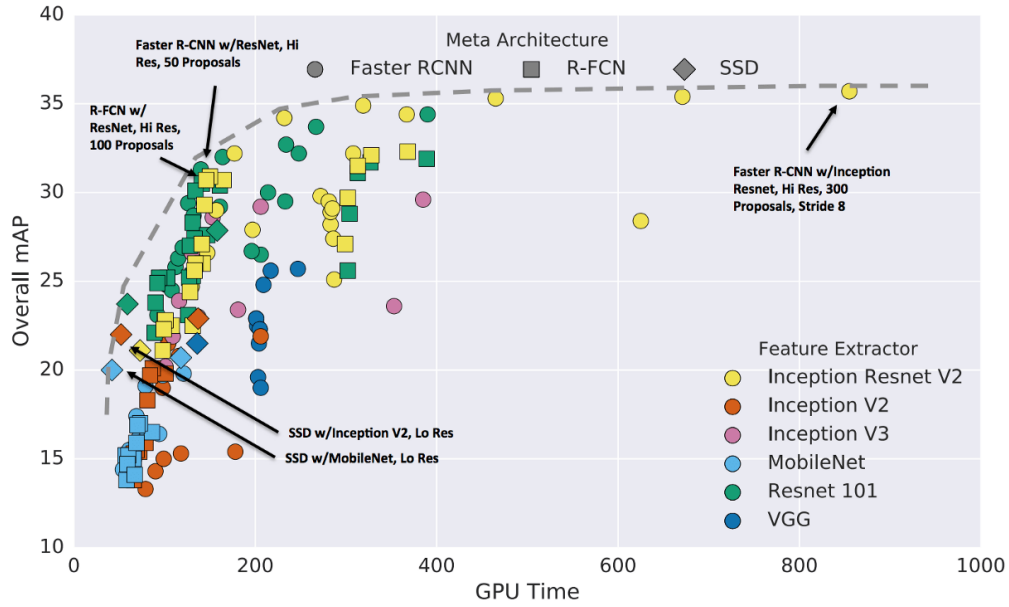
Prvou časťou siete je model, ktorý sa štandartne využíva na klasifikáciu obrázkov. Autori využili sieť VGG-16. Za túto sieť sú pridané konvolučné vrstvy, ktoré obrázok zmenšujú a dovoľujú predikovať detekcie na rôznych veľkostiach obrázku. Na rozdiel od modelu YOLO, ktoré vytvára detekcie len na jednej veľkosti feature mapy, SSD pracuje s viacerými veľkosťami, čo má za následok násobne viac predikcii.



Obr. 3.13: Architektúra siete SSD. [34]

3.5 Porovnanie YOLO, SSD a Faster-RCNN

Faster-RCNN rieši detekciu v dvoch krokoch, pričom najprv využije RPN na predikciu regiónov záujmu a potom tieto navrhnuté regióny pošle do druhej časti na klasifikáciu. Naopak takzvané single-shot detektory, YOLO a SSD, berú detekciu objektov ako regresný problém, pričom ich výstupom sú koordináty bounding boxov a pravdepodobnosť triedy. Vďaka tomu, že sú postavené len na jednej neurónovej sieti, sú oveľa rýchlejšie. V publikácii [34] bolo vykonané meranie rýchlosti na grafickej karte Titan X na datasete Pascal VOC2007.

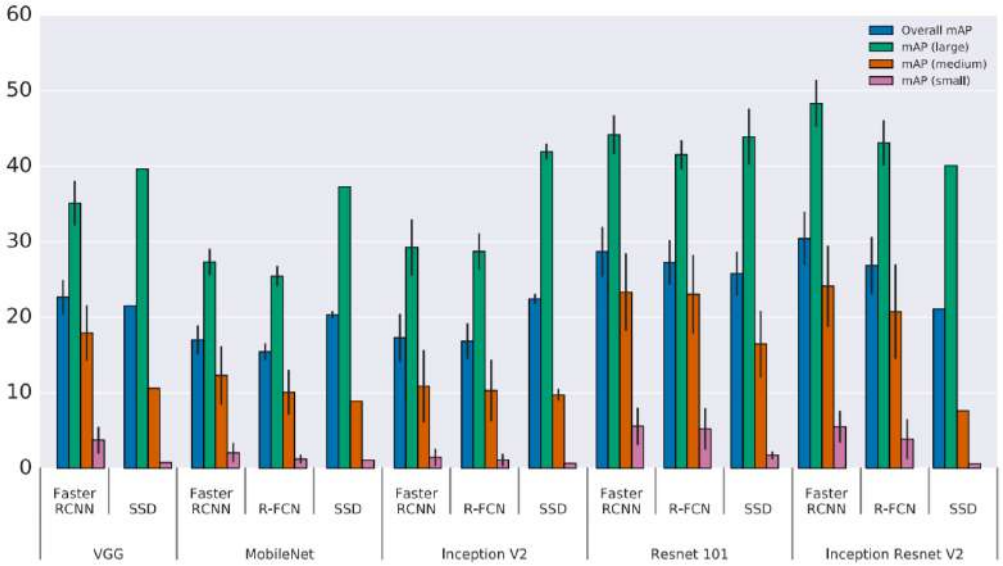


Obr. 3.14: Presnosť vs. čas, pričom tvary značiek označujú meta-architektúru a farby naznačujú extraktor príznakov. Každá (meta-architektúra, extraktor) môže zodpovedať viacerým bodom, kvôli zmene vstupných veľkostí, stridu atď. [25]

Na základe tohto merania dosahovalo Faster-RCNN rýchlosť 7 fps (na obrázkoch veľkosti približne 1000×800px), detektory SSD512 19 fps a SSD300 46 fps. YOLO dosiahlo rýchlosť 21 fps a najrýchlejším bolo Fast YOLO so 155 snímkami za sekundu. Pri takomto vysokom fps, však klesla presnosť modelu.

Na obrázku 3.14 je možné vidieť porovnanie rýchlosti a presnosti modelov SSD a Faster-RCNN. Meranie bolo vykonané na datasete COCO s rôznymi sieťami na extrakciu príznakov. Z tohto grafu je možné vidieť, Faster-RCNN dosahuje lepších výsledkov ako SSD. V publikácii [25] sa taktiež porovnáva presnosť modelov na základe veľkosti detekovaných objektov (obrázok 3.15). Obe modely majú oveľa lepšie výsledky na veľkých objektoch. SSD na malých objektov zlyhávajú, avšak na veľkých sú konkurencie schopné Faster-RCNN, dokonca ho v niektorých prípadoch predbiehajú.

V mojej práci ma zaujíma detekcia malých objektov v prípade detekcie slov a taktiež detekcia veľkých objektov v prípade celých textových regiónov. Taktiež sa zameriam skôr na presnosť ako rýchlosť, preto za najlepšiu alternatívu považujem architektúru Faster-RCNN.



Obr. 3.15: Presnosť podľa veľkosti objektu, meta-architektúry a extraktora príznakov. [25]

3.6 Evaluácia detekčných sietí

Na hodnotenie presnosti detekčných sietí sa najčastejšie využíva metrika mAP (mean Average Precision), ktorá bola s malými odlišnosťami v definícii a implementácii použitá aj detekčných súťažiach ako PASCAL VOC Challenge alebo COCO Detection Challenge [11]. V mojej práci budem využívať implementáciu využitú na vyhodnocovanie pri COCO Detection Challenge. Na vysvetlenie metriky mAP je najprv nutné definovať základné pojmy:

- **Intersection over Union (IoU)** - vyjadruje pomer prieniku a zjednotenia detekovaného a ground truth bounding boxu. Táto hodnota potom na základe zvoleného prahu rozhoduje o tom či je detekcia True Possitive (TP) alebo False Positive (FP). IoU je možné popísať pomocou:

$$IoU = \frac{plocha\ (B_D \cap B_{GT})}{plocha\ (B_D \cup B_{GT})}$$

  kde B_D je detekovaný a B_{GT} je ground truth bounding box.

- **True Positive (TP)** - správna detekcia, detektor vrátil box pre ktorý platí IoU > prah.
- **False Positive (FP)** - detektor vrátil box, pre ktorý však neexistuje odpovedajúci ground truth box a teda IoU < prah.
- **False Negative (FN)** - detektor nevrátil box pre existujúci objekt. K detekcii tohto objektu nedošlo.
- **True Negative (TN)** - detektor nedetekoval objekty, ktoré detekované byť nemali. V detekčných úlohách však v rámci jedného obrázku existuje veľa boxov, ktoré nedetekovali neexistujúci objekt, preto sa TN pri mAP ďalej nevyužíva.
- **Precision** - ide o pomer správne detekovaných objektov ku všetkým detekciám. Môžeme ju vyjadriť rovnicou:

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

- **Recall** - ide o pomer správne detekovaných objektov ku všetkým ground truth objektom. Hovorí o tom koľko skutočných objektov je aj naozaj označených.

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

Average Precision (AP)

Average precision (AP) je možné spočítať ako obsah pod Precision-Recall krivkou. Táto hodnota býva označovaná aj ako AUC (Area Under Curve). Výsledok modelu je tým lepší čím vyššie je AUC, teda ak sa AUC blíži k 1, model dosahuje vynikajúcich výsledkov. V praxi je AP spočítaná ako spriemerovaná presnosť zo všetkých recall hodnôt medzi 0 a 1. [14]

COCO Detection Challenge počíta AP zo 101 Recall bodov z intervalu [0, 1], ktorý sa prechádza s krokom 0.01 na Precision-Recall krivke [11]. Ku týmto bodom je priradená maximálna hodnota Precision napravo od aktuálneho bodu (takúto interpoláciu hodnôt Precision je možné vidieť na obr. 3.16). AP je v tomto prípade definované ako:

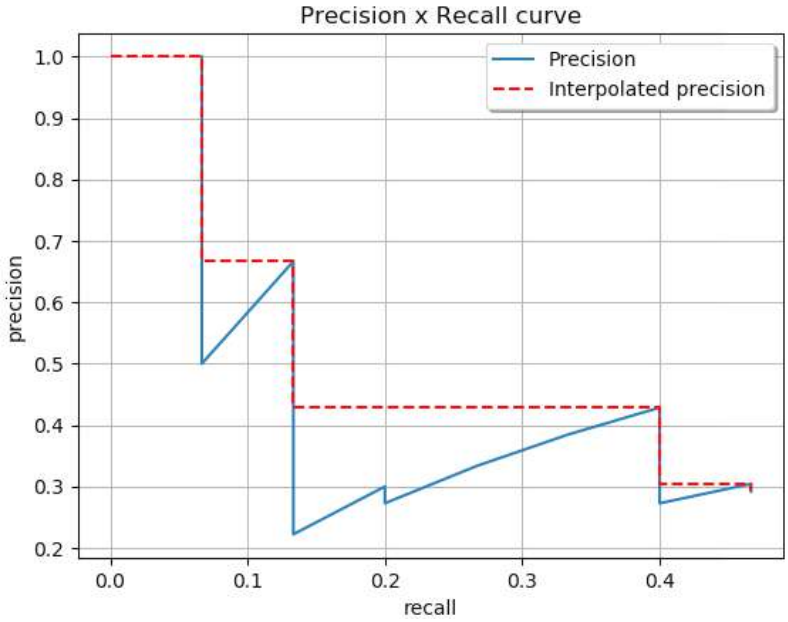
$$AP = \frac{1}{101} \sum_{r=\{0,0.01,\ldots,1\}} p_{interp}(r)$$

kde r sú hodnoty Recall a $p_{interp}(r)$ sú interpolované hodnoty Precision pre dané r.

Mean Average Precision (mAP)

Výpočet AP zahŕňa len jednu triedu objektov. Pri detekcii však zvyčajne detekujeme naraz niekoľko tried. Priemer cez všetkých K tried AP je definovaný ako mAP.

$$mAP = \frac{1}{K} \sum_{i=1}^{K} AP_i$$



Obr. 3.16: Precision-Recall krivka. [14]

COCO Detection Challenge

Hlavnou metrikou pri COCO Detection Challenge je však mAP [13], ktoré bolo vypočítané spriemerovaním AP cez všetky triedy a zároveň cez 10 IoU prahov. Tieto prahy sú z intervalu [0.5,0.95] s krokom 0.05. Najpv sú teda spočítané AP pre jednotlivé triedy s prahmi 0.5, 0.55, 0.6 ... Potom sú spočítané hodnoty mAP pre dané prahy. Výsledná hodnota je nakoniec spočítaná ako priemer cez všetky mAP.

$$mAP^{0.50:0.95} = \frac{1}{10} \sum_{i \in \{0.5,0.55,..,0.95\}} mAP^{IoU=i}$$

Popri metrike $mAP^{0.50:0.95}$ sa často uvádzajú aj hodnoty $mAP^{0.50}$ a $mAP^{0.75}$.

3.7 Evaluačná metrika analýzy layoutu

K hodnoteniu výkonu algoritmov na evaluáciu layoutu v súčastnosti neexistuje jednotný prístup. Existuje niekoľko evaluačných metrík, ktoré sa používajú naprieč rôznymi štúdiami.

V niekoľkých štúdiách, na ktoré som narazila bolo vyhodnocovanie robené na úrovni pixelov. Každému pixelu sa priradila hodnota true positive, true negative, false positive alebo false negative. Z toho sa potom vypočítalo recall, precision a accuracy. Táto metrika sa používala hlavne pri segmentačných algoritmoch, ktoré obrázok delili nielen na jednotlivé regióny, ale taktiež boli schopné označiť obrázok alebo rôzne ornamenty, ktoré sa v dokumente nachádzali. Táto metóda vyhodnocovania sa mi nezdala vhodná, pretože v mojom prípade potrebujem vyhodnotiť či skupina pixelov patrí do určitého regiónu, nie či patria do triedy región všeobecne. Mohlo by sa tak napríklad stať že dva regióny, ktoré majú byť vertikálne oddelené, budú detekované ako jeden región a metrika nám vráti vysoké skóre pretože pixely sú naozaj dobre zaradené do triedy región. My sme však nesprávne namiesto dvoch regiónov detekovali jeden.

Evaluáciu ktorú som sa rozhodla implementovať (ďalej ako F-score) bola použitá v challengi IDCAR2009 [17] a s miernymi úpravami použitá v niekoľkých ďalších štúdiách. V posledných IDCAR challengoch bola použitá novšia evaluácia, ktorá je robustnejšia a vyhodnocuje niekoľko rôznych metrík. Tieto metriky sú implementované v programe [47], ktorý je možné si bezplatne stiahnuť a používať. Z technických dôvodov však tento program nevyužila (nekompatibilita XML a nemožnosť spustiť program na celom testovacom datasete naraz). Výhodou vlastnej evaluačnej metriky bola taktiež možnosť jej prispôsobenia.

Metriku F-score som implementovala v jazyku Python. Základom tejto metriky je počítanie zhôd medzi detekovanými a ground truth regiónmi. Najprv sa vytvorí tabuľka MatchScore, ktorá tieto zhody reprezentuje. Nech G_j je množina pixelov ground truth regiónu j, D_i je množina pixelov detekovaného regiónu i a $T(s)$ je funkcia, ktorá spočíta prvky v množine s. Tabuľka $MatchScore(i, j)$ je potom:

$$MatchScore(i, j) = \frac{T(G_j \cap D_i)}{T(G_j \cup D_i)} \tag{3.8}$$

Slovne popísané ide o Intersection over Union (IoU), čo je prienik počtu pixelov oboch bounding boxov vydelený ich zjednotením. Číslo blížiace sa k 1 je maximálna zhoda, naopak číslo blížiace sa k 0 je minimálna zhoda. IoU sa vypočíta pre všetky možné dvojice regiónov a zapíše do MatchScore tabuľky.

	G_1	G_2	G_3	G_4	G_5	G_6
D_1		0.85				
D_2	0.05		1.0			
D_3	0.3			0.6		0.09
D_4		0.08		0.5	0.8	
D_5						0.9

Tabuľka 3.1: Príklad tabuľky MatchScore.

Následne sa z tabuľky odstránia hodnoty pod určitý prah. Ide o drobné prekryvy bounding boxov, ktoré môžu nastať ale vo výsledku na nich až tak nezáleží. Tento prah som nastavila na 0.1.

Ďalším krokom je v tabuľke nájsť one-to-one zhody. Ide o bunku tabuľky, ktorá v danom riadku a stĺpci nemá ďalšiu hodnotu a zároveň hodnota tejto bunky je vyššia ako akceptačný prah. V štúdiách bol väčšinou tento prah nastavený na 0.9, avšak pri jeho použití som si všimla že často krát dobre označené regióny neboli označené za one-to-one zhody. Išlo hlavne o menšie regióny ako nadpisy, keď jeden z bounding boxov bol o niečo vyšší alebo širší a pokrýval tak aj nejakú časť pozadia. Rozhodla som sa preto inšpirovať vyhodnocovaním ako pri COCO evaluácii, popísanej v kapitole 3.4 a zvolila som tri prahy: 0.5, 0.75 a 0.9. F-score je potom počítané pre každý prah osobitne. Tabuľku s vyznačenými one-to-one zhodami pre prah 0.5 zobrazuje tabuľka 3.2.

	G_1	G_2	G_3	G_4	G_5	G_6
D_1		0.85				
D_2			1.0			
D_3	0.3			0.6		
D_4				0.5	0.8	
D_5						0.9

Tabuľka 3.2: Príklad tabuľky MatchScore s vyznačenými one-to-one zhodami pre prah 0.5.

Ďalej majme N ako počet ground truth regiónov a M ako počet detekovaných regiónov. Detection rate (DR) a recognition accuracy (RA) potom vypočítame následovne:

$$DR = \frac{one - to - one}{N}, \quad RA = \frac{one - to - one}{M} \tag{3.9}$$

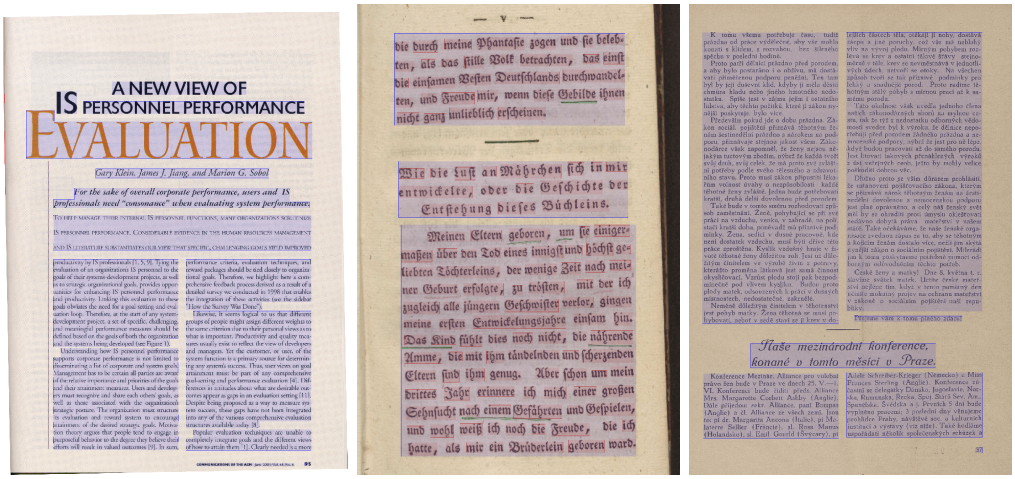
Detection rate nám vyjadruje koľko z detekovaných regiónov bolo detekovaných správne a recognition accuracy nám hovorí koľko z ground truth regiónov bolo detekovaných. Výsledná metrika F-score je potom definovaná:

$$F - score = \frac{2 * DR * RA}{DR + RA} \tag{3.10}$$

Kapitola 4

Dátové sady

Na túto tému som našla niekoľko dostupných datasetov, zameriavala som sa na datasety, ktoré obsahujú ground truth textových regiónov alebo slov.



(a) (b) (c)

Obr. 4.1: Ukážky obrázkov s ground truth z databáz: (a) PRImA Layout Analysis Dataset, (b) Handwritten Annotation Detection Dataset, (c) Noviny Dataset.

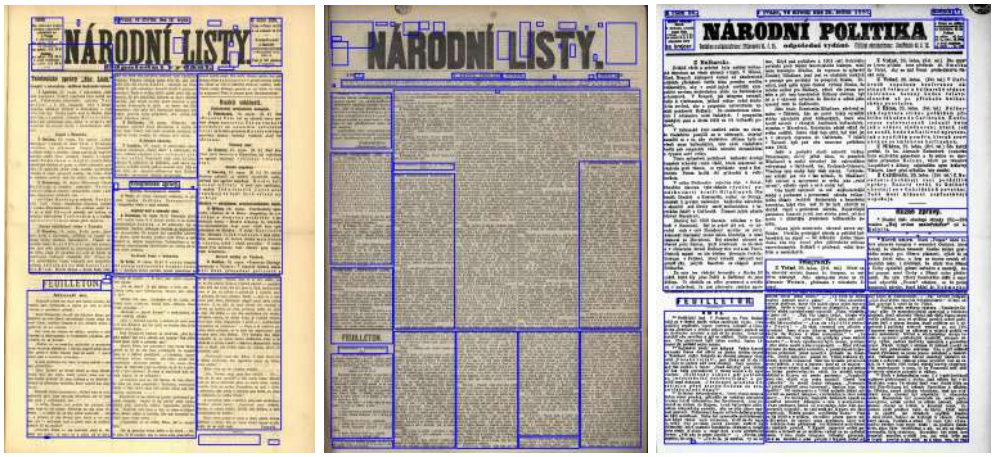
1. **PRImA Layout Analysis Dataset** [4] bol vytvorený na evaluáciu metód analýzy layoutu. Obsahuje rôzne druhy rozložení layoutov, jednoduchšie ale aj komplexnejšie rozloženia. Dataset sa zameriava na časopisy a vedecké publikácie, ktoré boli naskenované na rozlíšení 300 dpi. V datasete sa nachádza 478 obrázkov vo formáte TIF, pričom každý má svoje ground truth vo formáte XML. Ground truth poskytuje koordináty textových regiónov.

2. **DDi-100** [54] je syntetický dataset, ktorý je založený na 7000 naskenovaných stránkach kníh. Z každej stránky bolo augumentáciou (geometrické transformácie, pridanie pozadia, pečiakty..) vytvorených 15 obrázkov. Celý dataset tak obsahuje okolo 100 000 obrázkov. Obrázky sú vo formáte PNG a každý má priradený ground truth vo formáte PICKLE, ktorý obsahuje bounding boxy pre jednotlivé slová aj písmená.

3. **IEHHR** [15] obsahuje 125 obrázkov ručne písaných historických textov zo 17. storočia. Obrázky sú vo formáte JPG a ground truth obsahuje bounding boxy pre slová.

4. **Handwritten Annotation Detection Dataset** [33] poskytuje 50 obrázkov historických textov. Dokumenty pochádzajú z rôznych zdrojov a boli digitalizované rôznymi spôsobmi, čo robí tento tento dataset ťažším na spracovanie. Ground truth je vo formáte XML a obsahuje polohu slov, riadkov aj textových regiónov. Ukážku je možné vidieť na obrázku 4.1b.

5. Ďalší dataset mi bol poskytnutý vedúcim mojej práce (ďalej ako **Noviny Dataset**). Tento dataset obsahuje 6134 obrázkov dobových novín s prevažne komplexnými, ale aj jednoduchšími layoutami. Obrázky sú vo formáte jpg a ground truth obsahuje pozície textových regiónov, riadkov a slov. Extrakcia slov však bola robená automaticky pomocou CTC OCR siete, čo spôsobilo že bounding boxy slov neboli úplne presné. Posledné písmenko slova nebolo často zahrnuté v bounding boxe celé a nadpisy mali taktiež veľmi nepresné ohraničenie. Tento dataset bol pre mňa východiskový, preto som ako prvý krok vytvorila program, ktorý dokáže tieto bounding boxy upraviť. Popísaný je v nasledujúcej kapitole. Ďalším problémom tohto datasetu boli boxy regiónov, ktoré taktiež neboli značené ručne. V komplexnejších layoutoch sa preto skoro na každom obrázku nachádzali chybne označené regióny, čo komplikovalo ďalšiu prácu s týmto datasetom.



Obr. 4.2: Ukážky obrázkov s chybne označenými regiónmi z datasetu Noviny.

Kapitola 5

Návrh riešenia

Ako prvé je potrebné jasne si definovať, čo bude považované za textový región a kde sú jeho hranice. V rôznych datasetoch boli regióny stanovené rôzne. Už na obrázku 4.1 je možné vidieť že v datasete PRImA je za samostatný textový región považovaný každý odsek, zatiaľ čo v datasete Noviny tvoria textový región aj niekoľko odsekov. Keďže som primárne vychádzala z datasetu Noviny, na ktorom som aj trénovala neurónové siete, textové regióny boli definované podľa neho.

Ako textový región bude ďalej označovaný súvislý blok textu, pre ktorý navyše platia nasledujúce pravidlá:

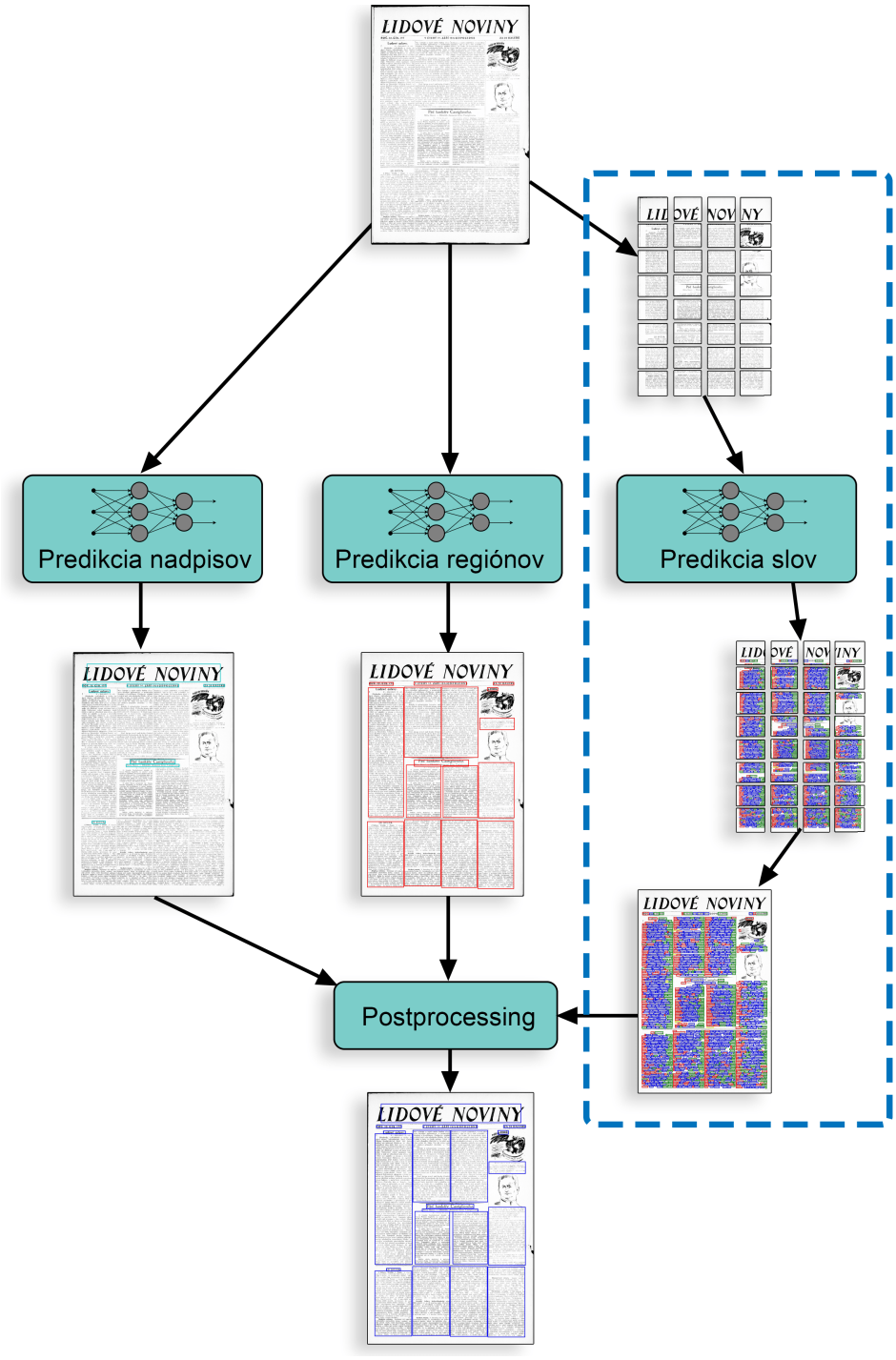
1. V jednom textovom regióne sa nemôžu nachádzať 2 a viac stĺpce textu, takéto stĺpce sú stále vertikálne oddelené.
2. Nadpis je považovaný za samostatný textový región. Nadpisy sa vyznačujú väčšou veľkosťou, zmenou fontu alebo medzerami v okolí nadpisu. Viac o nadpisoch je možné nájsť v podkapitole 6.5.
3. Odsadenie, ktoré značí nový odsek nerozdeľuje textové regióny.
4. Malé textové bloky ako napríklad číslo strany alebo dátum vydania časopisu, okolo ktorých je často veľká medzera alebo sú podčiarknuté sú osobitné textové regióny.
5. Stĺpec textu je rozdelený na viac textových regiónov v týchto prípadoch:
   (a) nachádza sa v ňom rozdelovač ako napríklad vodorovná čiara alebo bodkovaná čiara,
   (b) nachádza sa v ňom nadpis.

Finálny návrh riešenia sa postupne formoval počas riešenia práce a odvíjal sa od výsledkov rôznych experimentov. Tieto experimenty a ich výsledky sú detailnejšie popísané v ďalšej kapitole.

Na obrázku 5.1 je možné vidieť celkový výsledný návrh riešenia. Ten zahŕňa 3 natrénované neurónové siete: sieť na detekciu nadpisov, regiónov a slov, ktoré sú taktiež klasifikované podľa pozície v riadku. Slovo môže mať v riadku 4 pozície: je prvé slovo v riadku, je posledné slovo v riadku, je medzi prvým a posledným slovom alebo je v riadku samotné a teda je prvé aj posledné.

Na obrázku je taktiež vidieť, že slová sú ako jediné detekované na výrezoch vstupného obrázku. Je to kvôli nedostatočnej pamäti GPU. Už trénovanie takejto siete nebolo na celých

obrázkoch možné, pretože niektoré dokumenty obsahovali aj tisíce slov. Tieto výrezy je po predikcii potrebné znovu spojiť do jedného obrázku (viac o riešení tohto problému viz 6.2).



Obr. 5.1: Návrh riešenia.

V postprocessingu sú informácie z týchto sietí vhodne spracované. Ide napríklad o odstránenie prekrývajúcich sa textových regiónov, ich delenie, expanzia alebo zmenšie s cieľom dostať čo najlepšie výsledky.

Celkový návrh rišenia je možné rozdeliť na dva prípady. Prvým, je prípad keď sa využívajú informácie zo všetkých troch sietí. Na vstupnom obrázku sa detekujú nadpisy, regióny a slová. Tento prístup je výrazne pomalší a dosahuje len o málo lepšiu presnosť ako riešenie, keď sa využívajú len informácie zo siete na detekciu regiónov a nadpisov (schéma bez modro vyznačenej oblasti). Viac o rozdieloch týchto dvoch riešení je popísané v podkapitole 7.1.

Kapitola 6

Experimenty

V nasledujúcej kapitole bude popísaná praktická časť práce. Ide o experimenty s dátovou sadou, ktoré som robila s cieľom detekovať textové regióny.

Prvotnou myšlienkou bolo pristúpiť k analýze layoutu prístupom zdola nahor, teda najprv detekovať slová a následne ich zhlukovať do väčších celkov. Ako už bolo spomenuté, v datasete Noviny boli bounding boxy slov nepresné, preto som ako prvý krok vytvorila krátky program na ich úpravu, ktorý je popísaný v podkapitole 6.1.

Následne som pristúpila k trénovaniu neurónovej siete. Na základe zistených informácii z časti 3.5 som sa rozhodla využiť sieť Faster-RCNN. Nekladiem totiž dôraz na rýchlosť, preto som architektúru YOLO vylúčila ako prvú. Keďže predpokladám, že slová v dokumentoch budú malých rozmerov, rozhodla som sa využiť model Faster-RCNN, ktorý vykazuje o niečo lepšie výsledky na malých objektoch. Vo všetkých experimentoch som využila implementáciu Faster-RCNN Resnet-50 FPN z balíčku torchvision[41]. Torchvision je balíček, ktorý obsahuje populárne datasety, architektúry modelov a základné transformácie pre prácu s obrázkami. Obsahuje taktiež predtrénované modely na datasete COCO. Keďže sa mi dataset Noviny nezdal rozsiahly, rozhodla som sa vyskúšať trénovať detekciu slov na predučenej sieti. Vyskúšala som taktiež nepredučenú sieť a zistila som, že obe sa po určitom čase dostanú ku podobným presnostiam, avšak u nepredtrénovanej siete to trvá tri krát pomalšie. U všetkých ďalších experimentoch, ktoré budú spomuneté bola preto využitá predtrénovaná sieť Faster-RCNN.

Samotné trénovanie siete nebolo na mojom domácom počítači možné, kvôli slabej grafickej karte. Hľadala som preto alternatívy. Ako prvé som využívala Google Colab, avšak ten pri dlhých a náročných výpočtoch začne užívateľa limitovať a udeľovať aj niekoľkodňové cooldowny. Následne som preto prešla na MetaCentrum, ktoré študentom a vedeckým pracovníkom ponúka bezplatné výpočetné a úložné kapacity. Prideľovaných mi bolo 16GB pamäte GPU po dobu 10 hodín.

Sieť na detekciu slov sa mi podarilo úspešne natrénovať a aj keď by slová bolo možné podľa rôznych algoritmov zhlukovať do riadkov a následne regiónov, dalo sa však očakávať, že pri takýchto zložitých layoutoch, by výsledky neboli dobré a algoritmus by potreboval zo vstupného obrázku ďalšie informácie.

Ku poznatkom o pozícii slov som sa teda snažila získať ďalšie. Keďže dataset Noviny obsahoval aj informácie o riadkoch, ďalším experimentom bolo natrénovanie siete, ktorá by nielen detekovala slová, ale taktiež ich dokázala zaradiť do tried na základe ich polohy v riadku.

Následne som vyskúšala natrénovať sieť, ktorá by namiesto toho či je slovo prvé alebo posledné v riadku dokázala detekovať prvé a posledné slová v regióne. Tento experiment

však nedosahoval dobrých výsledkov a preto pri výslednej analýze layoutu nebude použitý. Tieto experimenty sú popísané v podkapitole 6.3.

Experiment so sieťou, ktorá dokáže detekovať počiatočné a konečné slová v riadkoch prinášal do analýzy layoutu nové informácie, avšak vykazoval chybovosť a pri zhlukovaní slov do regiónov sa na tieto výsledky nedalo spoliehať. Taktiež problémom pri detekcii slov boli príliš veľké slová ako nadpisy a názvy periodík. Tieto nadpisy nemali spoľahlivú anotáciu už ani v trénovacom datasete a chyba sa preniesla aj do výsledkov. Preto mojimi ďalšími experimentami boli trénovanie siete na celých textových regiónoch a nadpisoch. Priebeh a výsledky je možné nájsť v podkapitolách 6.4 a 6.5.

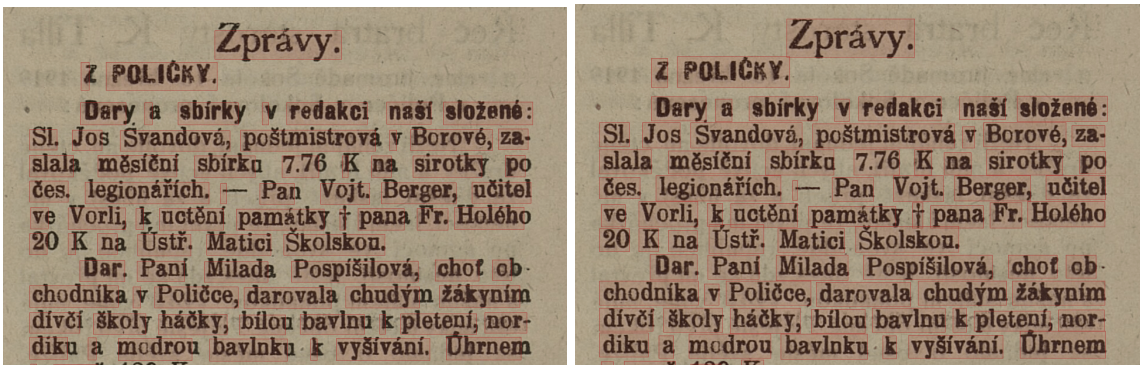
Týmto som zo vstupného obrázku dostala informácie o slovách, nadpisoch a regiónoch. Všetky však mali určitú chybovosť. Preto sa informácie z týchto troch sietí ešte spracujú v postprocessingu, ktorého popis je možné nájsť v časti 6.6.

Sieť na detekciu slov bola trénovaná na datasete Noviny, ktorý obsahuje 6134 obrázkov. Na tomto datasete boli prvotne trénované aj siete na detekciu regiónov a slov na základe polohy v riadku, avšak tento dataset obsahoval viacero nepresností a šum, ktorý sa prenášal do natrénovaných sietí. Preto som tieto siete skúsila vylepšiť pretriedením datasetu. Odstránila som obrázky, ktorých anotácie boli príliš nepresné. Zo 6134 obrázkov som tak dostala 4248, pričom prvých 300 z nich som ešte označila ručne tak, aby boli regióny čo najpresnejšie. Po pretrénovaní siete na detekciu prvých a posledných slov v riadku a siete na detekciu regiónov, sa výsledky výrazne zlepšili. Preto budú v ďalšej kapitole popisované trénovania týchto dvoch sietí práve na pretriedenom datasete Noviny.

Dataset na trénovanie nadpisov bude viac popísaný v časti 6.5.

6.1 Algoritmus na úpravu bounding boxov

Algoritmus na úpravu bounding boxov bol napísaný v jazyku Python. Na vstupe má obrázok s textom a k nemu prislúchajúci XML dokument, ktorý obsahuje pozície bounding boxov jednotlivých slov. Obrázok je najprv kvôli vyššej rýchlosti algoritmu zmenšený a je na neho aplikované Gaussovo rozmazanie s malým jadrom, kvôli odstráneniu šumu. Na obrázok je ďalej aplikované Otsu prahovanie. Jednotlivé bounding boxy slov sú následne upravované na takto predspracovanom obrázku.



(a) (b)

Obr. 6.1: Úprava bounding boxov slov: (a) pôvodný obrázok, kde bounding boxy majú správnu výšku ale často im chýba časť alebo takmer celé posledné písmeno, (b) obrázok po úprave mojim algoritmom.

Použitý bol algoritmus region growing, ktorý na vstupe potrebuje semienka. z ktorých sa bude postupne šíriť do okolitých pixelov, kým rozdiel v hodnotách susedných pixelov neprekročí daný prah. Keďže môj obrázok je po Otsu prahovaní binárny, s hodnotou prahu nebolo potrebné nijak experimentovať a stačilo ho nastaviť na hodnotu 1. Semienka do tohto algoritmu sú zvolené ako pixely patriace slovu, ktorých sa bounding box dotýka.

Pri týchto častiach slova sa predpokladá že pokračujú za hranicu bounding boxu (obrázok 6.2b). Algoritmus region growing je kvôli úspore času spustený na výreze okolo slova ako je možné vidieť na obrázku. Následne už len stačí rozšíriť pôvodný bounding box o pixely, ktoré do nášho výberu pridal region growing (obrázok 6.2c).

(a) (b)

(c) (d)

Obr. 6.2: Algoritmus na úpravu bounding boxov: (a) pôvodný bounding box, (b) semienka pre region growing (vyznačené červenou farbou na pravom okraji obrázku), (c) výstup z algoritmu region growing, (d) upravený bounding box.

Výsledok je krátky program, ktorý som spustila nad celým datasetom Noviny a získala som tak presnejšie XML anotácie. Tento program sa priamo nespúšťa pri analýze layoutu, ide len o predspracovanie dát, preto ani dlhší chod tohto algoritmu nemusel byť problém. Priemerné spracovanie jedného obrázku s procesorom Intel Core i7-9750H bolo okolo 1.5s.

6.2 Sieť na detekciu slov

Ako bolo spomenuté vyššie, na trénovanie bola použitá architektúra Faster-RCNN z balíku torchvision. Trénovanie prebiehalo na datasete Noviny, ktorý som rozdelila v pomere 80:20 na trénovaciu a validačnú sadu. Trénovanie na celých stránkach dokumentov však nebolo

možné kvôli nedostatočnej pamäti GPU. Jeden obrázok totiž mohol obsahovať aj tisíce slov. Preto sú vstupom do siete len výrezy obrázkov dokumentov, ako je možné vidieť na obrázku 6.3. Tieto výrezy sú z pôvodného obrázku robené na náhodnej pozícii.

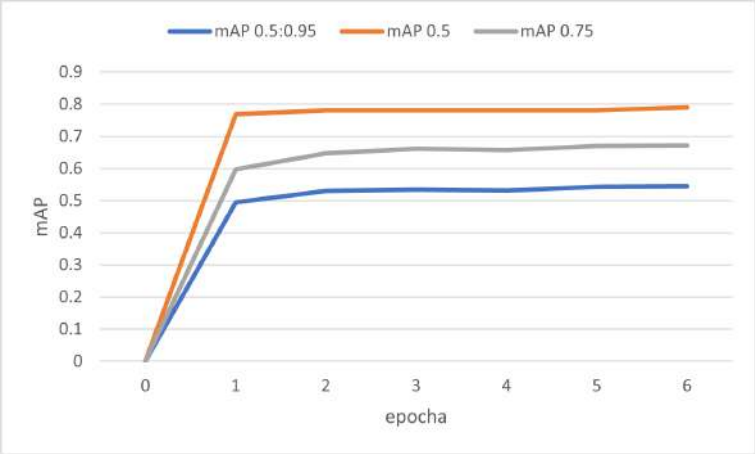
Ako optimizer bol použitý algoritmus Adam [28], ktorý je rozšírením pre stochastický gradientný zostup. Adam nemá fixný krok učenia, ale jeho veľkosť sa aktualizuje. Je to populárny algoritmus, ktorý poskytuje uspokojivé a rýchle výsledky. Learning rate bol na začiatku nastavený na na 0.001.

Batch size, alebo počet snímkov, ktoré sieťou prejdú pred úpravou parametrov bol stanovený na 8.

Presnosť mAP na validačnej sade počas trénovania je možné vidieť na grafe na obrázku 6.4. Výsledné hodnoty mAP boli: $mAP^{0.50} = 0.79$, $mAP^{0.75} = 0.671$ a $mAP^{0.50:0.95} = 0.545$.

Obr. 6.3: Príklady výrezov obrázkov dokumentov, ktoré boli vstupom pre neurónovú sieť.

Výrezy, na ktorých naučená sieť detekovala slová, bolo potrebné nakoniec znovu pospájať do jedného obrázku. Vytvorila som preto krátky program, ktorý postupne prechádza obrázok dokumentu s posuvným oknom. Bolo potrebné aby sa vedľajšie okná prelínali, aby malo aj slovo ktoré sa nachádza na okraji posuvného okna a nie je v okne obsiahnuté celé bounding box, ktorý ho celé obsiahne.



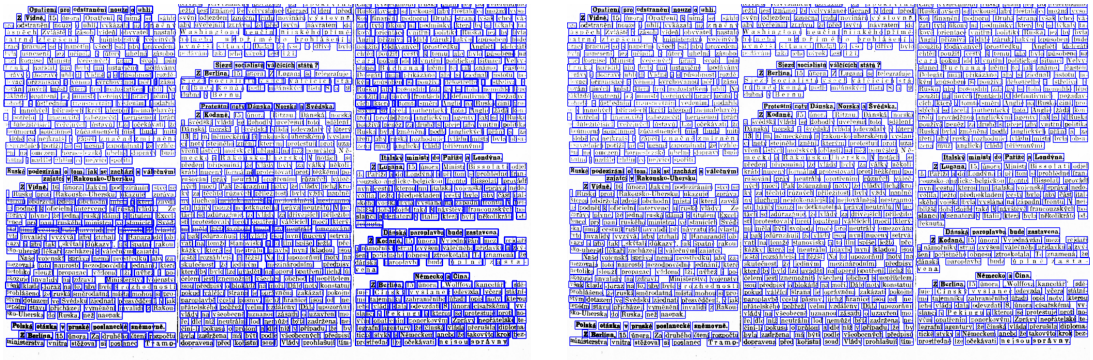
(a)

Obr. 6.4

Detekované bounding boxy zo všetkých posuvných okien boli ďalej spracované tak, aby sa odstránili prekrývajúce sa, alebo malé bounding boxy z okrajov posuvných okien. Všetky

bounding boxy boli zoradené podľa veľkosti od najmenšieho po najväčšie a pre každý bounding box sa overilo, či jeho plocha nie je prekrytá aspoň na 30% iným bounding boxom. Ak áno, menší bounding box bol odstránený. Postupne sa prešli všetky detekované bounding boxy. Výsledkom boli odstránené malé prekrývajúce sa detekcie. Výsledok predikcie a pretriedenia bounding boxov je možné vidieť na obrázku 6.5. Na ľavej časti obrázku, pred pretriedením, je možné pozorovať aj vertikálné pásy, ktoré sú spôsobené prekrývaním posuvného okna pri predikcii.

So sieťou som skúšala ďalej experimentovať a vylepšovať jej výsledky. Vyskúšala som augumentáciu obrázkov v podobe zmeny veľkosti. Veľkosť som menila náhodne v percentuálnom rozmedzí 20% a 40% s rovnomerným rozložením od pôvodného obrázku. Keď som však porovnala výsledky bez augumentácie a s augumentáciou veľkosti, všetky hodnoty boli viac menej rovnaké a $mAP^{0.50}$ sa pohybovalo okolo 0.79.



(a) (b)

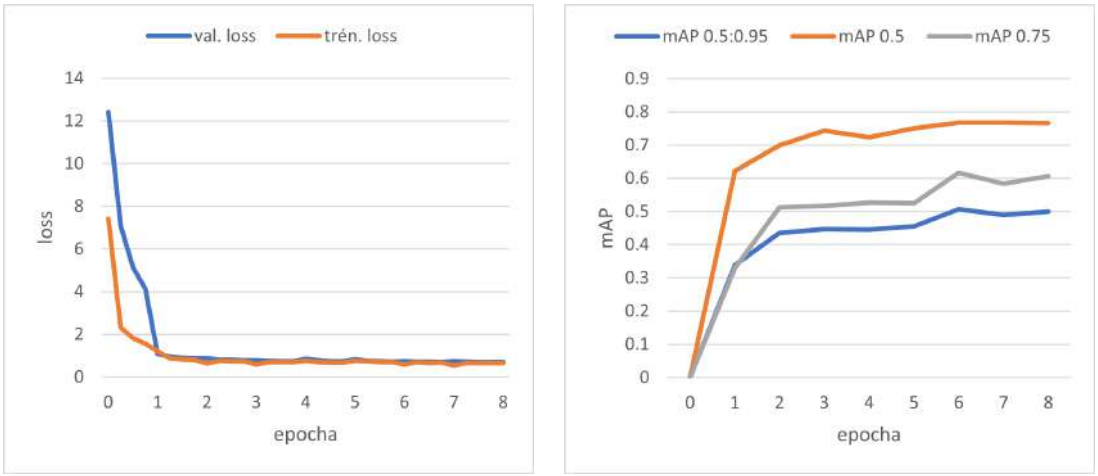
Obr. 6.5: Spojenie výrezov obrázku po detekcii slov: (a) detekované slová pred triediacim algoritmom, (b) detekované slová po vyradení prekrývajúcich sa detekcii.

Ďalej som skúšala meniť anchor boxy RPN. Najprv som si vytvorila krátky program, ktorý mi vypočítal histogram veľkostí a pomerov strán slov v datasete. Ako sa dalo u slov očakávať, väčšina bola širšia ako vyššia (bounding box, ktorý je vyšší ako širší je možné čakať len u jednopísmenných slov ako sú spojky a predložky) a ich veľkosť bola menšia ako prednastavené hodnoty v sieti. Tieto paramentre som na základe zistených informácii zmenila, avšak presnosť siete sa znovu nezlepšila a pohybovala sa okolo $mAP^{0.50} = 0.79$

6.3 Sieť na detekciu prvých a posledných slov riadku

Trénovanie prebiehalo znovu na predtrénovanom Faster-RCNN modely, kde vstupy boli menšie výrezy obrázku. V tomto prípade, však boli použité 4 triedy na klasifikáciu slov, kde triedy boli: prvé slovo v riadku, posledné slovo v riadku, slovo v strede riadku a slovo ktoré je zároveň prvé aj posledné, teda je v riadku len jedno.

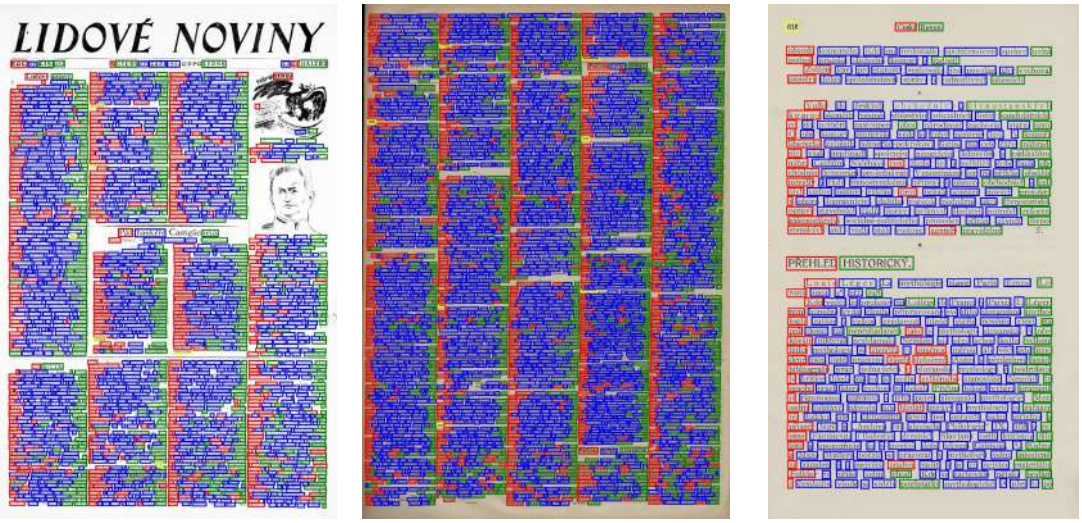
Na trénovanie bolo použitých spomínaných 4248 pretriedených fotiek, ktoré boli znovu rozdelené v pomere 80:20 na trénovaciu a validačnú množinu. Hodnoty batch size a Adam boli použité rovnaké ako v prípade trénovania slov, teda 8 a 0.001.



(a) (b)

Obr. 6.6: Grafy reprezentujúce priebeh trénovania neurónovej siete: (a) validačná a trénovacia chyba, (b) mAP.

Priebeh trénovania je možné vidieť na grafoch na obrázku 6.6. Výrezy s predikovanými slovami bolo potrebné znovu pospájať do celého obrázku dokumentu. Algoritmus na ich spojenie je veľmi podobný tomu pri detekcii slov v predchádzajúcej podkapitole, avšak v tomto prípade sa v úvahu bralo aj percento pravdepodobnosti že slovo patrí do danej triedy. Ak mali dve slová IoU vyššie ako 0.8 potom sa odstránilo to, ktorého trieda mala nižšiu pravdepodobnosť. Výsledok po pospájaní výrezov je možné vidieť na obrázku 6.7.



Obr. 6.7: Obrázky dokumentov s predikovanými slovami podľa pozície v riadku. Červenou sú zaznačené prvé, zelenou posledné slová riadku, modrou slová v strede riadku a žltou posledné a zároveň prvé slovo v riadku.

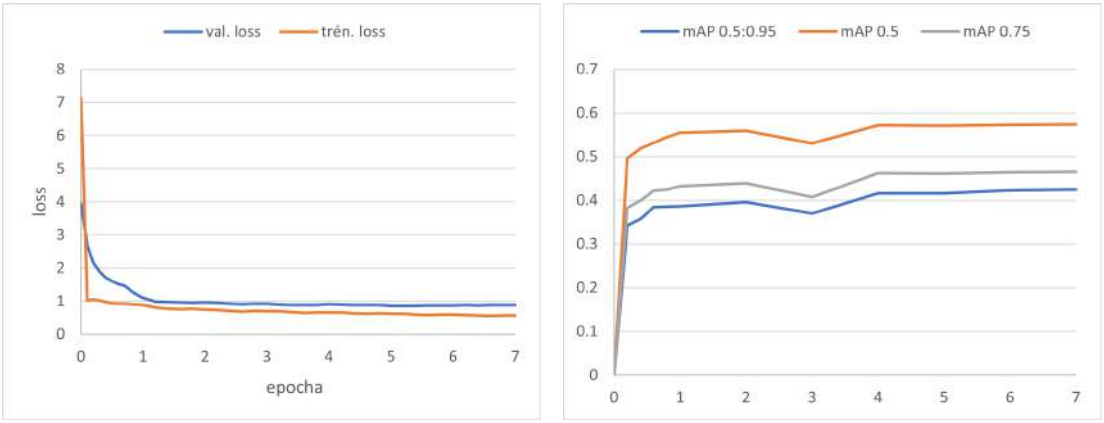
Ďalej som skúšala podobným spôsobom natrénovať prvé a posledné slová v regiónoch. Znovu som teda mala 4 triedy s prvým, posledným, stredovým slovom v regióne a so slovom ktoré je v regióne jediné a teda je prvé aj posledné. Dosiahnuté výsledky však neboli vôbec použiteľné. Sieť dosiahla presnosti $mAP^{0.50} = 0.414$, $mAP^{0.75} = 0.323$ a

$mAP^{0.50:0.95} = 0.272$, aj to len vďaka dobre detekovaným slovám v strede regiónov. Začiatočné a konečné slová neboli detekované dobre takmer vôbec. Výsledky z tejto siete je možné vidieť na obrázku 6.8. Možnou príčinou nefunkčnosti tohto experimentu môže byť málo dát alebo nevyvážené dáta pri trénovaní. Problémom môžu taktiež byť nesprávne anotácie regiónov v trénovacom datasete.

Obr. 6.8: Obrázky dokumentov s predikovanými slovami podľa pozície v regióne. Červenou sú zaznačené prvé, zelenou posledné slová regiónu, modrou slová v strede regiónu.

6.4 Sieť na detekciu regiónov

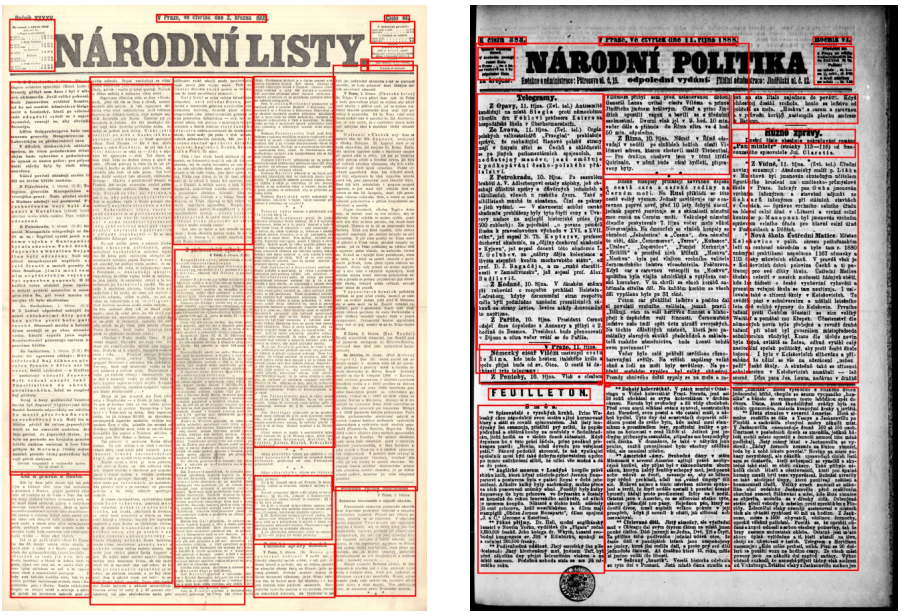
Pri trénovaní celých textových regiónov bola znovu použitá Faster-RCNN predtrénovaná na COCO datasete ako predošlých prípadoch. Dataset som znovu rozdelila v pomere 80:20 na trénovaciu a validačnú sadu.



(a) (b)

Obr. 6.9: Grafy reprezentujúce priebeh trénovania neurónovej siete na detekciu textových regiónov: (a) validačná a trénovacia chyba, (b) mAP.

Sieť však pri prvých pokusoch nemala dobré výsledky, preto som skúšala znižovať learning rate a pri hodnote 0.0001 sa sieť začala učiť a dosiahla $mAP^{0.50}$ okolo 0.5. Priebeh trénovania je možné vidieť na grafoch na obrázku 6.9. Výsledné presnosti mAP po siedmych epochách boli $mAP^{0.50} = 0.574$, $mAP^{0.75} = 0.465$ a $mAP^{0.50:0.95} = 0.425$.



Obr. 6.10: Ukážky detekcie regiónov.

6.5 Sieť na detekciu nadpisov

Jedným z nedostatkov, ktoré mal dataset Noviny, ktorý bol pre mňa východiskový bola anotácia nadpisov. Veľké nadpisy ako sú napríklad názvy novín neboli označené vôbec, poprípade boli zašumené malými bounding boxami v okolí nadpisu. Stredne veľké nadpisy zväčša boli označené správne a menšie nadpisy, občas zaznačené boli a inokedy zasa nie. Zo siete na detekciu textových regiónov, ktorá bola popísaná v predošlej podkapitole preto nebolo možné očakávať dobré výsledky v oblasti nadpisov. Vyskúšala som preto osobitne natrénovať sieť len na obrázkoch s vyznačenými nadpismi.

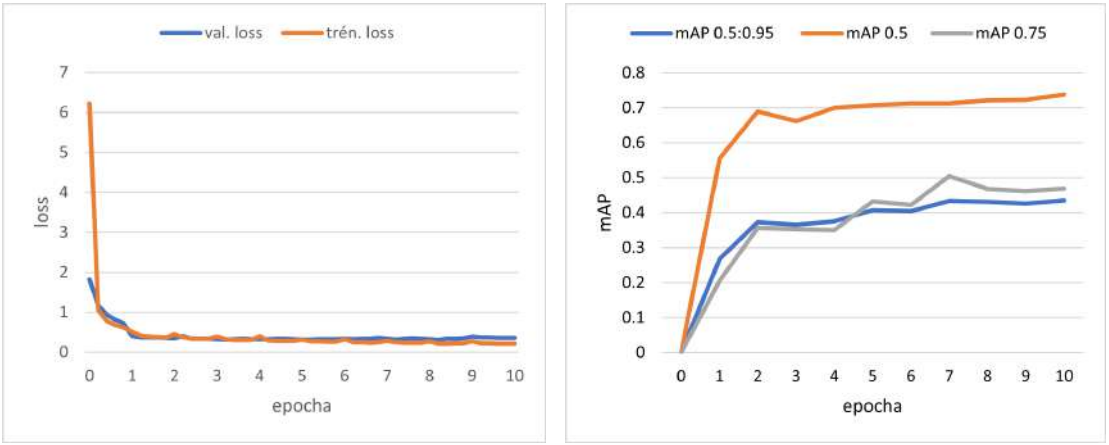
Ručne som pomocou nástroja Aletheia [10] anotovala 300 obrázkov z datasetu noviny. Pre konzistentnosť pri označovaní som si stanovila nasledujúce pravidlá:

1. jeden alebo viac riadkov textu sú označené ako nadpis ak sa vizuálne odlišujú od textu pod ním. Teda majú väčší alebo tučnejší font a nad ním, alebo pod ním sa nachádza medzera alebo vodorovná čiara,

2. nadpis a podnadpis sú oddelené, ak je medzi nimi jasne vidieť zmena fontu alebo veľkosti písma,

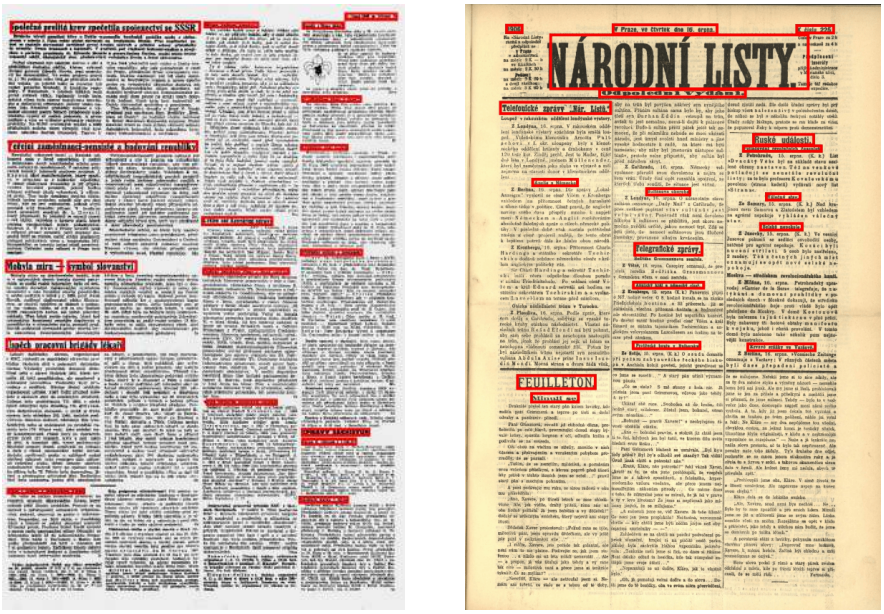
3. za nadpis je označený aj text, ktorý vyhovuje vizuálne podmienke číslo 1 ale logicky nadpisom nie je. Ide hlavne o text, ktorý je väčšinou väčší tučnejší a podčiarknutý, ako napríklad dátum, miesto vydania, cena časopisu atď. Tieto texty by vo výsledku mali byť detekované ako samostatné textové regióny, preto nie je problémom, aby sa ich sieť naučila detekovať a zároveň by to pre sieť malo byť jednoduchšie keďže

tieto texty vykazujú podobné znaky ako nadpisy (sú podčiarknuté alebo okolo nich je medzera).



(a) (b)

Obr. 6.11: Grafy reprezentujúce priebeh trénovania neurónovej siete na detekciu nadpisov: (a) validačná a trénovacia chyba, (b) mAP.



Obr. 6.12: Ukážky detekcie nadpisov

Ako prvý experiment s nadpismi som skúšala natrénovať sieť len so spomínanými 300 obrázkami. Presnosť $mAP^{0.50}$ sa však pohybovala len okolo 0.3 a po znížení LR okolo 0.5. Ako ďalší experiment som pri trénovaní okrem mojich ručne označených 300 obrázkov použila aj dataset PRImA [4]. Tento dataset obsahuje 478 obrázkov s anotáciami, ktoré sú zaradené do niekoľkých tried a bolo možné vytiahnuť si len anotácie ku nadpisom. Spolu som teda dostala 778 obrázkov, ktoré som znovu rozdelila v pomere 80:20 na trénovaciu sadu a validačnú sadu. Pri použití algoritmu Adam s počiatočnou hodnotou 0.0001 dosiahla

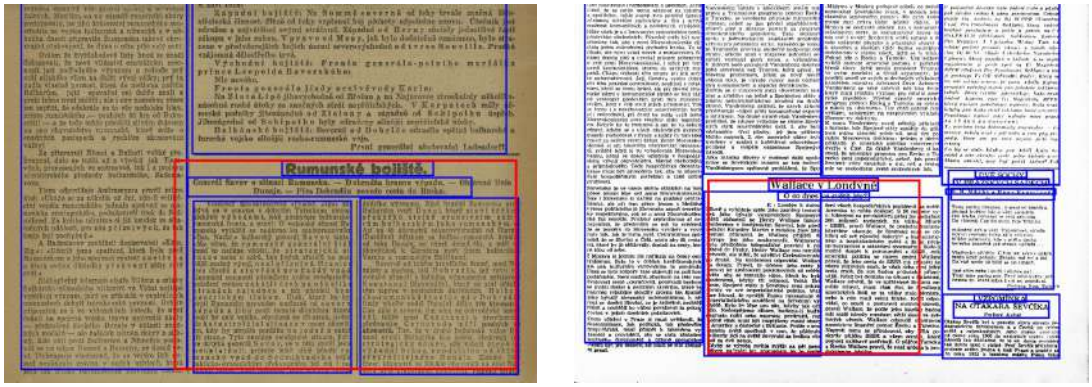
sieť $mAP^{0.50} = 0.738$. Priebeh trénovania siete na tejto rozšírenej sade je možné vidieť na obrázku 6.11.

6.6 Postprocessing

Po predikovaní slov, regiónov a nadpisov neurónovými sieťami sa v dátach vyskytovalo niekoľko opakujúcich sa chýb. Z tohto dôvodu je ďalším krokom v analýze layoutu postprocessing. Ide o úpravu hraníc regiónov, odstránenie prekrývajúcich sa regiónov a delenie regiónov.

Prekrývajúce sa textové regióny

Prvou opakujúcou sa chybou vo výsledkoch boli prekrývajúce sa regióny. Stávalo sa že relatívne dobre označené regióny boli prekryté jedným väčším región. Vo veľkej väčšine prípadov boli správne detekované práve malé regióny a odstránený mal byť veľký, ktorý ich prekrýval. Tento problém som vyriešila odstránením regiónov, ktorých plocha bola prekrytá z viac ako 60% inými regiónmi. Hodnota 60% bola zvolená, pretože dávala na testovacej sade najlepšie výsledky.



Obr. 6.13: Prekrývajúce sa textové regióny. Región vyznačený červenou farbou bol odstránený.

Malé regióny v oblasti nadpisov

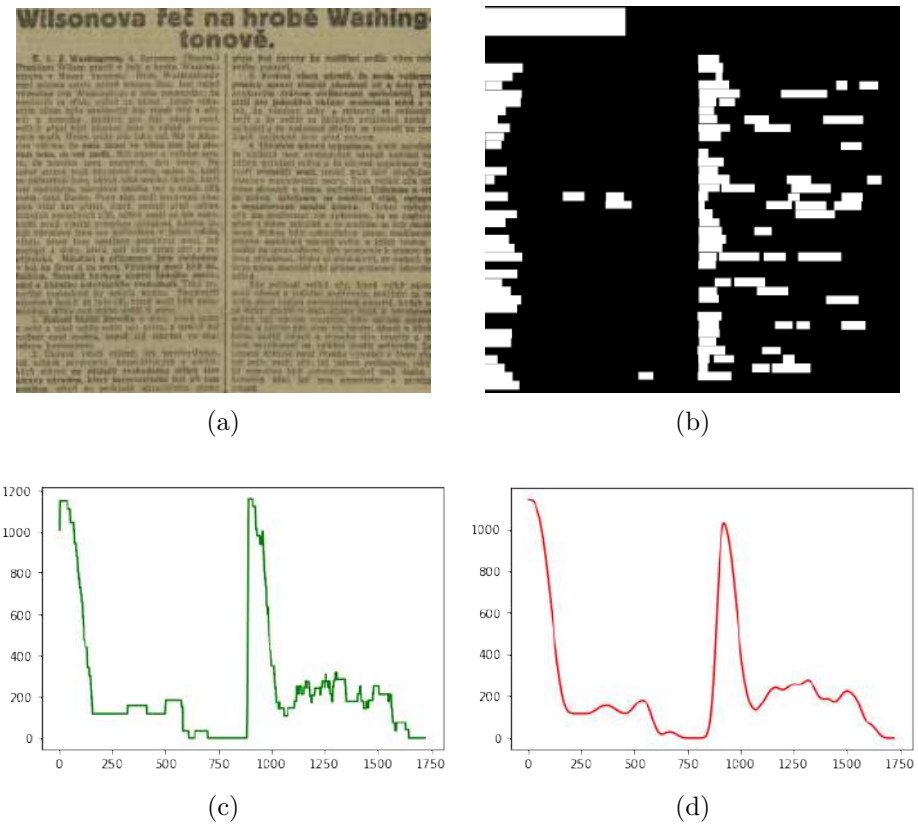
Obr. 6.14: Malé textové regióny v oblasti nadpisov: modrou sú vyznačené detekované textové regióny, tyrkysovou detekované nadpisy.

Pri veľkých nadpisoch občas dochádzalo k detekovaniu malých regiónov v jeho okolí. Tento jav bolo možné pozorovať už v ground truth dátach (obr. 4.2) a zaniesol sa aj do natrénovanej siete na detekciu regiónov. Detekcia nadpisov však fungovala solídne a detekovaný nadpis často tieto malé regióny prekryl. Preto sa v postprocessingu ako ďalšie odstránia

všetky textové regióny, ktorých plocha je aspoň na 70% prekrytá nadpisom. Táto hodnota dávala empiricky najlepšie výsledky. Tento krok taktiež rieši oblasti, ktoré boli zároveň detekované sieťou pre nadpisy aj sieťou pre regióny. To je možné pozorovať na obrázku 6.14 dole. V tomto prípade nejde o nesprávnu detekciu ani jednej zo sietí, avšak takéto duplikáty je vhodné odstrániť.

Textové regióny spojené horizontálne

Pri dvoch textových regiónoch, ktoré mali jeden spoločný nadpis (obr. 6.15a) občas sieť tieto regióny vyhodnotila ako jeden. Tento región je však možné na základe dát zo siete detekujúcej začiatky a konce riadkov rozdeliť. V úvahu som brala len slová na začiatku riadku pre daný región a vytvorila som binárnu masku (obr. 6.15b). Pre obrázok masky je vytvorená horizontálna projekcia, čo v tomto prípade znamená spočítanie bielych pixelov v každom stĺpci obrázka (obr. 6.15c). Graf je následne vyhladený gaussovým filtrom (obr. 6.15d), čím sa odstráni drobný šum. V tomto grafe je možné pozorovať 2 veľké lokálne maximá, ktoré nasvedčujú tomu, že v obrázku sa nachádzajú 2 textové regióny. Ak sa teda v grafe nachádza druhý vrchol, ktorý dosahuje aspoň 60% výšky prvého, región rozdelíme na dva v oblasti druhého vrcholu. To, že vertikálne hranice regiónu v mieste delenia nebudú úplne presné nám momentálne nevadí. Tieto hranice budú upravované v ďalšej časti postprocessingu.



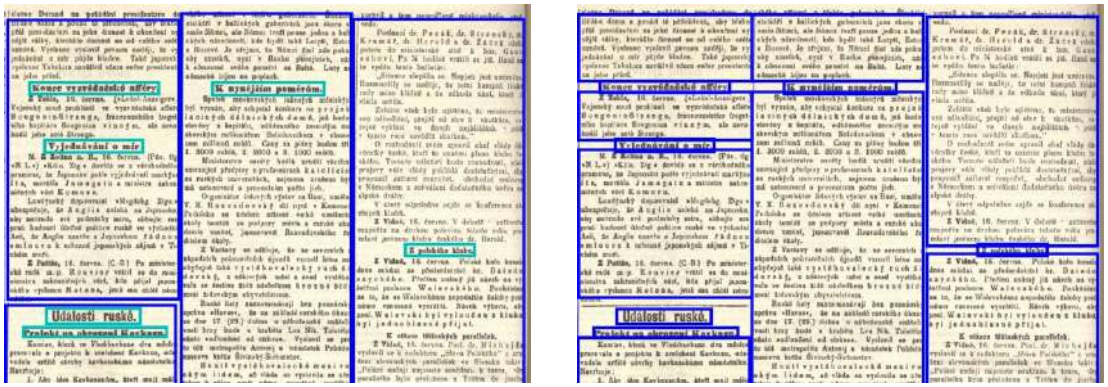
(a) (b)

(c) (d)

Obr. 6.15: Detekcia spojených regiónov: (a) dva horizontálne spojené textové regióny, (b) maska začiatočných slov riadkov, (c) horizontálna projekcia masky, (d) horizontálna projekcia vyhladená gaussom.

Textové regióny spojené vertikálne

Ďalším častým problémom bolo, že sieť na detekciu regiónov často spájala regióny, ktoré boli oddelené nadpisom. Tento problém sa taktiež vyskytoval už v ground truth dátach a preniesol sa aj do výsledkov. Na obrázku 6.16a je možné vidieť tyrkysovou výstup z detekcie nadpisov a modrou detekciu textových regiónov. Tieto dva výstupy bolo potrebné ďalej spracovať a regióny podľa nadpisov rozdeliť. Ak sa teda nadpis s regiónom prekrývajú a nadpis je približne v strede daného regiónu, región sa rozdelí na 2 časti. Výsledok po rozdelení takýchto regiónov zobrazuje obrázok 6.16b.



(a) (b)

Obr. 6.16: Regióny spojené vertikálne: (a) obrázok pred úpravou, (b) obrázok po úprave.

Iným prípadom bolo, ak sieť na detekciu textových regiónov správne regióny detekovala, avšak z vrchu alebo zo spodu do nich zahrnula aj nadpisy. V tomto prípade nie je potrebné región deliť na 2 časti, stačí ho skrátiť či už z hornej alebo spodnej časti, podľa toho kde sa nadpis nachádza. To, či sa región bude deliť na 2 časi alebo len redukovať je dané tým v ktorej časti regiónu sa nadpis nachádza. Ak je v horných alebo dolných 10% výšky regiónu, región sa skráti, inak sa rozdelí.

(a) (b)

Obr. 6.17: Nadpisy zahrnuté v regiónoch: (a) obrázok pred úpravou, (b) obrázok po úprave.

Úprava hraníc regiónov

Úprava dĺžky

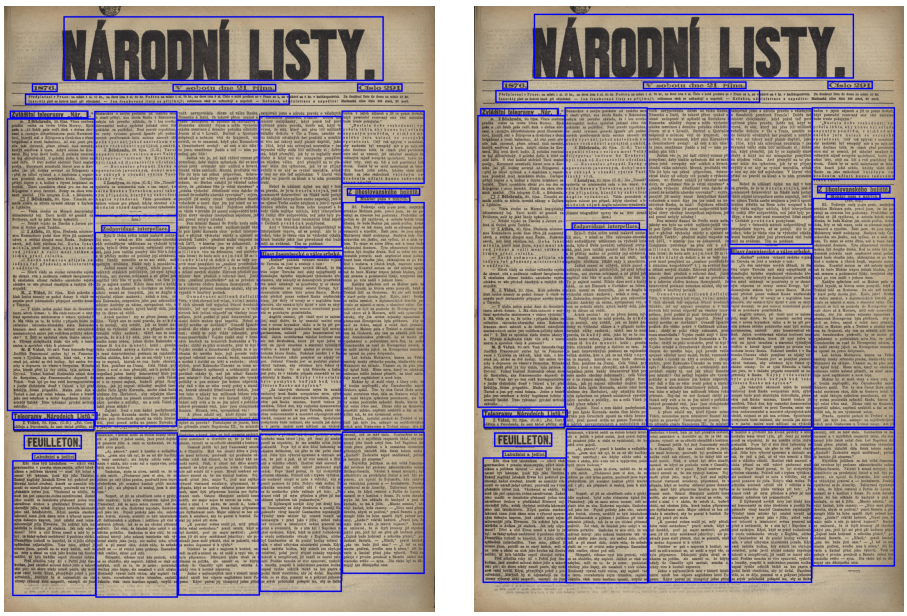
Pri úprave dĺžky textových regiónov môžu nastať dva prípady. Prvým je že región je príliš dlhý a zaberá aj pozadie, na ktorom sa nenachádzajú žiadne slová (obr. 6.18a vľavo dole).

Riešením tohto problému bolo nájsť všetky slová v danom regióne. Tieto slová boli výstupom zo siete, popísanej v časti 6.3. Slovo patrí do regiónu ak sa aspoň 80% jeho plochy nachádza v danom regióne. Následne sa nájde najvyššie a najnižšie slovo (s najnižšími a najvyššími y-ovými súradnicami) a podľa nich sa upraví horná a dolná hranica regiónu.

Druhým prípadom je že textový región je príliš krátky a neobsahuje všetky slová, ktoré by obsahovať mal. Tento problém je riešený následovne:

- najprv sa detekujú všetky slová, ktoré nepatria do žiadneho regiónu,
- slová sa zoradia podľa y-ových súradníc od najvrchnejších po najspodnejšie,
- vyberie sa najvrchnejšie slovo, nájde sa prvý región, ktorý sa nachádza nad ním a v danom regióne sa spočíta priemerná výška slov,
- slovo sa pridá do regiónu ak je jeho výška v maximálnom rozmedzí 10% od priemernej výšky slov v regióne a je od regiónu vzdialené maximálne 1 riadok (priemerná výška slov z daného regiónu)
- postup sa opakuje pre všetky nezaradené slová.

Týmto postupom sa textové regióny expandujú smerom nadol. Podobný postup sa aplikuje na expanziu smerom nahor. Rozdielom je že slová sa radia od najspodnejších po najvrchnejšie a hľadá sa pre nich región po nimi.



(a) (b)

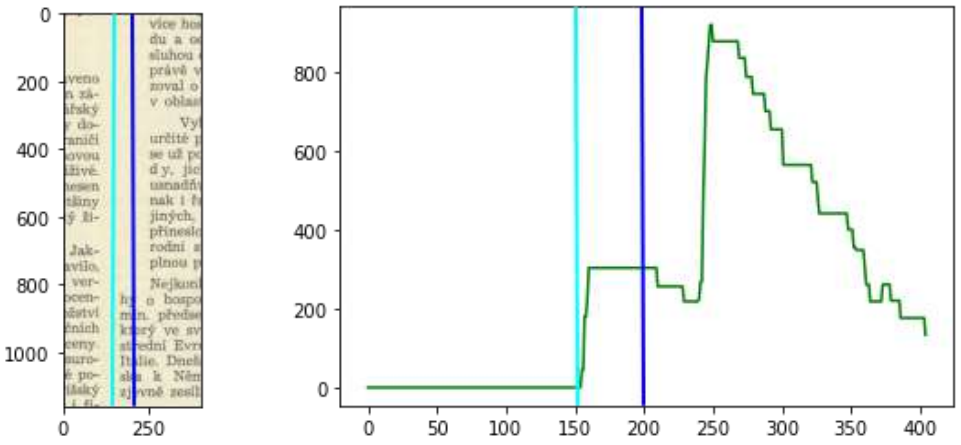
Obr. 6.18: Úprava dĺžky regiónov: (a) obrázok pred úpravou, (b) obrázok po úprave.

Úprava šírky

Pri úprave šírky textového regiónu sa vystrihne výsek okolo oboch vertikálnych hrán bounding boxu regiónu. Výsek okolo ľavej hrany regiónu (pravý horný región z obrázku 6.20a) je možné vidieť na obrázku 6.19a. Stredom tohto výseku je práve ľavá hrana a široký je 20%

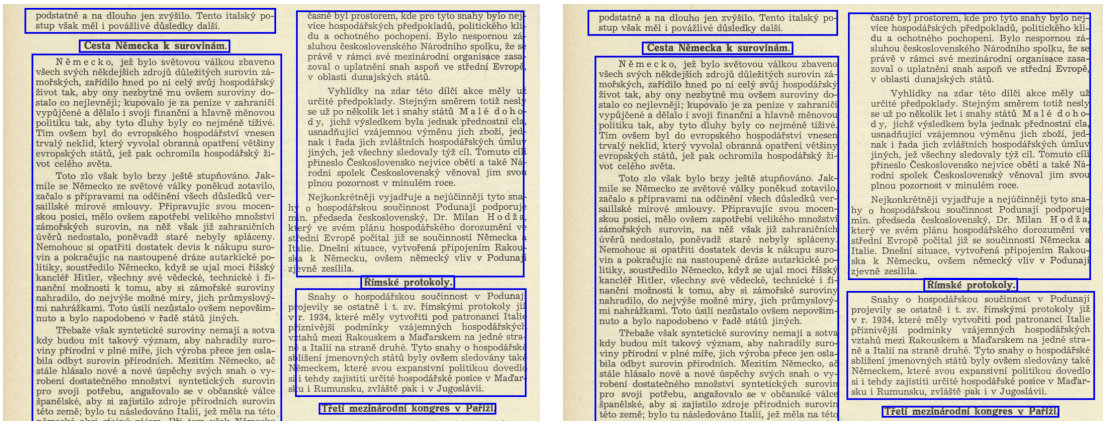
šírky pôvodného textového regiónu na každú stranu od stredu. Pre tento výsek sa vytvorí maska prvých slov v riadkoch podobne ako u delení horizontálne spojených regiónov popísaných vyššie. Pre masku sa následne vypočíta horizontálna projekcia (obr. 6.19b). Pre ľavú hranu regiónu sa postupne postupuje po hodnotách projekcie smerom doľava a ak sa klesne pod určitý prah, stanoví sa nová hranica regiónu. Tento prah som nastavila na 10% výšky pôvodného regiónu. Prah nie je nastavený na 0, pretože detekcia počiatočných slov riadkov nie je natoľko spoľahlivá a občas sa môže stať, že do tejto triedy bude zaradené posledné slovo riadku a tým by sa hranica regiónu posúvala až za toto nesprávne klasifikované slovo.

Obdobne sa postupuje pri pravej hranici textových regiónov, až na rozdiel, že horizontálna projekcia sa vytvorí z masky posledných slov v riadku a po grafe sa postupuje smerom doprava.



(a) (b)

Obr. 6.19: Úprava šírky regiónu: (a) výsek okolo ľavej hranice pravého horného textového regiónu z obrázku 6.20a, (b) horizntálna projekcia počiatočných slov v riadku pre daný výsek. Modrá čiara na oboch obrázkoch označuje pôvodnú hranicu regiónu, tyrkysová farba označuje hranicu po úprave.



(a) (b)

Obr. 6.20: Úprava šírky regiónu: (a) obrázok pred úpravou, (b) obrázok po úprave.

Kapitola 7

Evaluácia analýzy layoutu

Pri evaluácii môjho algoritmu bolo využitá metrika F-score popísaná v časti 3.7.

7.1 Výsledky

Na výslednú evaluáciu som si v nástroji Aletheia označila 100 obrázkov z datasetu Noviny, ktoré sa pri trénovaní sietí nachádzali vo validačnej sade. Vybrala som obrázky s rozličnými layoutami, okrem tých ktoré obsahovali reklamy. Pri reklamách bolo špecifické, že obsahovali veľké písmo, ktoré svojím vzhľadom evokuje nadpis, avšak v tomto prípade išlo len zvýraznené slovo, ktoré logicky patrí do textu. Reklamy taktiež obsahovali rôzne ornamenty, text ktorý nebol písaný vodorovne alebo obsahovali rôzne rámy. Niekedy som si preto ani ja pri označovaní nebola istá ako správne regióny zaznačiť. Príklady dokumentov s reklamami je možné vidieť na obrázku 7.1.

Obr. 7.1: Príklady dokumentov ktoré obsahujú reklamy.

Metriku F-score som merala na troch prahoch: 0.5, 0.75 a 0.9. Pri vyhodnocovaní som taktiež merala čas spracovania jedného obrázku na mojom domácom počítači s procesorom Intel Core i7-9750H a grafickou kartou NVIDIA GeForce GTX 1650. Priemerný čas bol

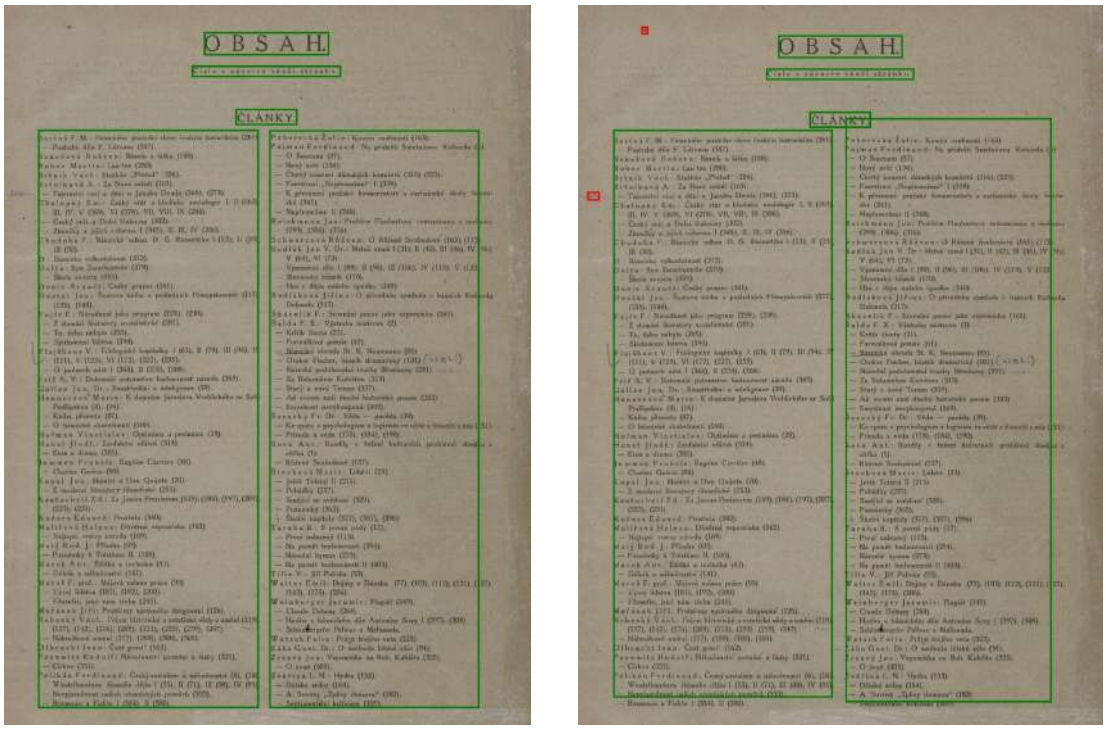
odmeraný na 100 obrázkoch, pričom modely siete už boli načítané na grafickej karte, meraná teda bola predikcia všetkých 3 sietí (respektíve 2 sietí v druhej variante) a postprocessing.

	F-score			**time [s]**
	prah 0.5	**prah 0.75**	**prah 0.9**	
slová + regióny + nadpisy + postprocessing	79.57 %	68.38 %	38.3 %	7.429
regióny + nadpisy + postprocessing	78.75 %	67.9 %	35.86 %	0.883
regióny + nadpisy	49.65 %	43.66 %	23.72 %	0.87

Tabuľka 7.1: Výsledky.

V tabuľke 7.1 je možné vidieť výsledky môjho riešenia. V prvom riadku je prístup, ktorý využíva výstupy zo sietí na detekciu nadpisov, regiónov a slov klasifikovaných podľa pozície v riadku. Je možné všimnúť si, že tento prístup trvá cez 7 sekúnd a to kvôli predikcii slov, kde je obrázok potrebné rozdeliť na niekoľko častí a každú detekovať zvlášť.

V druhom riadku je verzia, kde sa využívajú výstupy len zo sietí na detekciu nadpisov a regiónov. Z postprocessingu sú potom vynechané kroky, kde sa využívali informácie o slovách. Konkrétne ide o úpravu hraníc regiónov a delenie regiónov horizontálne. Na výslednú presnosť to vplýva len minimálne, zato tento prístup je niekoľko násobne rýchlejší.



(a) (b)

Obr. 7.2: Ukážky výstupov z prvých dvoch prístupov: (a) prístup s detekciou slov, (b) prístup, ktorý využíva len detekciu nadpisov a regiónov.

Na obrázku 7.2 je možné porovnať výsledky z obidvoch prístupov. Zelenou sú vyznačené regióny, ktoré boli detekované správne, červenou sú vyznačené regióny detekované nesprávne

a nevyznačené ostali regióny, ktoré neboli detekované vôbec (tie sa však vo výsledkoch často neobjavujú). Pri postprocessingu na obrázku vľavo sa využívali aj informácie o slovách, teda boli odstránené detekcie, ktoré neobsahovali žiadne slovo a regióny boli upravené tak, aby obsahovali aj nezaradené slová (na obrázku dole, pravá časť druhého bounding boxu nebola upravená z dôvodu nedostatočného počtu posledných slov riadkov).

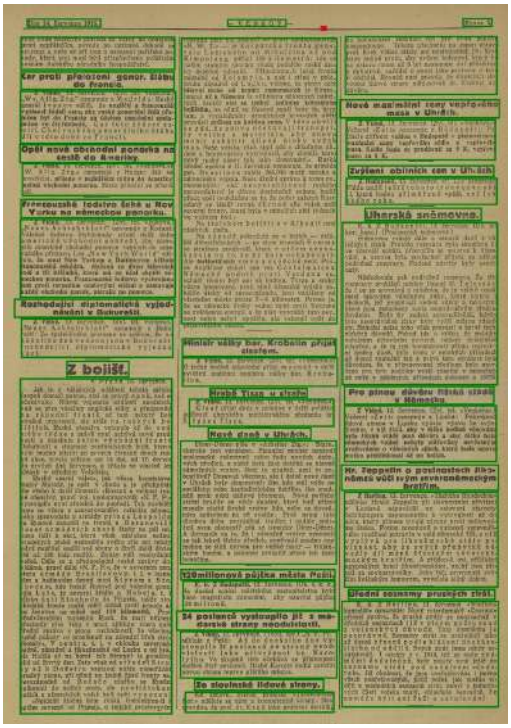
V poslednom riadku tabuľky je prístup kde sa vynecháva postprocessing. Detekované nadpisy a regióny sa jednoducho spoja bez ďalšieho pretriedenia alebo úprav. V tomto prípade je už v presnosti vidieť markantný rozdiel, zatiaľ čo čas zostáva približne rovnaký.

Za výsledné riešenie preto považujem ten, ktorý využíva len siete na detekciu regiónov a nadpisov a na záver prevádza postprocessing. Tento prístup predstavuje kompromis medzi presnosťou a rýchlosťou. Problémom je, že tento prístup občas vynecháva celé slová alebo riadky, čo by vo veľa aplikáciách mohlo predstavovať problém. Riešením by v budúcnosti mohol byť prístup k detekcii slov, ktorý by bol rýchlejší, alebo slová detekovať len v okolí hraníc detekovaných regiónov.

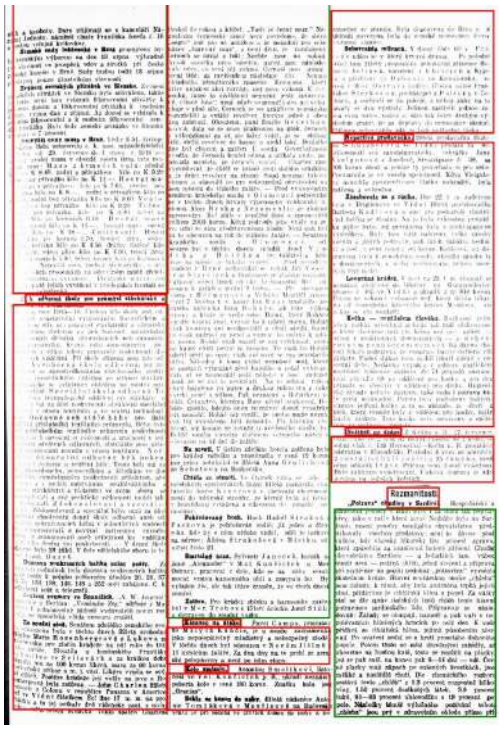
Ukážky výsledkov z tohto prístupu je možné vidieť na obrázku 7.3. Ďalšie ukážky je možné vidieť v prílohe A. Obrázok obsahuje 2 presné a 2 nepresné detekcie textových regiónov podľa metriky F-score s prahom 0.5. Jav, ktorý je možné pozorovať na viacerých fotkách je zámena tučnejších slov na začiatku riadku za nadpis a následné nesprávne rozdelenie regiónu týmto nadpisom v postprocessingu. Vidieť to je možné aj na obrázku 7.3c.

Na ilustráciu chybovosti datasetu Noviny, na ktorom som trénovala neurónové siete, som touto metrikou vyhodnotila 100 mojich ručne označených testových obrázkov s XML anotáciami daných obrázkov z datasetu Noviny. Vyšli mi hodnoty $F\text{-}score^{0.5} = 49.72$ %, $F\text{-}score^{0.75} = 35.93$ % a $F\text{-}score^{0.9} = 26.14$ %. Výsledky môjho algoritmu teda dosahujú lepšiu presnosť ako pôvodné dáta použité na trénovanie. Ukážky z datasetu Noviny, ktoré boli chybne označené je možné vidieť na obrázku 4.2.

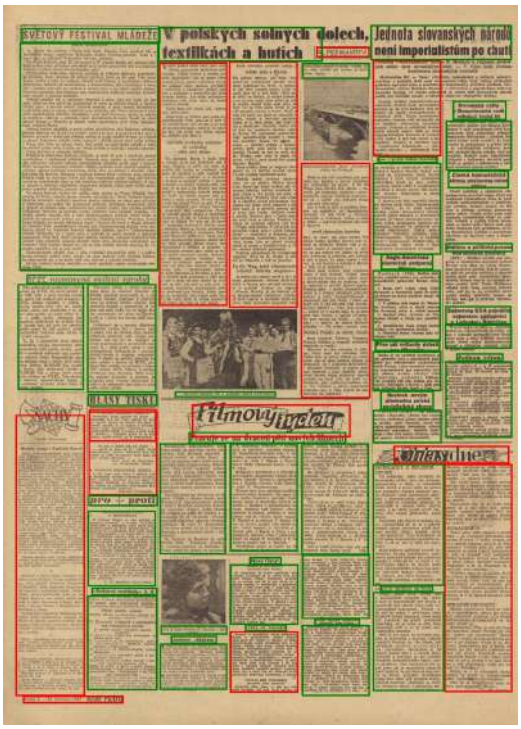
(a)



(b)



(c)



(d)

Obr. 7.3: Výsledné textové regióny. Zelenou sú označené regióny, ktoré boli detekované správne a červenou regióny detekované nesprávne za použitia metriky F-score s prahom 0.5.

Kapitola 8

Záver

Cieľom tejto diplomovej práce bolo navrhnúť a implementovať algoritmus pre analýzu rozloženia textu v historických dokumentoch so zložitým rozdelením. Ako prvé som si urobila prehľad v existujúcich prístupoch, prenikla do problematiky analýzy layoutu dokumentu a vyhľadala a pripravila vhodné datasety. Ďalej som navrhla riešenie, v ktorom som sa rozhodla využiť neurónové siete. Preštudovala som si existujúce architektúry detekčných neurónových sietí a vybrala Faster-RCNN, ktorá sa mi javila ako najvhodnejšia pre riešenie mojej úlohy.

V rámci práce som s touto architektúrou urobila niekoľko experimentov a výsledkom sú 4 natrénované neurónové siete. Prvými dvoma sieťami sú sieť na detekciu slov a sieť, ktorá tieto slová dokáže klasifikovať do kategórii podľa pozície slova v riadku. Obe tieto siete dosahovali $mAP^{0.50}$ takmer 0.8. Samotná detekcia slov síce vo finálnom riešení nie je použitá, avšak využiť by sa dala pri tvorbe výstupného XML vo formáte ALTO. Tento formát okrem pozície textových blokov môže popisovať aj pozície jednotlivých slov.

Ďalšími sieťami s o niečo horšou presnosťou sú sieť na detekciu nadpisov s $mAP^{0.50} = 0.74$ a sieť na detekciu regiónov s $mAP^{0.50} = 0.57$.

Následne som na základe spozorovaných často sa opakujúcich chýb spracovala výstupy zo sietí tak, aby program dosahoval čo najlepších výsledkov. Podľa metriky F-score, ktorú som implementovala s 3 rôznymi prahmi dosahuje moja metóda presnosť $F\text{-}score^{0.50} = 78.75\,\%$, $F\text{-}score^{0.75} = 67.9\,\%$ a $F\text{-}score^{0.90} = 35.86\,\%$ s priemerným časom 0.88s na jeden obrázok.

U iných existujúcich riešení sa pohybujú presnosti analýzy layoutu od 70% do 99%, s priemerom približne 90%. Tieto prístupy je však ťažké porovnávať, pretože vyhodnocované boli na rôznych datasetoch a s rôznymi metrikami. S týmito riešeniami je taktiež náročné porovnať môj prístup, keďže dataset Noviny, s ktorým som pracovala nie je verejným datasetom a teda s ním nikto iný nepracoval. Myslím si, že môj program sa nevyrovná state-of-art algoritmom na analýzu layoutu, ktoré dosahujú vysoké presnosti, avšak pri iných menej presných prístupoch môže byť konkurencie schopný.

Možnosti vylepšenia algoritmu vidím v pretrénovaní sietí na lepších a rozsiahlejších dátach. Detekcia nadpisov bola totiž trénovaná len na 778 obrázkoch a dataset na trénovanie regiónov obsahoval chyby, pričom niektoré z nich sa zaniesli aj do natrénovanej siete. Problémom sú taktiež textové regióny, ktoré nemajú obdĺžnikovité textové regióny, v tomto prípade by mohol byť využitý napríklad algoritmus alpha shape. Ďalším vylepšením by mohla byť detekcia slov iným prístupom, ktorý by bol rýchlejší. Algoritmus považujem za využiteľný v praxi v prípade, žeby výsledky boli ešte vizuálne prekontrolované a prípadné nesprávne detekované regióny ručne opravené. Niektoré obrázky totiž vykazujú vyššiu chybovosť, čo znamená že niektoré regióny nedetekujú, či spoja alebo občas neobsiahnu všetky

slová regiónu. To by mohol byť problém pri niektorých aplikáciách ako napríklad automatickom porozumení textu, určení poradia čítania alebo vyhľadávaní textových reťazcov. Potenciál využitia takéhoto algoritmu je napríklad v programoch na digitalizáciu historických dokumentov knižnicami.

Literatúra

[1] *Matematický model a aktivní dynamika neuronu* [online]. Dostupné z: https://portal.matematickabiologie.cz/index.php?pg=analyza-a-hodnoceni-biologickych-dat--umela-inteligence--neuronove-site-jednotlivy-neuron--jednotlivy-neuron--matematicky-model-a-aktivni-dynamika-neuronu.

[2] *Selective Search for Object Detection | R-CNN* [online]. 2020 [cit. 2020-12-26]. Dostupné z: https://www.geeksforgeeks.org/selective-search-for-object-detection-r-cnn/.

[3] Aljaafari, N. *Ichthyoplankton Classification Tool using Generative Adversarial Networks and Transfer Learning*. Dizertačná práca.

[4] Antonacopoulos, A., Bridson, D., Papadopoulos, C. a Pletschacher, S. A Realistic Dataset for Performance Evaluation of Document Layout Analysis. In:. Január 2009, s. 296–300. DOI: 10.1109/ICDAR.2009.271.

[5] Antonacopoulos, A. a Ritchings, R. Representation and classification of complex-shaped printed regions using white tiles. In:. Január 1995, s. 1132–1135. DOI: 10.1109/ICDAR.1995.602119.

[6] Bataineh, B., Abdullah, S. a Omar, K. Generating an Arabic Calligraphy Text Blocks for Global Texture Analysis. *International Journal on Advanced Science, Engineering and Information Technology*. Január 2011, zv. 1. DOI: 10.18517/ijaseit.1.2.33.

[7] Binmakhashen, G. M. a Mahmoud, S. A. Document Layout Analysis: A Comprehensive Survey. *ACM Comput. Surv.* New York, NY, USA: Association for Computing Machinery. október 2019, zv. 52, č. 6. DOI: 10.1145/3355610. ISSN 0360-0300. Dostupné z: https://doi.org/10.1145/3355610.

[8] Bukhari, S., Breuel, T. M., Asi, A. a El Sana, J. Layout Analysis for Arabic Historical Document Images Using Machine Learning. In: *2012 International Conference on Frontiers in Handwriting Recognition (ICFHR 2012)*. Los Alamitos, CA, USA: IEEE Computer Society, Sep 2012, s. 639–644. DOI: 10.1109/ICFHR.2012.227. Dostupné z: https://doi.ieeecomputersociety.org/10.1109/ICFHR.2012.227.

[9] Chen, K., Seuret, M., Hennebert, J. a Ingold, R. Convolutional Neural Networks for Page Segmentation of Historical Document Images. In: *14th IAPR International Conference on Document Analysis and Recognition, ICDAR 2017, Kyoto, Japan, November 9-15, 2017*. IEEE, 2017, s. 965–970. DOI: 10.1109/ICDAR.2017.161. Dostupné z: https://doi.org/10.1109/ICDAR.2017.161.

[10] Clausner, C., Pletschacher, S. a Antonacopoulos, A. *Aletheia - Document Analysis System*. 2011. Dostupné z: https://www.primaresearch.org/tools/Aletheia.

[11] COCO. *Detection Evaluation* [online]. 2020 [cit. 2021-1-31]. Dostupné z: https://cocodataset.org/#detection-eval.

[12] Cohen, R., Asi, A., Kedem, K., El Sana, J. a Dinstein, I. Robust Text and Drawing Segmentation Algorithm for Historical Documents. In: *Proceedings of the 2nd International Workshop on Historical Document Imaging and Processing*. New York, NY, USA: Association for Computing Machinery, 2013, s. 110–117. HIP '13. DOI: 10.1145/2501115.2501117. ISBN 9781450321150. Dostupné z: https://doi.org/10.1145/2501115.2501117.

[13] Context, C. O. in. *Detection Evaluation*. Dostupné z: https://cocodataset.org/#detection-eval.

[14] Dubey, V. *Evaluation Metrics for Object detection algorithms* [online]. Medium, 2020. Dostupné z: https://medium.com/@vijayshankerdubey550/evaluation-metrics-for-object-detection-algorithms-b0d6489879f3.

[15] Fornés, A., Romero, V., Baro, A. a Lladós, J. *Information Extraction in Historical Handwritten Records* [online]. Robust Reading Competition, 2017. Dostupné z: https://rrc.cvc.uab.es/?ch=10&com=introduction.

[16] Fukusima, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*. 1980, s. 193–202. Dostupné z: https://doi.org/10.1007/BF00344251.

[17] Gatos, B., Antonacopoulos, A. a Stamatopoulos, N. ICDAR2007 Handwriting Segmentation Contest. *Document Analysis and Recognition, International Conference on*. September 2007, zv. 2, s. 1284–1288. DOI: 10.1109/ICDAR.2007.4377122.

[18] Ghosh, A., Sufian, A., Sultana, F., Chakrabarti, A. a De, D. Fundamental Concepts of Convolutional Neural Network. In:. Január 2020, s. 519–567. DOI: 10.1007/978-3-030-32644. ISBN 978-3-030-32643-2.

[19] Girshick, R. B. Fast R-CNN. *CoRR*. 2015, abs/1504.08083. Dostupné z: http://arxiv.org/abs/1504.08083.

[20] Girshick, R. B., Donahue, J., Darrell, T. a Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*. 2013, abs/1311.2524. Dostupné z: http://arxiv.org/abs/1311.2524.

[21] Grippe, J. *The Project Behind a Front Page Full of Names* [online]. The New York Times, 2020 [cit. 2021-01-10]. Dostupné z: https://www.nytimes.com/2020/05/23/reader-center/coronavirus-new-york-times-front-page.html.

[22] grt. *Princip diskrétní dvourozměrné konvoluce* [online]. 2006. Dostupné z: https://cs.wikipedia.org/wiki/Konvoluce#/media/Soubor:Konvoluce_2rozm_diskretni.jpg.

[23] Grüning, T., Leifert, G., Strauss, T., Michael, J. a Labahn, R. A two-stage method for text line detection in historical documents. *International Journal on Document Analysis and Recognition (IJDAR)*. Springer Science and Business Media

LLC. Jul 2019, zv. 22, č. 3, s. 285–302. DOI: 10.1007/s10032-019-00332-1. ISSN 1433-2825. Dostupné z: http://dx.doi.org/10.1007/s10032-019-00332-1.

[24] Ha, J., Haralick, R. a Phillips, I. Document page decomposition by the bounding-box project. In:. Január 1995, s. 1119–1122. DOI: 10.1109/ICDAR.1995.602115.

[25] Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A. et al. Speed/accuracy trade-offs for modern convolutional object detectors. *CoRR*. 2016, abs/1611.10012, [cit. 2019-12-25]. Dostupné z: http://arxiv.org/abs/1611.10012.

[26] Hubel, D. a T., W. Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*. 1959, s. 574–591. Dostupné z: https://physoc.onlinelibrary.wiley.com/doi/abs/10.1113/jphysiol.1959.sp006308.

[27] Journet, N., Ramel, J.-Y., Mullot, R. a Eglin, V. Document Image Characterization Using a Multiresolution Analysis of the Texture: Application to Old Document. *IJDAR*. Október 2008, zv. 11, s. 9–18. DOI: 10.1007/s10032-008-0064-6.

[28] Kingma, D. a Ba, J. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*. December 2014.

[29] Kise, K. Page Segmentation Techniques in Document Analysis. In: Doermann, D. a Tombre, K., ed. *Handbook of Document Image Processing and Recognition*. London: Springer London, 2014, s. 135–175. DOI: 10.1007/978 − 0 − 85729 − 859 − 1_5. ISBN 978-0-85729-859-1. Dostupné z: https://doi.org/10.1007/978-0-85729-859-1_5.

[30] Kise, K., Sato, A. a Iwata, M. Segmentation of Page Images Using the Area Voronoi Diagram. *Computer Vision and Image Understanding*. 1998, zv. 70, č. 3, s. 370 – 382. DOI: https://doi.org/10.1006/cviu.1998.0684. ISSN 1077-3142. Dostupné z: http://www.sciencedirect.com/science/article/pii/S1077314298906841.

[31] Krizhevsky, A., Sutskever, I. a Hinton, G. ImageNet Classification with Deep Convolutional Neural Networks. *Neural Information Processing Systems*. Január 2012, zv. 25. DOI: 10.1145/3065386.

[32] Kvasnička, V., Beňušková, L., Pospíchal, J., Farkaš, I., Tiňo, P. et al. *Úvod do teórie neurónových sietí*. Iris, 1997. ISBN 9788088778301.

[33] Kölsch, A. *Handwritten Annotation Detection Dataset (AnnotationDB)* [online]. TC-11 Online Resources, 2018. Dostupné z: http://tc11.cvc.uab.es/datasets/AnnotationDB_1.

[34] Liu, W., Anguelov, D., Erhan, D. a Szegedy, C. SSD: Single Shot MultiBox Detector. In: *Computer Vision and Pattern Recognition*. 2016 [cit. 2019-12-25]. DOI: 10.1007/978-3-319-46448. Dostupné z: https://arxiv.org/pdf/1512.02325.pdf.

[35] Mehri, M., Gomez Krämer, P., Héroux, P., Boucher, A. a Mullot, R. Texture Feature Evaluation for Segmentation of Historical Document Images. In:. New York, NY, USA: Association for Computing Machinery, 2013, s. 102–109. HIP '13. DOI: 10.1145/2501115.2501121. ISBN 9781450321150. Dostupné z: https://doi.org/10.1145/2501115.2501121.

[36] Namboodiri, A. a Jain, A. Document Structure and Layout Analysis. In:. Marec 2007, s. 29–48. DOI: 10.1007/978 − 1 − 84628 − 726 − 8_2. ISBN 978-1-84628-501-1.

[37] Navlani, A. *Neural Network Models in R* [online]. 2019. Dostupné z: https://www.datacamp.com/community/tutorials/neural-network-models-r.

[38] Nesbitt's Nursery, I. *Spruce* [online]. 2016 [cit. 2021-01-10]. Dostupné z: http://www.nesbittsnursery.com/spruce.html.

[39] O'Gorman, L. The document spectrum for page layout analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 1993, zv. 15, č. 11, s. 1162–1173. DOI: 10.1109/34.244677.

[40] Perera, A. *What is Padding in Convolutional Neural Network's(CNN's) padding* [online]. 2018. Dostupné z: https://ayeshmanthaperera.medium.com/what-is-padding-in-cnns-71b21fb0dd7.

[41] PyTorch. *TORCHVISION*. Dostupné z: https://pytorch.org/vision/stable/index.html.

[42] Raju, R. *Convolutional Neural Network | Deep Learning* [online]. Dostupné z: https://developersbreach.com/convolution-neural-network-deep-learning/.

[43] Redmon, J., Divvala, S., Girshick, R. a Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, s. 779–788 [cit. 2019-12-25]. DOI: 10.1109/CVPR.2016.91. Dostupné z: https://ieeexplore.ieee.org/document/7780460.

[44] Redmon, J. a Farhadi, A. YOLO9000: Better, Faster, Stronger. *CoRR*. 2016, abs/1612.08242. Dostupné z: http://arxiv.org/abs/1612.08242.

[45] Redmon, J. a Farhadi, A. YOLOv3: An Incremental Improvement. *CoRR*. 2018, abs/1804.02767. Dostupné z: http://arxiv.org/abs/1804.02767.

[46] Ren, S., He, K., Girshick, R. B. a Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *CoRR*. 2015, abs/1506.01497. Dostupné z: http://arxiv.org/abs/1506.01497.

[47] research, P. *Performance Evaluation: A framework for Performance Analysis of OCR methods* [online]. Dostupné z: https://www.primaresearch.org/tools/PerformanceEvaluation.

[48] Saabni, R. a El-Sana, J. Language-Independent Text Lines Extraction Using Seam Carving. In: *2011 International Conference on Document Analysis and Recognition*. 2011, s. 563–568. DOI: 10.1109/ICDAR.2011.119.

[49] Shi, Z. a Govindaraju, V. Line separation for complex document images using fuzzy runlength. *First International Workshop on Document Image Analysis for Libraries, 2004. Proceedings.* 2004, s. 306–312.

[50] Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S. et al. Going deeper with convolutions. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015, s. 1–9. DOI: 10.1109/CVPR.2015.7298594. ISSN 1063-6919. Dostupné z: https://ieeexplore.ieee.org/document/7298594.

[51] Tran, T. A., Na, I.-S. a Kim, S.-H. Hybrid Page Segmentation Using Multilevel Homogeneity Structure. In:. New York, NY, USA: Association for Computing Machinery, 2015. IMCOM '15. DOI: 10.1145/2701126.2701138. ISBN 9781450333771. Dostupné z: https://doi.org/10.1145/2701126.2701138.

[52] Wahl, F. M., Wong, K. Y. a Casey, R. G. Block segmentation and text extraction in mixed text/image documents. *Computer Graphics and Image Processing*. 1982, zv. 20, č. 4, s. 375 – 390. DOI: https://doi.org/10.1016/0146-664X(82)90059-4. ISSN 0146-664X. Dostupné z: http://www.sciencedirect.com/science/article/pii/0146664X82900594.

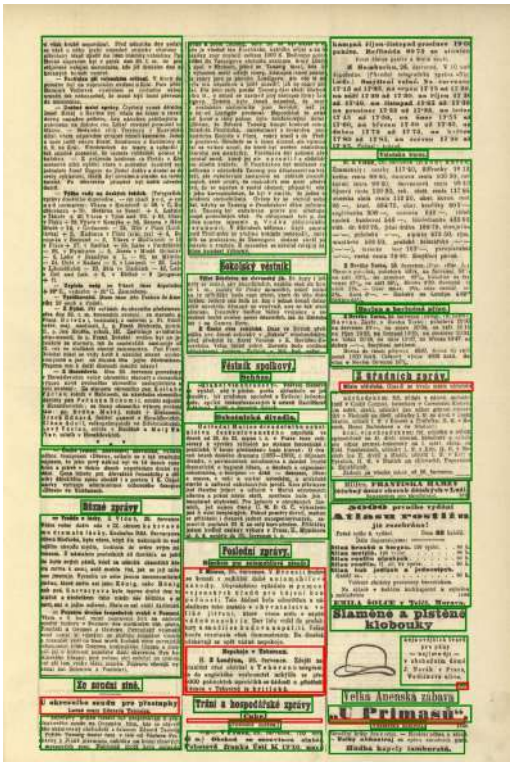
[53] Xiao, Y. a Yan, H. Text region extraction in a document image based on the Delaunay tessellation. *Pattern Recognition*. Marec 2003, zv. 36, s. 799–809. DOI: 10.1016/S0031-3203(02)00082-1.

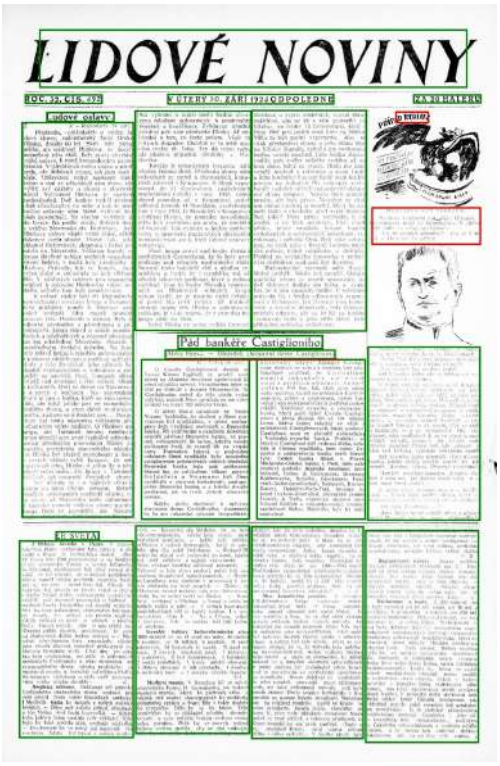
[54] Zharikov, I., Nikitin, F., Vasiliev, I. a Dokholyan, V. DDI-100: Dataset for Text Detection and Recognition. *CoRR*. 2019, abs/1912.11658. Dostupné z: http://arxiv.org/abs/1912.11658.

[55] Šíma, J. a Neruda, R. *Teoretické otázky neuronových sítí*. Matfyzpress, 1996. 390 s. ISBN 80-85863-18-9.

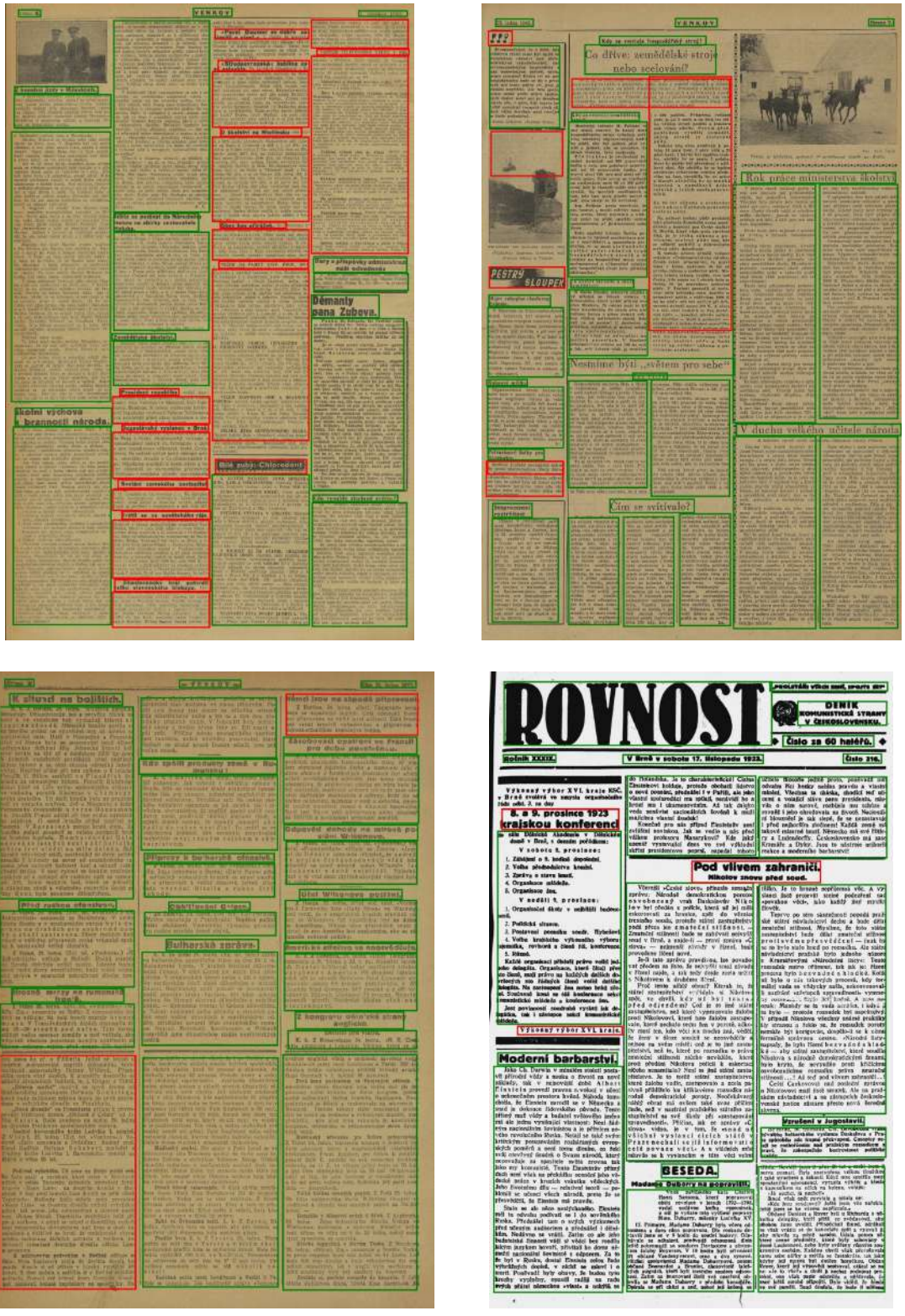
Príloha A

Ukážky detekcie textových regiónov

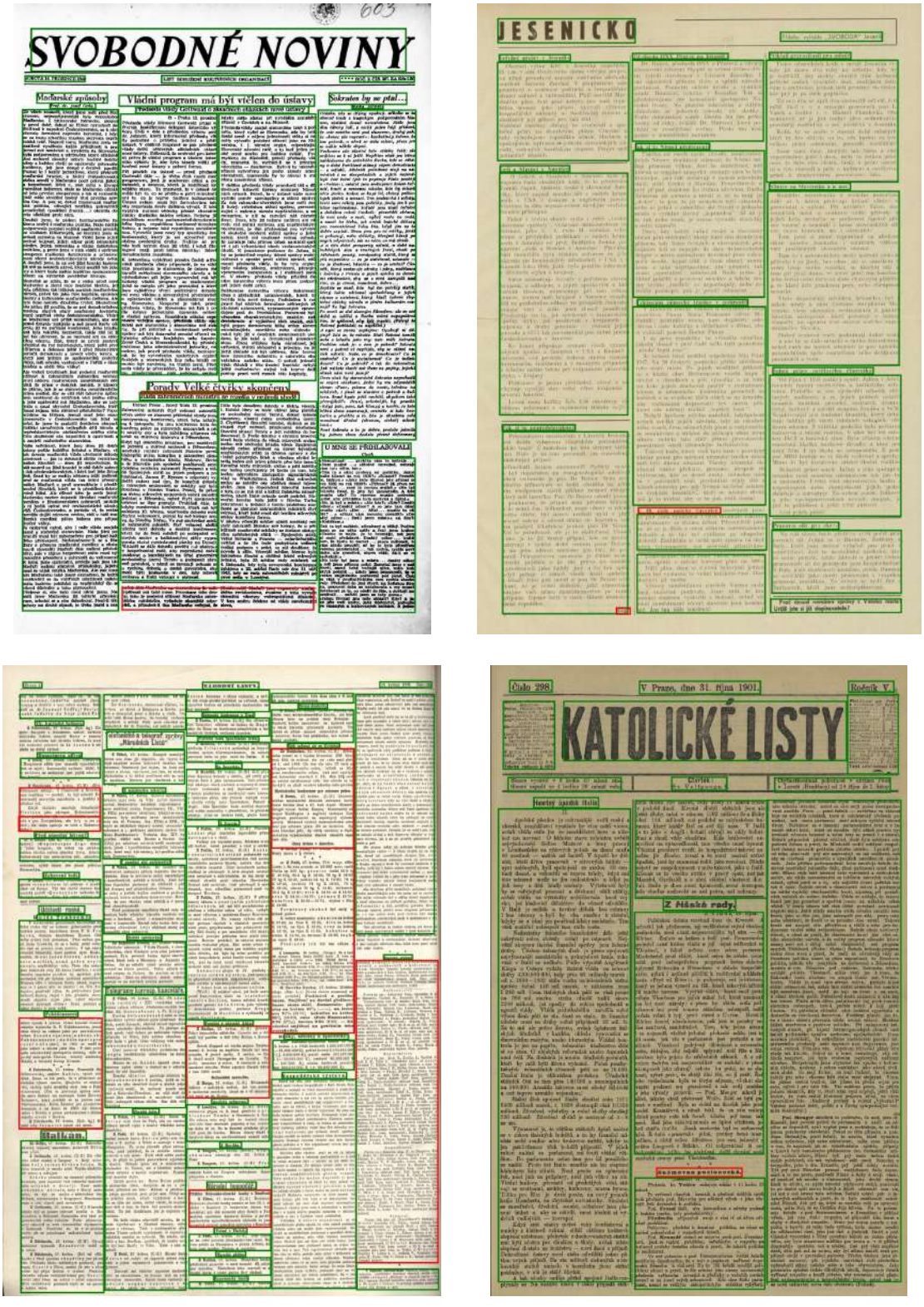




Obr. A.1: Výsledné textové regióny. Zelenou sú označené regióny, ktoré boli detekované správne a červenou regióny detekované nesprávne za použitia metriky F-score s prahom 0.5.



Obr. A.2: Výsledné textové regióny. Zelenou sú označené regióny, ktoré boli detekované správne a červenou regióny detekované nesprávne za použitia metriky F-score s prahom 0.5.



Obr. A.3: Výsledné textové regióny. Zelenou sú označené regióny, ktoré boli detekované správne a červenou regióny detekované nesprávne za použitia metriky F-score s prahom 0.5.

Príloha B

Obsah DVD

Priložené DVD obsahuje:

- *latex/* - zdrojové súbory tejto práce pre LaTeX
- *src/upravaBB/* - program na úpravu bounding boxov slov
- *src/detekcia_slov/* - program na detekciu slov v dokumente
- *src/trenovanie_sieti/* - jupyter notebooky so zdrojovým kódom tu trénovaniu sietí
- *src/analyza_layoutu/* - výsledný program na detekciu textových regiónov
- *ukazky_vystupov/* - obrázky s detekovanými slovami a textovými regiónmi
- *plagat.pdf* - plagát reprezentujúci výsledky mojej práce
- *DP_xpalac03.pdf* - pdf súbor obsahujúci text tejto práce