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

GRAFOVÉ NEURÓNOVÉ SIETE PRE ANALÝZU ROZLOŽENIA STRÁNOK

PAGE LAYOUT ANALYSIS WITH GRAPH NEURAL NETWORKS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. MATEJ OTČENÁŠ**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. MICHAL HRADIŠ, Ph.D.**
SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání diplomové práce

157993

Ústav:	Ústav počítačové grafiky a multimédií (UPGM)
Student:	**Otčenáš Matej, Bc.**
Program:	Informační technologie a umělá inteligence
Specializace:	Strojové učení
Název:	**Grafové neuronové sítě pro analýzu rozložení stránek**
Kategorie:	Zpracování obrazu
Akademický rok:	2023/24

Zadání:

1. Seznamte se s konvolučními a grafovými neuronovými sítěmi.
2. Vytvořte si přehled o současných metodách pro analýzu rozložení stránek a o dostupných datových sadách.
3. Navrhněte metodu nebo rozšiřte existující metodu pro analýzu rozložení stránek.
4. Připravte si vhodnou datovou sadu a proveďte na ní experimenty se základním referenčním přístupem.
5. Implementujte navrženou metodu a vyhodnoťte ji na datové sadě.
6. Porovnejte dosažené výsledky s jinými metodami a diskutujte možná budoucí rozšíření.
7. Vytvořte stručné video prezentující vaši práci, její cíle a výsledky.

Literatura:

- Jose Ramón Prieto and Enrique Vidal: Improved Graph Methods for Table Layout Understanding. ICDAR, 2021.
- Clausner et al.: ICDAR2019 Competition on Recognition of Documents with Complex Layouts - RDCL2019. ICDAR 2019.
- Yepes et al.: ICDAR 2021 Scientific Literature Parsing Competition, ICDAR 2021.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce:	**Hradiš Michal, Ing., Ph.D.**
Vedoucí ústavu:	Černocký Jan, prof. Dr. Ing.
Datum zadání:	1.11.2023
Termín pro odevzdání:	17.5.2024
Datum schválení:	9.11.2023

Abstrakt

Cieľom tejto práce je experimentálne otestovať silu grafových neurónových sietí pri komplexnej analýze rozloženia dokumentov. Z pohľadu typov dokumentov sú cieľom predovšetkým novinové články a historické spisy, ako napríklad ručne písané knihy alebo stredoveké manuskripty. Tie sa totižto vyznačujú komplexnosťou svojho rozloženia, nemajú pevne danú štruktúru alebo je samotný text veľmi členitý. Práca sa zaoberá vytvorením vhodných datasetov, ktoré slúžia na natrénovanie a otestovanie prístupu pre globálne zoradenie postupnosti čítania riadkov stránky a pridelením jednej z množiny zadefinovaných tried každému riadku. Predmetom skúmania je taktiež vytvorenie vhodnej reprezentácie grafu, ktorý bude zachytávať vzťahy medzi jednotlivými komponentami na stránke a zvolením vhodnej grafovej neurónovej siete s príslušnými parametrami. Na záver sú jednotlivé prístupy vyhodnotené a porovnané na viacerých metrikách vhodných pre danú problematiku a zistenia zosumarizované s diskusiou o možných vylepšeniach a limitáciach.

Abstract

The aim of this work is to experimentally test the power of graph neural networks in the comprehensive analysis of document layout. In terms of document types, the focus is primarily on newspaper articles and historical writings, such as handwritten books or medieval manuscripts. These are characterized by the complexity of their layout, lacking a fixed structure or having highly segmented text. The work deals with the creation of suitable datasets for training and testing an approach for globally ordering the sequence of reading lines on a page and assigning each line to one of the defined classes. The research also involves creating an appropriate representation of a graph that captures relationships between individual components on the page and selecting a suitable graph neural network with the appropriate parameters. Finally, the different approaches are evaluated and compared on multiple metrics suitable for the given problem, and the findings are summarized with a discussion on possible enhancements and limitations.

Kľúčové slová

GNN, GCN, GraphSAGE, KNN graf, Page XML, postupnosť čítania, analýza rozloženia, klasifikácia hrán, klasifikácia uzlov

Keywords

GNN, GCN, GAT, GraphSAGE, KNN graph, Page XML, reading order, layout analysis, edge classification, node classification

Citácia

OTČENÁŠ, Matej. *Grafové neurónové siete pre analýzu rozloženia stránok*. Brno, 2024. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Michal Hradiš, Ph.D.

Grafové neurónové siete pre analýzu rozloženia stránok

Prehlásenie

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne pod vedením Ing. Michala Hradiša Ph.D. V zozname som uviedol všetky literárne pramene, publikácie a ďalšie zdroje, ktoré boli použité počas prípravy tejto práce.

.....
Matej Otčenáš
16. mája 2024

Poďakovanie

Chcel by som úprimne poďakovať Ing. Michalovi Hradišovi Ph.D, ktorý mi dal príležitosť pracovať na tejto práci, za jeho nápady a odborné rady, ktoré viedli k jej úspešnému dokončeniu.

Obsah

Kapitola 1

Úvod

V súčasnej dobe digitalizácie informácií vzniká veľké množstvo dát, s ktorými človek prichádza denne do kontaktu. Snaha predkladať informácie ľuďom v digitálnej forme je čoraz väčšia a s ňou prichádza aj množstvo úloh, ktoré je možné riešiť, pričom motiváciou ostáva zjednodušenie práce ľudom a obohacovanie o informácie, ktoré môžu byť častokrát ľudskému oku skryté. V prípade, že sa zameriame na užšiu sféru digitalizácie informácií, tak je možné sa špecializovať na získavanie informácií z obrazu. Táto oblasť sa zameriava na zisk dôležitých a zaujímavých informácií, ktoré je možné získať rôznymi spôsobmi. Pod pojmom digitálny obraz si môžme predstaviť rôzne typy dokumentov, ako napríklad fotografie, noviny, historické knihy, vedecké články, publikácie a podobne. Každý z týchto dokumentov obsahuje rôzne typy informácií, akými môžu byť napríklad v prípade novín určenie autora článku, detekcia nadpisu alebo určenie samotného poradia čítania jednotlivých odstavcov či riadkov článku. Takéto dokumenty majú viditeľne rôznu štruktúru. Preto je veľmi dôležité pre získanie takýchto informácií pochopiť samotné rozloženie dokumentu a správne identifikovať jednotlivé komponenty. Vo všeobecnosti je možné kategorizovať analýzu dokumentu do dvoch častí, a to fyzická a logická analýza. Tieto dva kroky sú nevyhnutné na hlbšie pochopenie rozloženia dokumentu a ich detailnejší rozbor je popísaný v kapitole 2, kde sa bližšie popisuje, aké kroky a postupy je nutné spraviť na to, aby bolo možné obrázok stránky analyzovať, a aké informácie je možné získať. Kapitola taktiež popisuje aktuálne práce a metódy na analýzu rozloženia stránok pomocou grafových neurónových sietí. Súčasťou kapitoly je definícia riešeného problému, analýza stránky z pohľadu grafov a predstavenie vyhodnocovacích metrík, ktoré práca ďalej používa. Kapitola 3 sa venuje popisu grafových neurónových sietí a rozoberá rôzne varianty grafových neurónových sietí. Kapitola 4 sa venuje prehľadu dostupných dátových sád, ktoré boli vytvorené za účelom skúmania analýzy rozloženia stránok, a taktiež je venovaná pozornosť prístupu na vytvorenie vlastnej dátovej sady Lidové noviny, ako aj modifikácia populárneho datasetu IMPACT, za účelom trénovania a validácie modelu. Kapitola 5 popisuje navrhnutý prístup k riešeniu zvolených úloh s využitím GNN a samotnú implementáciu. Kapitola sa detailne venuje spôsobu extrakcie príznakov z dát, tvorbe grafu a zakódovaniu príznakov do štruktúry grafu. Ďalej popisuje vytvorenú architektúru modelu, a taktiež spôsob, akým bolo z výstupu modelu získané konečné poradie čítania riadkov. Kapitola 6 sumarizuje rôzne typy experimentov prevedených na zvolených dátových sadách a vyhodnotenie spomedzi nich pomocou zvolených metrík, uvádza detaily nastavenia parametrov modelu a porovnáva dosiahnuté výsledky. Následne sa venuje pozornosť limitáciam navrhnutého prístupu a zhrnutiu poznatkov a objavov v tejto páci. Súčasťou tejto kapitoly je diskusia o mož-

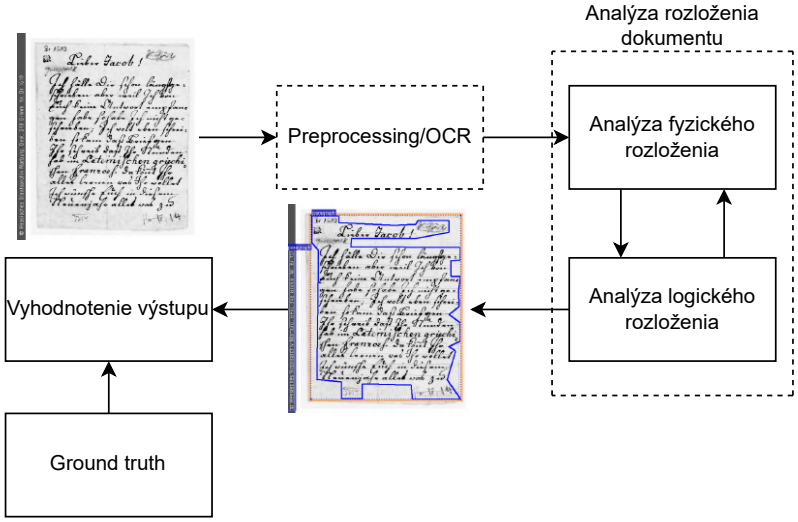
nostiach reprodukovateľnosti samotného riešenia, problémy, ktoré táto práca nevyriešila a návrhy na prípadné vylepšenia. Na záver kapitola 7 sumarizuje celkovú prácu.

Kapitola 2

Spracovanie a analýza rozloženia stránok

Na to, aby bolo možné akúkoľvek analýzu rozloženia dokumentov (*angl. Document layout analysis*) vykonávať, je na začiatok potrebné získať samotný dokument v digitálnej podobe. Dokumenty môžu byť vytvorené manuálne alebo strojovo. V oboch prípadoch je však niekoľko faktorov, ktoré ovplyvňujú jeho vzhľad a obsah, ako napríklad použitý pigment, nástroj, ktorý nanáša pigment na papier alebo samotná fyzická štruktúra papieru. Spôsob, akým stroje a ľudia vnímajú vzhľad dokumentu preto závisí od toho, akým spôsobom bol vytvorený či získaný. Obrázky dokumentu sú zvyčajne vytvorené digitalizáciou z fyzických dokumentov pomocou skeneru alebo digitálnej kamery, no v dnešnej dobe je taktiež veľké množstvo dokumentov vytvorených priamo v digitálnej forme, pri ktorých už externé zariadenia nie sú potrebné. Mnohé z týchto dokumentov, ako napríklad noviny, magazíny, brožúry sa skladajú z veľmi komplexného rozloženia, takže umiestnenie nadpisov, odstavcov, obrázkov má veľmi širokú variabilitu. Človek pri čítaní takýchto dokumentov používa rôzne druhy pomôcok ako pochopenie textového kontextu, konvencie pri čítaní obsahu, jazykové znalosti alebo vizuálny kontext. Väčšinu z týchto krokov čitateľ vykonáva takmer automaticky, častokrát bez hlbšieho zamýšľania. Každá z takýchto nápoved je však strojovému spracovaniu skrytá, kvôli čomu sa analýza dokumentu s komplexným rozložením stáva pomerne náročnou úlohou. Napriek tomu je možné nájsť inšpiráciu práve u ľudí a aplikovať ich vnímanie obsahu dokumentu pri strojovom spracovaní, pričom sa prevedú ľudské paralelné vnemy na niekoľko postupných krokov, ktoré spracujú rozloženie stránok. Obecne je strojovú analýzu rozloženia možné rozdeliť do dvoch, prípadne troch častí, ktoré sú zobrazené na obrázku 2.1.

Prvou časťou je optické rozpoznávanie znakov (*angl.* OCR) [47]. Tento krok nie je síce priamou súčasťou procesu analýzy rozloženia, ale vytvára veľmi dôležitý podklad. Ďalšia časť sa už priamo vzťahuje na analýzu rozloženia, ktorá sa nazýva fyzické rozloženie stránky, prípadne segmentácia stránky alebo geometrické rozloženie stránky. Poslednou časťou je analýza logického rozloženia stránky alebo aj porozumenie dokumentu (*angl.* Document Understanding). Cieľom analýzy rozloženia [11, 7] je dekompozícia dokumentu a extrakcia homogénnych komponent, ako napríklad textové bloky (zóny), riadky textov, grafické symboly, tabuľky alebo obrázky, ktoré je možné potom logicky roztriediť na paragrafy, nadpisy, autora, titulky a podobne.

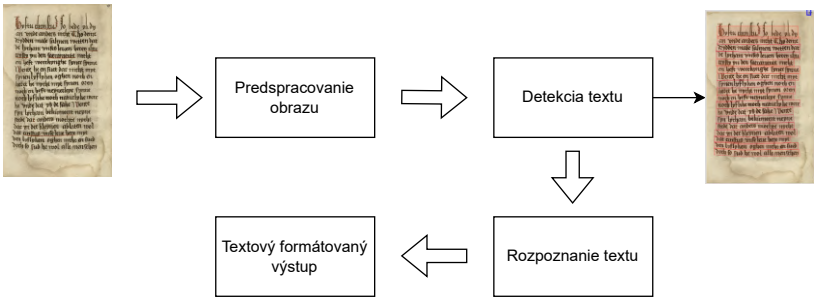


Obr. 2.1: Schematický diagram analýzy rozloženia dokumentu. Prevzaté a upravené zo zdroja [7].

2.1 Optické rozpoznávanie znakov (OCR)

Aby bolo možné vykonať komplexnú analýzu rozloženia dokumentu je nutné vykonať niekoľko krokov predspracovania obsahu obrazového dokumentu. Zásadným z nich je proces optického rozpoznávania znakov. OCR je vytvorený software, ktorý transformuje ručne písaný alebo tlačený text do digitálnej formy, vďaka ktorej môže stroj ďalej manipulovať s takýmto výstupom. Samotné riešenie problému OCR nie je vôbec jednoduché a vyžaduje komplexný prístup [23, 47], keďže kvalita výstupu závisí od mnohých faktorov, na ktoré je potrebné brať ohľad, ako napríklad rôznorodosť jazykov, font a štýl znakov, pravidlá jazyka, či vôbec samotná kvalita vstupného dokumentu.

Proces optického rozpoznávania je možné rozdeliť do niekoľkých krokov [23], ktoré sú zobrazené na obrázku 2.2. V úvodnom kroku je potrebné dostať na vstup dokument v jeho digitálnej podobe. Keďže nie je možné predpokladať, že vstupný obraz obsahuje text, ktorý je dôkladne zarovnaný a obsahuje iba digitálne napísaný text, tak sa musí jeho vstup typicky upraviť do podoby, ktorá bude pre vstup do nástroja OCR čo najmenej komplikovaná. Tento krok sa nazýva predspracovanie obrazu. Predspracovanie obrazu zahŕňa rôzne metódy, akými môžeme vstupný obraz transformovať do čo najviac čitateľnej podoby pre stroj, akými sú rotácie obrazu o určitý uhol, v prípade jeho naklonenia, odstránenie šumu, binarizácia, ktorá oddelí pozadie od textu, dilatácia či erózia textu. Krok detekcie textu je možné nazvať taktiež aj ako segmentácia obsahu stránky. Typicky sa jedná o proces, kde sa hierarchicky rozdelia jednotlivé fyzické segmenty do viacerých podskupín. Na začiatku sa vykoná segmentácia pre celé rozloženie stránky, v ktorej sa detekujú rôzne typy oblastí, obvykle textové oblasti, tabuľky a obrázky. Následne sa v prípade textu môže zvoliť ešte jemnejšia granularita, ktorá rozdelí text do odstavcov, riadkov, slov a nakoniec samotných znakov. Nasledujúci krok je rozpoznanie textu z vysegmentovaných regiónov. Do tohto kroku je možné zahrnúť aj tzv. *postprocessing* rozpoznaného textu, počas ktorého sa vykonáva kon-



Obr. 2.2: Základná pipeline pre optické rozpoznávanie znakov.

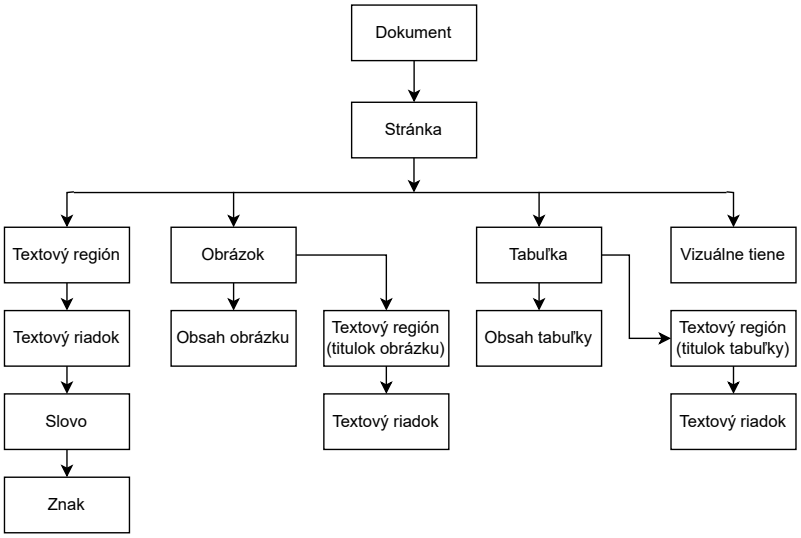
trola pravopisu. Ako posledný, no o nič menej dôležitý krok je uloženie výstupu. Zväčša sa jedná o textový výstup, ktorého informácie sú uložené vo vhodnom štandardizovanom formáte, akým je napríklad Page XML [41], ktorý je detailnejšie popísaný v kapitole 4. Okrem samotného textu detekovaných častí dokumentu môžu byť súčasťou aj mnohé pridané informácie, ako napríklad súradnice polygónu, rozpoznaných odstavcov, riadkov, výška riadkov a podobne [4]. Pomocou takýchto informácií je potom možná prípadná rozsiahlejšia analýza rozloženie dokumentu, ktorá vedie k zisku nových informácií.

2.2 Fyzické rozloženie

Fyzické rozloženie dokumentu je zamerané na fyzické umiestnenia a ohraničenia rôznych regiónov na obrázku dokumentu [37]. Úlohou fyzickej analýzy dokumentu je správne hierarchicky rozdeliť dokument na jednotlivé homogénne komponenty, akými sú napríklad obrázky, pozadie, odstavce, riadky, slová, znaky, a iné. Spôsoby, akými je možné realizovať fyzickú analýzu sa dajú rozdeliť na dve skupiny [7]. Prvou z nich je tzv. *bottom-up* algoritmus, alebo aj algoritmus zdola hore. Ten sa na začiatku zameriava na najmenšie komponenty dokumentu (pixely, spojené komponenty) a opakovane ich spája do väčších homogénnych celkov (regiónov). Naopak, prístup *top-down* alebo zhora dole pracuje, ako už z názvu vyplýva, presne opačným spôsobom. Algoritmus zhora dole začína od celého obrazu dokumentu, ktorý postupne rozdeľuje na menšie regióny, tak ako je zobrazené na obrázku 2.3. Okrem toho je možné tieto dva prístupy spojiť ich vhodnou kombináciou do jedného.

2.3 Logické rozloženie

Logická analýza rozloženia nadväzuje na svoju fyzickú analýzu [37]. V tejto časti je možné fyzickým častiam stránky priradiť ďalšie informácie, zamerané na obsah fyzických komponent. Tieto informácie môžu byť napríklad to, či sa jedná o nadpis, podnadpis, autora, titulok, paragraf, číslo stránky, poznámka pod čiarou a mnohé ďalšie. Kategorizácia dokumentu je dôležitou súčasťou pre vykonanie logickej analýzy rozloženia, keďže logické komponenty, ktoré chceme extrahovať z dokumentu sú závislé predovšetkým od samotnej kategórie dokumentu [11]. Napríklad v prípade vedeckej publikácie z obrázku 2.4c nás môže



Obr. 2.3: Hierarchická štruktúra komponent dokumentu. Prevzaté a upravené podľa práce [11].

(a) (b) (c)

Obr. 2.4: Príklady rozložení stránok. (a) Komplexná moderná novinová stránka obsahujúca množstvo odstavcov, riadkov a viacero článkov. Obrázok z datasetu Lidové noviny. (b) Príklad historickej stránky. Obrázok z datasetu cBAD [13]. (c) Stránka vedeckého textu s komplexnejším rozložením a rôznymi logickými komponentami. Obrázok z datasetu DocBank [35].

zaujímať autor, nadpis alebo abstrakt. V prípade historických kníh a spisov ako je zobrazené napríklad na obrázku 2.4b sa môže jednať taktiež o autora, číslo stránky, názov kapitoly. Pri čítaní novinových článkov, obrázok 2.4a, môže byť zaujímavou informáciou samotné poradie čítanie riadkov či odstavcov, pre lepšie pochopenie kontextu. Z tohto dôvodu je preto veľmi dôležité na vykonanie správnej logickej analýzy pochopiť samotný charakter kolekcie dokumentov, s ktorou sa pracuje, aby bolo možné získať rozumné informácie.

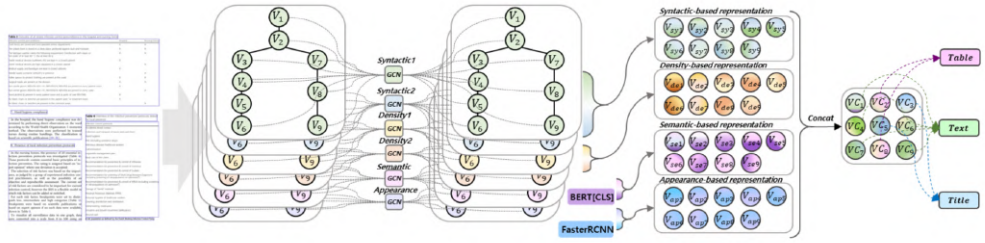
2.4 Prehľad súčasných metód na analýzu stránok

V tejto sekcii je venovaná pozornosť niektorým zaujímavým prístupom, ktoré sú zamerané na analýzu rozloženia dokumentov s použitím GNN, alebo prácam, ktoré používajú iné techniky, ale ich cieľ je podobný ako v tejto práci, a to rekonštrukcia poradia čítania a určovanie tried jednotlivých textových komponent na stránke.

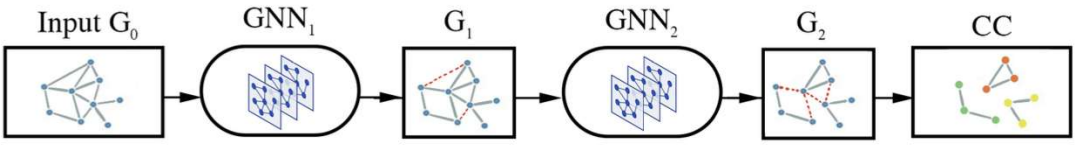
Cieľom práce autorov *Wang et al.* [50] je vytvoriť globálne poradie čítania riadkov na stránke. Na tento účel autori vytvorili metódu založenú na modifikovanej grafovej konvolučnej neurónovej sieti (GCN) [27] nazývanej MPNN [17] a tvorbu grafu pomocou β-skeleton metódy [28]. Práca využíva výhradne geometrické vlastnosti bounding boxov riadkov ako vstup do uzlových embeddingov a aplikuje nad výstupom z GNN množinu definovaných pravidiel z práce [5], ktoré vo výsledku dopomáhajú zostaviť konečné poradie. Prístup založený na pravidlách je vo všeobecnosti jednoduchý a efektívny, avšak môže zlyhať pri dokumentoch so špecifickými vlastnosťami (*angl. domain specific*), ako napríklad ručne písaný text v historických dokumentoch alebo novinové články, ktoré častokrát obsahujú značný počet stĺpcov textu a množstvo obrázkov, ktoré text prerušujú. Autori navrhujú model nezávislý od jazyka dokumentu, čím dosahujú kompaktnosť a efektivitu. Na zakódovanie informácií do hrán využívajú vizuálne vlastnosti z výrezov obrázku, tzv. *region of interest*. Táto oblasť stránky je braná do úvahy pre dva susediace riadky, ktoré sú prepojené hranou a výsledný bounding box, na ktorom sa počítajú vizuálne príznaky, je ohraničením týchto dvoch riadkov. Na extrakciu vizuálnych príznakov používajú *MobileNetV3-Small* [21]. Rekonštrukcia výsledného poradia čítania prebieha spájaním uzlov grafu na základe predikovaných hrán do paragrafov (clustrov) a následným zoradením týchto clustrov heuristickou metódou založenou na pravidlách z práce [5]. Tento krok označujú ako *cluster-and-topological-sort*.

Autori *Prasad et al.* [42] vytvorili metódu, ktorá spája textové riadky do sémantických objektov (paragraf, článok z novín, bunka tabuľky atď.). Každá stránka je reprezentovaná jedným grafom, kde riadky sú uzly a hrany reprezentujú vzťah medzi dvoma riadkami. Počiatočne vytvorený graf následne prevedú do tzv. *conjugate* grafu, kde uzly predstavujú pôvodné hrany a prepojenie dvoch uzlov v grafe nastane práve vtedy, keď v pôvodnom grafe tieto dve hrany zdieľali jeden spoločný uzol. Takýmto spôsobom sa zmenila štruktúra grafu, a taktiež sa prehodili príznaky hrán a uzlov. Vo fáze trénovania je použitý conjugate graf na natrénovanie klasifikátora, ktorý určuje každý uzol do dvoch tried. Trieda uzlu (predtým hrany) je klasifikovaná do triedy 1, ak spája dva textové riadky, ktoré patria do rovnakej subdivízie v pôvodnom grafe, inak je trieda 0. V práci bola použitá hranová konvolučná sieť ECN [10] a štatistická modelovacia metóda Conditional Random Field (gCRF)[10], kde ECN je modifikovaná GCN.

Luo et al. [36] používajú grafové konvolučné neurónové siete pri heterogénnych grafoch. Cieľom tejto práce je klasifikácia segmentov stránky do jednotlivých tried (tabuľka, text, nadpis), kde sa jedná o klasifikáciu uzlov. V uzloch sa zakódujú pozičné príznaky a jednotlivé vzťahy medzi komponentami stránky. Základom je konštrukcia 6 grafov, kde každý z grafov kóduje jeden aspekt príznakov komponent (syntaktické, sémantické, hustota textu a vizuálne príznaky). Syntaktické príznaky a príznaky hustoty textu sú zakódované v uzloch dvoch grafov (dohromady teda 4 grafy), v závislosti na zvolenej inicializačnej metóde. Syntaktické príznaky sú extrahované pomocou tzv. *constituency parseru* [30], ktorého princíp spočíva v mechanizme *attention*. Výpočet príznakov hustoty textu je realizovaný pre každú komponentu, ako podiel medzi počtom tokenov na úrovni znakov a veľkosťou oblasti bounding boxu, ktorý daná komponenta na stránke zaberá. Vizuálne príznaky sú extrahované pomocou predtrénovaného modelu Faster-RCNN [45] a sémantické príznaky sú vypočítané



Obr. 2.5: Architektúra klasifikátoru Doc-GCN. Obrázok prevzatý z [36].

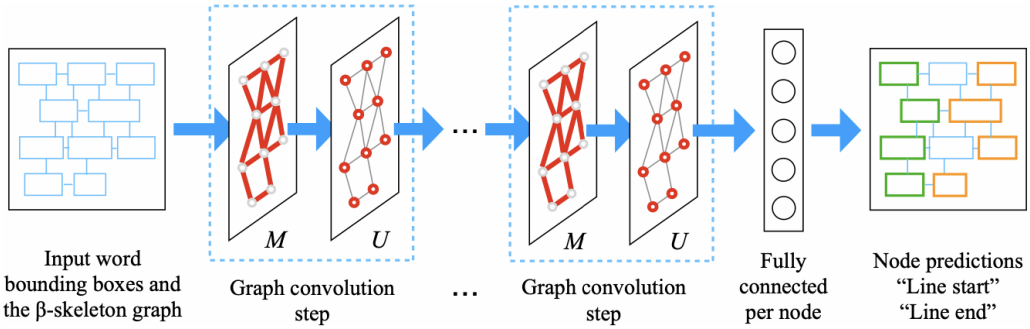


Obr. 2.6: Návrh tzv. *two-pass* modelu založeného na GNN na detekciu riadkov, stĺpcov a buniek v tabuľke. Obrázok bol prevzatý z [43].

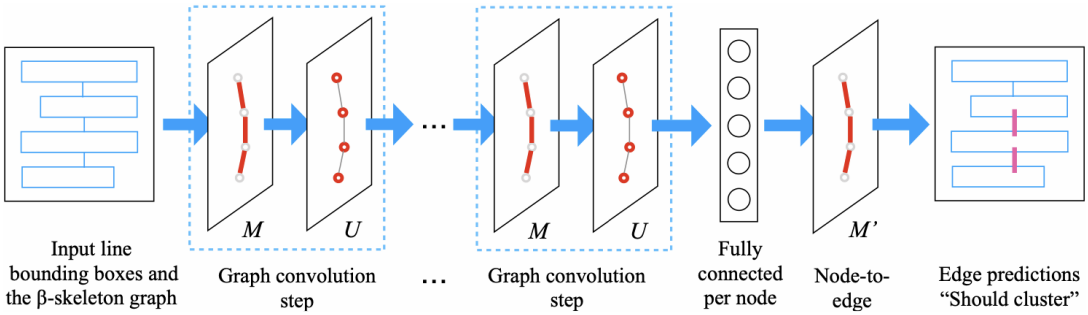
pre každú komponentu pomocou jazykového modelu BERT [12]. Takto vytvorené grafy s vytvorenými príznakmi uloženými v uzloch sú následne vložené na vstup do grafovej neurónovej siete GCN. Na záver sú spočítané vektory uzlov z každého grafu konkatenované do jedného vektoru, ktorý je potom klasifikovaný do jednotlivých tried. Tento navrhnutý postup je ilustrovaný obrázkom 2.5.

Práca autorov *Jose Ramón Prieto a Enrique Vidal* [43] predstavuje metódu na identifikáciu riadkov, stĺpcov a buniek v tabuľkách pomocou grafovej neurónovej siete s názvom *EdgeConv* [52]. Graf je konštruovaný na základe konceptu *line of sight*, kde dva uzly sú prepojené hranou, ak medzi nimi neexistuje žiadny iný bounding box, ktorý by im bránil v priamej viditeľnosti. Takto prepojený graf umožňuje zachytiť štruktúru tabuľky a uľahčuje identifikáciu jej výsledných komponentov. Metóda implementuje tzv. dva prechody cez GNN, ktorý je zobrazený na obrázku 2.6, kde v prvom kroku tzv. GNN_1 vytvorí nový graf z pôvodného vstupu, a na základe predikovaných tried hrany odstráni. Následne je takto modifikovaný graf znovu vstupom do GNN_2, kde podobne ako po výstupe z GNN_1 sa v grafe niektoré hrany odstránia. Vo výsledku je potom na výstupe niekoľko spojených komponent (podgrafov), kde každá z spojených kompontent priamo korešponduje jednému z typov riadok, stĺpec alebo bunka v tabuľke.

Autori *Wang et al.*[49] sa zameriavajú na klasifikáciu sémantického obsahu dokumentov pomocou grafovej konvolučnej neurónovej siete (GCN) s multi-task prístupom, ktorá zahŕňa klasifikáciu uzlov a hrán. Vstupnými dátami sú segmentované riadky textu, pričom cieľom je ich následne zoskupiť do väčších celkov (odstavcov). Uzlová klasifikácia priraďuje riadku jednu z definovaných kategórií (nadpis, poznámka pod čiarou, paragraf, atď.). Hranová klasifikácia je binárna, kde negatívna hrana indikuje, že dva riadky nepatria do rovnakého segmentu, zatiaľ čo pozitívna hrana naznačuje, že sa majú spojiť do jedného. Trieda výsledného segmentu sa potom určuje na základe najpočetnejšej kategórie v danej spojenej komponente. Na trénovanie modelu sa používa ako *loss* funkcia *CrossEntropy* pre obe kla-



Obr. 2.7: Architektúra modelu *line splitting* na identifikáciu riadkov z úrovne slov, pomocou klasifikácie uzlov do tried *line start* a *line end*. Prevzaté z [51].



Obr. 2.8: Architektúra modelu *line clustering*, ktorá spája jednotlivé riadky do celých odstavcov pomocou binárnej klasifikácie hrán v GNN. Prevzaté z [51].

sifikačné úlohy. Celková loss funkcia potom definovaná ako súčet loss z uzlov a váhovanej loss z hrán, pomocou vhodne nastaveného parametru α.

Wang et al. [51] riešia problém spájania slov a riadkov do odstavcov pomocou GCN, ktorá je založená na TF-GNN [14], pričom táto sieť funguje rovnako ako MPNN [17] a GraphSAGE [19]. Ich vstupné features sú čisto pozičné príznaky bounding boxov slov a riadkov, pričom tak robia zámerne, aby ich model bol čo najmenší, a zároveň dosahoval optimálne výsledky. Tvorba grafu je, podobne ako v práci [50], pomocou β-skeleton grafu. Autori riešia problém spájania slov do odstavcov pomocou dvoch krokov, a to kroky *line splitting* a *line clustering*. Krok line splitting (obrázok 2.7), má na vstupe β-skeleton graf s uloženými bounding box informáciami pre každé slovo v texte. Tento vstup následne pošlú do už spomínanej GCN, kde na záver pomocou plne prepojenej siete klasifikujú uzly do dvoch tried. Tieto triedy sú *line start* a *line end*, kde prvá z nich označuje slovo, ktorým riadok začína a trieda line end udáva, kde riadok končí. Pomocou tohto prístupu sú schopní previesť vstupnú granularitu slov na úroveň celých riadkov. V kroku line clustering (obrázok 2.8) je znovu na vstupe β-skeleton graf, ale už zrekonštruovaných riadkov, ktoré sú ďalej na vstupe do GCN. Na záver je po plne prepojenej vrstve transformovaná reprezentácia uzlov na reprezentáciu hrán, keďže je ich cieľom binárna klasifikácia hrán, kde pozitívne predikcie značia, že dva riadky k sebe patria. Takýmto spôsobom autori vytvoria z pôvodného vstupu na úrovni slov až celé odstavce, pričom sa ale ďalej nezaoberajú sémantickým významom jednotlivých odstavcov.

2.5 Typy úloh prevediteľných na graf

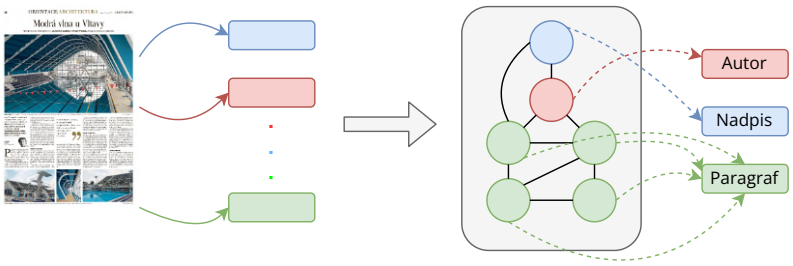
V tejto sekcií sa detailnejšie popisujú a analyzujú niektoré z možných typov úloh, ktoré je z pohľadu analýzy rozloženia dokumentu možné reprezentovať pomocou grafu, čo umožňuje aj následnú aplikáciu grafových neurónových sietí nad nimi. Popis všetkých úloh vychádza z mojej analýzy, počas ktorej som skúmal, ako by sa vhodne mohla reprezentovať stránka pomocou grafu, a následne aké typy úloh z pohľadu analýzy rozloženia by nad nimi bolo možné skúmať. Následne sa táto analýza opiera aj o už dostupné práce, z ktorých taktiež čerpá. Úlohy klasifikácie komponent stránky a rekonštrukcia reading order sú z pohľadu analýzy rozloženia pomerne časté a existuje niekoľko prác, ktoré sa touto problematikou zaoberajú, ako už bolo popísané v časti 2.4. Ďalšie dva typy úloh, menovite spojovanie klasifikovaných riadkov a spojovanie rozdelených riadkov, sú úlohy, ktoré som sám analyzoval a skúmal. Je dôležité spomenúť, že nie všetky popísané úlohy sú v tejto práci aj implementované, ale bude sa jednať iba o ich podmnožinu, menovite rekonštrukciu poradia čítania a klasifikáciu komponent dokumentu. Pre úplnosť je však vhodné spomenúť všetky úlohy, ktorými sa práca vo svojom skorom začiatku zaoberala.

Klasifikácia komponent dokumentu

Klasifikácia homogénnych komponent obsahu stránky zameriava na správne priradenie logických komponent do jednotlivých tried, ako napríklad nadpis, paragraf, poznámka pod čiarou, marginália, autor článku a mnohé iné [54, 36]. Samotný problém je možné dobre reprezentovať pomocou grafu. Obrázok 2.9 vykresľuje, ako by prebiehala klasifikácia takýchto komponent. Vstupný obraz je možné abstrahovať na niekoľko individuálnych fyzických komponent, kde každá z týchto komponent zastupuje svoju logickú pozíciu v dokumente. Vďaka tomu sa dá celá stránka reprezentovať pomocou grafovej štruktúry, kde uzly toho grafu predstavujú individuálne fyzické časti, ako napríklad riadky či odstavce. Každá z týchto častí má určité vlastnosti, kde jednou zo základných sú geometrické črty, ako napríklad pozície súradníc (x, y) v texte, výška riadkov a pod. Z vizuálneho hľadiska môže ísť napríklad o rôzne druhy fontov textu, šírku znakov alebo farbu textu. Podobne je možné využiť aj iné druhy príznakov, ako napríklad z pohľadu významu (sémantiky) textu alebo jeho syntaxe (nadpisy môžu obsahovať väčší počet podstatných a prídavných mien a menej slovies). Ďalšou pozitívnou vlastnosťou grafovej reprezentácie je využitie hrán. Hrany môžu, podobne ako uzly, niesť dôležité informácie, ktoré dopomáhajú k lepšiemu pochopenie rozloženia. Jednou z takýchto informácií môže byť geometrická vzdialenosť medzi dvomi uzlami. V prípade, že by existovala hrana medzi uzlom, ktorý reprezentuje nadpis a uzlom, ktorý reprezentuje riadok v odstavci, je pravdepodobné že ich vzdialenosť bude väčšia, ako vzdialenosť medzi dvomi uzlami riadkov, ktoré významovo patria do jedného odstavca, keďže paragrafy majú tendenciu hustejšieho zoskupenia. Podobne by však platilo, že vzdialenosť medzi dvomi riadkami dvoch odlišných odstavcov by taktiež mali medzi sebou väčšiu vzdialenosť. Z pohľadu grafov teda pôjde o klasifikáciu uzlov do svojich individuálnych tried.

Spojovanie rozdelených riadkov

Úloha spojovania riadkov vychádza z mojej analýzy. Spojovanie rozdelených riadkov nie je až tak bežné, no môže slúžiť ako veľmi dobrá pomoc na vytvorenie správnej *ground truth* v prípade chybných výstupov z nástroja OCR. Chybné výstupy z OCR nástrojov sú vo všeobecnosti ich bežnou súčasťou a užívateľ túto skutočnosť musí brať vždy na vedo-

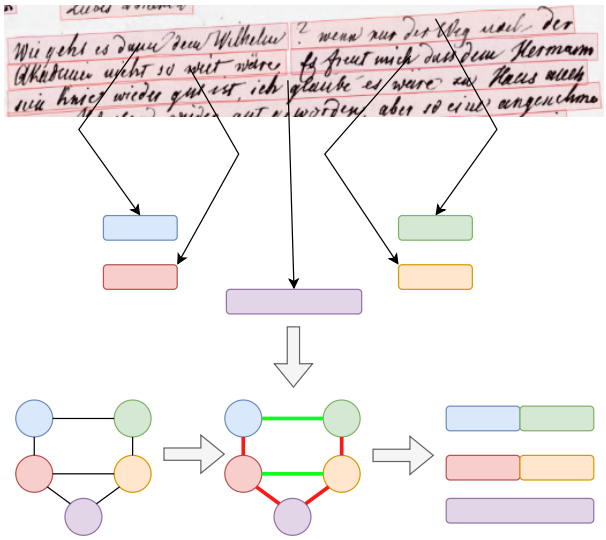


Obr. 2.9: Základný postup klasifikácie logického obsahu prevedený do reprezentácie grafu.

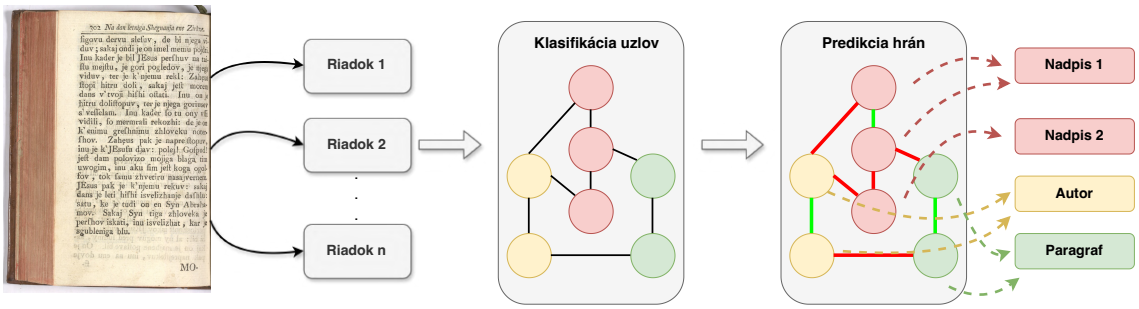
mie. Na obrázku 2.10 je možné vidieť, že na vstupe je časť detekovaných riadkov z mnou použitého nástroja Pero OCR [31, 29, 32], kde vstupom bol pomerne náročný ručne písaný text, ktorý obecne nemá danú pevnú štruktúru textu, ako to býva pri digitálnom texte. Problém však nastáva v tom, že niektoré riadky nástroj rozdelí na dva separátne, čo sa považuje za chybnú identifikáciu riadku. Na obrázku 2.10 je možné vidieť, že prvý riadok obsahuje vo svojom strede znak otázniku, čo môže viesť k chybnej identifikácii. Úlohou teda ostáva spojiť pomocou vhodného prístupu takéto riadky do jedného, a tým vytvoriť korektný výstup. Znovu je možné zaviesť prístup pomocou grafovej reprezentácie. V tomto prípade nepôjde o klasifikáciu samotných uzlov, ako pri úlohe zo sekcie 2.5. Úlohou bude predikcia hrán (*angl.* link prediciton), kde bude cieľom prepojiť dva uzly (riadky) pomocou jednej hrany. Na obrázku je znovu ilustrovaná transformácia riadkov na jednotlivé uzly, kde každý z uzlov má v sebe zakódované vhodné charakteristické príznaky. Podobne ako v úlohe zo sekcie 2.5 je možné využiť niektoré vhodné geometrické príznaky, ktoré budú ďalej slúžiť ako vstup do neurónovej siete. Výstupom pre danú úlohu budú existenčné hrany a finálne spojenie riadkov do jedného. Ako je možné vidieť na obrázku 2.10, hrany označené zelenou farbou symbolizujú existenciu spojenia medzi dvomi uzlami, naopak červené hrany znamenajú, že medzi danými uzlami prepojenie nie je.

Spojovanie klasifikovaných riadkov

Podobne ako pri spojovaní rozdelených riadkov, aj táto úloha vychádza z mojej analýzy a skúmania. Podstatou spájania riadkov, ktoré boli kategorizované do určitých tried je zistenie, či dané riadky predstavujú v dokumente tú istú komponentu. Keďže pri analýze rozloženia na úrovni riadkov je klasifikovaný každý riadok, tak je taktiež potrebné určiť, či sa významovo nejedná o rovnakú komponentu v dokumente. Obrázok 2.11 zobrazuje, ako by daný problém mohol vzniknúť, a akým spôsobom by mal byť vyriešený. Dokument zväčša obsahuje veľké množstvo riadkov, z ktorých každý má svoj význam, no je možné pri ich klasifikácii uvažovať o tom, že zastupujú rovnaký význam v texte. V prípade, že je text komplexný a členitý, tak sa bude pravdepodobne skladať z viacerých odstavcov. Problémom ale je určiť, ktoré riadky textu tvoria jeden ucelený odstavec, a preto je dôležité nielen identifikovať, že daný riadok je súčasťou paragrafu, ale aj určiť, ku ktorým ostatným riadkom paragrafu sa vzťahuje. Podobne tomu môže byť aj pri nadpisoch, ktoré sa môžu geometricky rozprestierať na viacero riadkov, ale významovo sa jedná iba o jeden nadpis. Preto je tento problém prevedený na klasifikáciu uzlov (riadkov) do svojich tried a následne



Obr. 2.10: Princíp spojovania riadkov pomocou predikcie hrán v grafe.

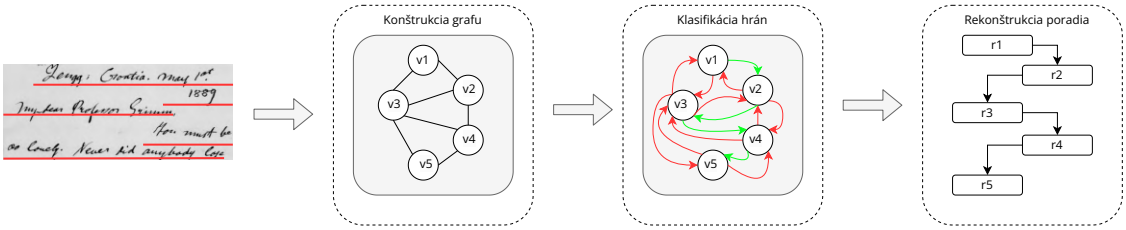


Obr. 2.11: Princíp spojovania klasifikovaných riadkov.

do predikcie hrán, ktoré budú rozhodovať o tom, či teda dané riadky formujú jeden logický celok.

Postupnosť čítania

Postupnosť čítania (*angl. reading order*) určuje logické poradie textových regiónov v dokumente [8, 44]. Cieľom určenia postupnosti čítania je dodať dokumentu celkovú informáciu o tom, v akom poradí sú jeho jednotlivé komponenty usporiadané. Textové regióny typicky obsahujú bloky textu, ktoré sú usporiadané do textových riadkov. Tieto riadky, v závislosti na jazyku, môžu byť uložené horizontálne alebo vertikálne. V bloku textu sa jednotlivé riadky čítajú zvyčajne zhora dolu (horizontálne riadky) a zľava doprava (vertikálne riadky) [11]. Počas analyzovania problematiky reading order som zistil, že reading order nemusí byť vo všeobecnosti lineárny ani jedinečný, keďže je možné stránky s komplexným rozložením čítať v ľubovoľnom poradí (napríklad prečítať poznámku pod čiarou predtým, ako sa prečíta nadpis, alebo sa zastaviť v strede čítania paragrafu, prečítať si titulok obrázku a vrátiť sa naspäť). Napriek tomu je ale vhodné považovať reading order za najbežnejší po-



Obr. 2.12: Princíp klasifikácie hrán v grafe na určenie lokálneho poradia čítania.

stup, akým by človek text čítal aby získal čo najucelenejší prehľad o obsahu. Preto budem uvažovať o postupnosti čítania ako o striktne jedinečnom a lineárnom.

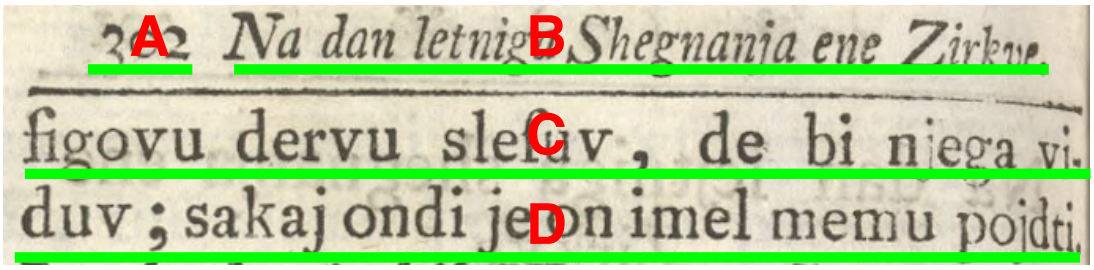
Naivným prístupom pri detekcií postupnosti čítania je možné postupovať napríklad tak, že ak uvážime jeden textový riadok s, ktorý sa nachádza pred iným textovým riadkom $\bar{s}$, tak to platí iba vtedy, ak vertikálna alebo horizontálna súradnica stredu riadku s je menšia ako súradnice riadku $\bar{s}$. Z toho je potom odvodený vzťah $y_s < y_{\bar{s}} \vee x_s < x_{\bar{s}}$. Tento postup sa nazýva top-to-bottom left-to-rigth (TBLR). TBLR je jedným z najtradičnejších postupov, ako automatizovane vytvoriť postupnosť čítania v dokumentoch, a častokrát je to osvedčená metóda pri menej komplexných typoch dokumentov (napríklad kniha, ktorej riadky vždy postupujú lineárne za sebou) [44]. Avšak v prípade, že by tento prístup bol zvolený pri náročnejších rozloženiach dokumentu, ako napríklad noviny, kde už nie je možné sa spoliehať na jednoduché geometrické pozície komponent, tak by tento prístup jednoducho zlyhal. Z tohto dôvodu je možnosťou aplikovať sofistikovanejšiu metódu, ktorá by bola schopná sa vysporiadať aj s takýmito typmi dokumentov.

Keďže sa táto práca venuje použitiu grafových neurónových sietí, je možné znovu uvažovať o reprezentácii tohto problému pomocou grafu, tak ako ilustruje obrázok 2.12. Jednotlivé komponenty (v tomto prípade riadky) sa prevedú na uzly grafu a aplikáciou vhodnej metódy tvorby grafu sa tieto uzly prepoja hranami. Potom už len ostáva úlohou neurónovej siete tieto hrany klasifikovať do správnych tried a nakoniec zostaviť konečné poradie. Tieto triedy si je možné jednoducho predstaviť ako binárnu klasifikáciu, kde zelená hrana symbolizuje, že uzol v_1 má následníka uzol v_2, ale zároveň platí, že uzol v_2 nemá ako svojho následníka uzol v_1. Je teda potrebné uvažovať hrany orientované, kde medzi dvomi uzlami určite neplatí symetria. Práve zostavenie konečného (globálneho) poradia nie je v tejto úlohe triviálnym problémom. Keďže binárnou klasifikáciou hrán získame len čiastočné zoradenie, v zmysle, že vieme, ktoré dva riadky (prípadne odstavce) nasledujú za sebou, nedá sa uvažovať o celkovom poradí, ktoré by určovalo presné poradie čítania riadku na stránke. Spôsobom, akými je možné zostaviť celkové poradie čítania sa viacej venuje sekcia implementácie 5.2.

2.6 Definícia problému riešeného v práci

Cieľom tejto práce je využiť grafovú neurónovú sieť, pomocou ktorej sa natrénuje klasifikačný model, ktorého výstup v kombinácií s heuristickou metódou bude schopný identifikovať postupnosť čítania textu a priradí tomuto textu jeho sémantický význam. Matematicky je možné daný problém definovať pomocou množiny n-tíc, ktorá bude konečným výstupom.

Majme množinu blokov textu vygenerovaných pomocou OCR detektoru z pôvodného digitalizovaného obrázku [34, 44]. Chceme vytvoriť správnu n-ticu týchto blokov, pomocou ktorej následne porozumieme správnej postupnosti čítania textu, a súčasne bude táto n-tica



Obr. 2.13: Príklad obrázku s textom, na ktorom sa vykoná analýza z pohľadu rekonštrukcie poradia čítania a sémantickej klasifikácie textu.

obsahovať správny sémantický význam daného textového bloku. Preto budeme o jednej n-tici uvažovať ako o trojici. Postupnosť čítania musí byť lineárna a jedinečná, čo znamená, že nesmie obsahovať jeden prvok viac ako raz vo svojej postupnosti. Formálne môžeme teda definovať množinu OCR blokov textu ako $\mathscr{T} = \{t_1, t_2, \cdots, t_n\}$, kde t_i odkazuje na i^{ty} blok textu. K tomu definujeme aj množinu $\mathscr{D} = \{d_1, d_2, \cdots, d_m\}$, kde d_j odkazuje na j^{ty} sémantický typ daného textového bloku. Potom definujeme cieľovú množinu n-tíc $\mathbf{z}$ ako n trojíc $\{(t, d, v) : t \in \mathscr{T}, d \in \mathscr{D}, v \in \mathbb{N}^{\leq n}\}$ kde v, označujúce poradie, spĺňa $v_i \neq v_j, 1 \leq i, j \leq n \forall j \neq i$. Napríklad pre obrázok 2.13, výstupná množina trojíc môže byť definovaná ako $\hat{\mathbf{z}} = \{(A, \text{číslo strany}, 1), (B, \text{nadpis}, 2), (C, \text{paragraf}, 3), (D, \text{paragraf}, 4)\}$.

2.7 Prehľad evaluačných metrík a ich aplikácia

Aby bolo možné vyhodnotiť kvalitu vytvoreného modelu na analýzu rozloženia stránok, tak je potrebné zvoliť vhodnú metriku vyhodnotenia, ktorá je relevantná pre daný problém a prípadne bola použitá v iných prácach, s ktorými by sa dala porovnať kvalita vytvoreného modelu. Táto práca sa venuje rekonštrukcii poradia čítania riadkov a ich klasifikácii do príslušných tried. Kým druhý spomenutý typ je pomerne štandardnou úlohou viactriednej klasifikácie, kde budeme uvažovať o takzvanom vyhodnotení nevyvážaného datasetu, prvý typ úlohy má viacero aspektov, ktoré je možné zohľadňovať pri konečnom vyhodnotení. Toto viedlo v konečnom dôsledku k implementácii a aplikácii viacerých typov metrík.

Jedny z najpopulárnejších a široko využívaných metrík v oblasti extrakcie informácií sú **Recall** a **Precision** [11, 18]. Metrika Recall vyjadrená rovnicou 2.2 je definovaná ako podiel počtu správne klasifikovaných pozitívnych elementov (*angl. true positive*) voči súčtu počtu správne klasifikovaných pozitívnych elementov a počtu klasifikovaných negatívnych elementov, ktoré mali byť v skutočnosti pozitívne (*false negative*). Metrika Precision uvedená v rovnici 2.1 je definovaná ako podiel počtu správne klasifikovaných pozitívnych elementov voči súčtu počtu správne klasifikovaných pozitívnych elementov a počtu pozitívne klasifikovaných, ktoré mali byť v skutočnosti negatívne (*false positive*).

Na vyhodnotenie celkovej kvality systému existuje tretia metrika, ktorá berie do úvahy obe z vyššie popísaných metrík. Cieľom je vziať Precision a Recall a vypočítať ich nevážený harmonický priemer, ktorý je definovaný v rovnici 2.3. Táto metrika sa nazýva F_1-skóre.

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \tag{2.1}$$

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{2.2}$$

$$F_1 = \frac{2 \times P \times R}{P + R}. \tag{2.3}$$

Obrovským nedostatkom pri použití týchto metrík v prípade vyhodnotenia postupnosti čítania je to, že pomocou nich viem vyhodnotiť iba klasifikáciu vzťahov medzi dvojicami uzlov, nie však globálne usporiadanie riadkov na stránke. Z tohto dôvodu nie je možné o týchto metrikách ďalej uvažovať ako o relevantných v kontexte vyhodnotenia postupnosti riadkov, ale je dôležité vysvetliť, prečo nie sú postačujúce pri riešení globálneho reading order pomocou grafových neurónových sietí.

Metriky na vyhodnotenie postupnosti čítania

Keďže vyhodnotenie postupnosti čítania je v konečnom dôsledku vyhodnotenie sekvencie postupnosti riadkov, budem sa v práci zameriavať na metriky, ktoré mi dajú čo najlepší prehľad o presnosti zoradenej sekvencie.

Metrika **Spearman footrule distance** [33] je normalizovaná kumulatívna suma vzdialeností medzi dvojicami v zoradených sekvenciach definovaná ako:

$$\rho(\boldsymbol{t}, \boldsymbol{v}) = \frac{\sum_{i=1}^{n} |t_i - v_i|}{\left\lfloor \frac{1}{2} n^2 \right\rfloor}, \tag{2.4}$$

kde t je ground truth zoradená sekvencia, v je modelom predikovaná sekvencia, $\left\lfloor \frac{1}{2} n^2 \right\rfloor$ je maximálna možná kumulatívna vzdialenosť medzi všetkými dvojicami (t_i, v_i), kde n je celkový počet objektov (v mojom prípade riadkov) a $0 \leq \rho(\cdot) \leq 1$ [44].

Normalizovaná Spearman footrule distance dáva prehľad nie len o tom, koľko elementov bolo nesprávne zoradených, ale taktiež poskytuje informáciu o tom, ako veľmi vzdialené sú elementy od svojich správnych pozícií. Ideálnou hodnotou tejto metriky je 0, a teda cieľom je minimalizácia tejto hodnoty.

Ak budeme teda uvažovať túto metriku pri vyhodnotení reading order, tak na jej vstup potrebujeme dve sekvencie. Jedna sekvencia bude reprezentovať ground truth ako postupnosť hodnôt, kde index sekvencie predstavuje riadok a hodnota na danom indexe udáva jeho poradie. Druhá sekvencia bude nami odhadnutá, a musí spĺňať to, že index príslušného riadku je rovnaký, ako v prvej sekvencii. Jednou zo zásadných nevýhod tejto metriky je to, že počíta s totálnym zoradením. Počas experimentálnej fázy som zistil, že v prípade, ak model nepredikuje lokálne prepojenie medzi dvomi riadkami, tak sa stráca aj celková informácia o tomto riadku, a daný riadok tým pádom zo sekvencie vypadne. Aby bolo teda možné túto metriku využiť, tak počas výsledného zoraďovania riadkov sú vynechané riadky nahradené hodnotou celkového počtu riadkov. Tým je zaistené, že metrika porovnáva vždy dve sekvencie o rovnakej dĺžke. Tento spôsob tým pádom penalizuje vynechané riadky maximálnou možnou hodnotou.

Z dôvodu modifikácie metriky Spearman footrule distance, a jej menej zreteľnej interpretovateľnosti budem v tejto práci uvažovať aj o jednoduchšej a priamočiarej metrike, ktorou je **Normalizovaný počet výmen**, ktorú som vymyslel a implementoval, aby boli výsledky lepšie interpretovateľné. Táto metrika je, podobne ako Spearman footrule distance, založená na dvoch vstupných zoradených sekvenciách a počíta normalizovaný počet nesprávne umiestnených elementov v danej sekvencií. Matematicky ju môžeme vyjadriť ako:

$$\rho(\boldsymbol{t}, \boldsymbol{v}) = \frac{1}{n} \sum_{i=1}^{n} 1_{(t_i \neq v_i)}, \tag{2.5}$$

kde $1_{(t_i \neq v_i)}$ predstavuje hodnotu 1 ako sa hodnoty na danom indexe i v oboch sekvenciách nezhodujú a $0 \leq \rho(\cdot) \leq 1$. Problémom pri tejto metrike je však to, že v tejto práci sa uvažuje postupnosť čítania pre riadky, ktorých je typicky na stránke pomerne veľké množstvo. V prípade akejkoľvek zámeny riadku dochádza ku chybe. Podobný problém bol zdôraznený aj v práci [34]. Touto metrikou je možné odhadnúť, ako často je model schopný zostaviť úplné zoradenie postupnosti čítania bez urobenia jedinej chyby. V takom prípade sa jedná o doplnok k danej hodnote.

Poslednou metrikou, ktorá bola mnou vymyslená a implementovaná je **Normalizovaný počet prerušení**. Cieľom je zistiť, ako často je výsledná postupnosť zoradených riadkov prerušená. V tomto prípade ide o hľadanie najdlhších monotónnych a lineárne rastúcich sekvencií, bez ohľadu na ich umiestnenie. Keďže obe predošlé metriky brali do úvahy správnosť modelu iba pri úplnom zoradení, tak táto metrika zanedbáva, kedy správna sekvencia začne. Ak teda uvážime prípad, že by globálne zoradenie riadkov vynechalo prvý riadok zo svojej postupnosti, bude sa jednať o jedno prerušenie, ale zvyšná postupnosť je braná ako správna. Metriku definujeme následovne:

- $S = (s_1, s_2, \ldots, s_n)$ je konečná postupnosť.
- n je počet elementov v S.
- $\theta = n + 1$ je zvolený prah.

Definujeme funkcia prerušení $\delta(i)$ ako:

$$\delta(i) = \begin{cases} 1 & \text{if } s_i = \theta \\ 1 & \text{if } i > 1 \text{ and } s_i \neq s_{i-1} + 1 \\ 0 & \text{inak} \end{cases}. \tag{2.6}$$

Potom počet prerušení D je daný ako:

$$D = \sum_{i=1}^{n} \delta(i). \tag{2.7}$$

Nech $N = \theta - 1$ je celkový možný počet prerušení, potom normalizovaný počet prerušení je definovaný ako:

$$D_{\text{norm}} = \frac{D}{N}. \tag{2.8}$$

Je potrebné dodať, že metrika nepenalizuje samotné pozície sekvencií voči sebe samým. Ak teda začína postupnosť sekvencie poradovým číslom väčším ako i (i-ty riadok v globálnom zoradení), a za koncom tejto sekvencie nasleduje nová sekvencia, ktorá začína poradovým číslom menším ako i, tak sa v tomto prípade nejedná o prerušenie.

Preto sa pri samotnom vyhodnocovaní experimentov pre úlohu postupnosti čítania riadkov budú brať do úvahy všetky tri definované metriky, keďže každá z nich nám dáva inú informáciu o presnosti daného modelu, ale celkovou kombináciou je možné kvalitu daného modelu odhadnúť veľmi dobre.

Metriky pre vyhodnotenie sémantiky textu

Pri klasifikácií riadkov, podobne ako pri určení poradia čítania, bude pre samotné vyhodnotenie použitých niekoľko metrík, pričom je ale ich voľba viac štandardizovaná ako pre predošlú úlohu. Prvou metrikou je **Váhovaný priemer** (*Weighted average*) pre Precision, Recall a F-1 skóre [24, 2]. Výpočet Precision, Recall a F1 je z definície rovnaká ako v rovnici , ale pri viactriednej klasifikácií sa počíta suma cez všetky triedy, a súčasne sa každá metrika samostatne prenásobí tzv. hodnotou *support*, kde táto hodnota predstavuje počet zastúpení jednotlivých vzorkov pre danú triedu. Cieľom váhovaného priemeru je priradiť určitú dôležitosť danej triede na základe jej počtu zastúpení v datasete. Je to spôsob, ako si poradiť s vyhodnotením pri nevyváženom dataste. Ďalej je teda možné definovať výpočet váhovaného priemeru pre Precision, Recall a F1 ako:

$$\text{Precision} = \frac{1}{N}\sum_{k=1}^{K}\text{Precision}_k \times \text{support}_k, \tag{2.9}$$

$$\text{Recall} = \frac{1}{N}\sum_{k=1}^{K}\text{Recall}_k \times \text{support}_k, \tag{2.10}$$

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{2.11}$$

kde $N = \sum_{k=1}^{K} \text{support}_k$.

Poslednou metrikou pre vyhodnotenie klasifikácie riadkov je **Accuracy pre každú triedu**. Keďže vo všeobecnosti metrika Accuracy nie je vhodná na použitie pri viactriednej klasifikácií [18], najmä ak existuje väčšie zastúpenie jednej triedy voči ostatným, tak sa v tejto práci vyhodnocuje každá trieda pomocou Accuracy samostatne. Takýmto spôsobom vieme identifikovať, ktoré triedy je model schopný lepšie, resp. horšie, rozpoznať. Accuracy pre každú triedu definujeme ako:

$$\text{Accuracy}_k = \frac{\text{TP}_k + \text{TN}_k}{\text{TP}_k + \text{TN}_k + \text{FP}_k + \text{FN}_k}. \tag{2.12}$$

Kapitola 3

Grafové neurónové siete a ich aplikácia

V súčastnosti aj vďaka neustálemu pokroku vo vývoji hardwaru sa stalo jednou z najrozšírenejších metód v oblasti umelej inteligencie použitie neurónových sietí, ktoré sa preukázali ako veľmi dobrý nástroj na vyriešenie rôznych typov úloh, na ktoré tradičné metódy nepostačujú. Z pohľadu neurónových sietí sa jedná hlavne o hlboké neurónové siete.

Hlboké učenie (*angl. deep learning*) sa stalo jedným z najpopulárnejších prístupov v oblasti výskumu umelej inteligencie. Konvenčné techniky hlbokého učenia, ako rekurentné neurónové siete a konvolučné neurónové siete dosahujú výborné výsledky na euklidovských typoch dát, ako napríklad obrázky, textové sekvencie, signály a pod. Napriek tomu je ale množstvo dát z reálneho sveta, a problémov s nimi spájajúcich sa, ktoré je možné reprezentovať a formulovať lepšou a prirodzenejšou štruktúrou, a to napríklad grafom [57]. Cieľom grafových neurónových sietí je iteratívne obnovovať a získavať reprezentáciu uzlov pomocou agregovania reprezentácií susedných uzlov a samotného cieľového uzlu z predošlej iterácie. V súčasnosti existuje pomerne veľké množstvo rôznych grafových neurónových sietí [61], pričom ich použitie závisí od zvoleného typu riešenej úlohy. Akonáhle sa učiaci proces dostane do bodu, že sú uzlové reprezentácie optimálne naučené, tak jednou zo základných úloh, detailnejšie popísaných v sekcii 3.3, je klasifikácia uzlov do preddefinovaných tried. GNN sú v súčasnej dobe na rozmachu a dosahujú kvalitné výsledky, no aj pri nich platí, že sa stretávajú s určitými problémami najmä vo fáze trénovania. Jedným z najčastejších je tzv. problém prílišného vyhladenia (*angl. oversmoothing*) [57], pri ktorom dochádza k tomu, že majú všetky uzly veľmi podobnú reprezentáciu. To znamená, že ak sa snažíme vytvoriť model, ktorý má dobrú vyjadrujúcu schopnosť a prehľad o grafovej štruktúre (napríklad pridaním väčšieho počtu vrstiev), tak to môže viesť k tomu, že sa tento model pretransformuje a bude pristupovať ku všetkým uzlom rovnako (reprezentácie uzlov budú konvergovať na nerozlišiteľné vektory).

Na popis chovania grafových neurónových sietí je potrebné na úvod formálne definovať graf, kde budeme rozlišovať medzi dvomi typmi grafov, a to graf orientovaný a neorientovaný [59].

Definícia 3.0.1 *Nech orientovaný graf G je definovaný ako dvojica (V, E), kde*

- *V je konečná množina uzlov a*
- *$E \subseteq V^2$ je množina hrán.*

Definícia 3.0.2 *Nech neorientovaný graf G je definovaný ako dvojica (V, E), kde*

- *V je konečná množina uzlov a*
- *$E \subseteq \binom{V}{2}$ je množina hrán.*

Vo všeobecnosti je teda možné popísať prácu grafových neurónových sietí nasledovným spôsobom. Na začiatku sa inicializujú uzlové reprezentácie $H^0 = X$, kde H^0 sú uzlové reprezentácie vo vstupnej vrstve a X sú počiatočné príznaky (*features*) pre uzly. V každej z vrstiev GNN máme dve nasledovné funkcie [57]:

- **AGREGÁCIA**, ktorá sa snaží agregovať informácie pre každý uzol z okolitých uzlov.
- **KOMBINÁCIA** (UPDATE), ktorá sa snaží aktualizovať uzlovú reprezentáciu kombinovaním agregovaných informácií z okolitých susedných uzlov s ich aktuálnou reprezentáciou.

Tieto funkcie je možné z pohľadu práce v GNN zapísať následne ako:

Inicializácia: $H^0 = X$
Pre $k = 1, 2, ..., K$,

$$\begin{aligned} \alpha_v^k &= \mathbf{AGREGÁCIA}^k\{H_u^{k-1} : u \in N(v)\} \\ H_v^k &= \mathbf{UPDATE}^k\{H_v^{k-1}, \alpha_v^k\}, \end{aligned} \tag{3.1}$$

kde $N(v)$ je množina susedov pre v-ty uzol. Uzlové reprezentácie v poslednej vrstve H^k môžme považovať za konečnú reprezentáciu uzlov.

V prípade, že sme dosiahli konečnú reprezentáciu, je možné ďalej tieto uzly napríklad klasifikovať do jednotlivých tried. Triedu uzlu v označenú ako $\hat{y}_v$ je možné predikovať napríklad pomocou funkcie Softmax:

$$\hat{y}_v = Softmax(WH_v^\top), \tag{3.2}$$

kde $W \in \mathbf{R}^{|\mathscr{L}| \times F}$, $|\mathscr{L}|$ je počet výstupov (*angl.* labels). Na záver je možné celý model trénovať pomocou vhodne zvolenej objektívnej funkcie pomocou klasického minimalizovania chybovej funkcie ako:

$$O = \frac{1}{n_l} \sum_{i=1}^{n_l} loss(\hat{y}_i, y_i), \tag{3.3}$$

kde y_i je ground truth label uzlu i, n_l je počet takýchto uzlov a chybová funkcia ($loss$) je nejaká správne zvolená funkcia, ako napríklad *cross-entropy*. Celú sieť je potom možné optimalizovať minimalizovaním objektívnej funkcie O pomocou algoritmu backpropagation.

3.1 Prehľad existujúcich typov GNN

Odhliadnuc od generického algoritmu, akým sa aktualizujú váhy v jednotlivých vrstvách, existuje hneď niekoľko variant GNN, kde každá z nich pristupuje k danému výpočtu iným spôsobom. Pozornosť bude venovaná primárne trom typom GNN, ktoré sú jedny z najpopulárnejších, a taktiež sú súčasťou experimentálnej časti tejto práce.

Grafová konvolučná neurónová sieť (GCN)

Ako prvou sieťou je grafová konvolučná neurónová sieť [27]. Je to najpopulárnejšia a najviac používaná architektúra vďaka svojej jednoduchosti a efektívnosti pri rôznych typoch úloh. Uzlové reprezentácie v každej vrstve sú aktualizované pomocou nasledujúceho pravidla [27, 57]:

$$H^{k+1} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^k W^k), \tag{3.4}$$

kde $\tilde{A} = A + \mathbf{I}$ je matica susednosti daného neorientovaného grafu $\mathscr{G}$, v ktorej sú zahrnuté aj prepojenia uzlov samých so sebou, čo umožňuje rátanie uzlov s vlastnými príznakmi počas fázy aktualizácie uzlových reprezentácií. $\mathbf{I} \in \mathbf{R}^{N\times N}$ je matica identity, $\tilde{D}$ je diagonálna matica, ktorej hodnoty na diagonále sú $D_{ii} = \sum_j \tilde{A}_{ij}$ a σ je aktivačná funkcia, typicky ReLU alebo Tanh. Matica $W^k \in \mathbf{R}^{F\times\bar{F}}$ je transformačná matica, ktorá sa trénuje počas optimalizácie a $F, \bar{F}$ sú dimenzie reprezentácie uzlu v k-tej respektíve (k+1)-tej vrstve. Rovnicu 3.4 môžme ďalej rozpísať na lepšie porozumenie funkciam AGREGÁCIE a UPDATE. Pre uzol i bude teda aktualizácia uzlu definovaná rovnicou:

$$H_i^k = \sigma\Big(\sum_{j\in N(i)} \frac{A_{ij}}{\sqrt{\tilde{D}_{ii}\tilde{D}_{jj}}} H_j^{k-1}W^k + \frac{1}{\tilde{D}_i}H_i^{k-1}W^k\Big) \tag{3.5}$$

V rovnici 3.5 je funkcia agregácie definovaná ako vážený priemer reprezentácií susedných uzlov. Váha susedného uzlu j je podmienená váhou hrany medzi uzlami i a j. Funkcia aktualizácie je v rovnici definovaná ako suma agregovaných správ od okolitých uzlov a reprezentácie uzlu, ktorý je aktualizovaný. K tomu je reprezentácia uzlu ešte normalizovaná jeho vlastným stupňom.

Grafové attention neurónová sieť (GAT)

Ako ďalší typ GNN je tzv. *Graph Attention Network* vytvorená autormi *Veličković et al.* [48], ktorá využíva mechanizmus *attention*. Pri grafových konvolučných sieťach je dôležitosť suseda j pre cieľový uzol i daná váhou ich hrany A_{ij}. V praxi môže byť vstupný graf častokrát veľmi zašumený (*angl. noisy*). Váhy hrán nemusia odzrkadľovať skutočnú interakciu medzi dvoma uzlami. Z toho dôvodu by bolo vhodné vytvoriť prístup, ktorý by sa dôležitosť každého suseda naučil, k čomu slúžia práve GAT. Tie sa snažia túto dôležitosť naučiť pomocou mechanizmu attention. Attention je využívaný v mnohých odvetviach, ako napríklad spracovanie jazyka, počítačovom videní a podobne [57].

Grafová attention sieť definuje spôsob, ako sa prenášajú skryté uzlové reprezentácie vo vrstve $k-1$ (značené ako $H^{k-1} \in \mathbf{R}^{N\times F}$) na nové reprezentácie uzlov $H^k \in \mathbf{R}^{N\times F'}$ [48]. Na zabezpečenie dostatočnej výrazovej sily na transformáciu reprezentácií uzlov nižšej úrovne na reprezentácie uzlov vyššej úrovne, je na každom uzle použitá spoločná lineárna transformácia, označená ako $W \in \mathbf{R}^{F\times F'}$. Následne je na uzloch definovaná tzv. *self-attention*, ktorá meria attention koeficienty pre každú dvojicu uzlov pomocou tzv. zdieľaného *(angl. shared)* attention mechanizmu $a : \mathbf{R}^{F'} \times \mathbf{R}^{F'} \to \mathbf{R}$

$$e_{ij} = a\left(WH_i^{k-1}, WH_j^{k-1}\right), \tag{3.6}$$

kde e_{ij} označuje silu vzťahu medzi uzlami i a j. Pre ucelenejšie pochopenie uvažujeme H_i^{k-1} ako o stĺpcovom vektore. Aby boli attention koeficienty porovnateľné medzi rôznymi uzlami, tak sú zvyčajne normalizované funkciou softmax:

$$\alpha_{ij} = \text{Softmax}_j\left(\{e_{ij}\}\right) = \frac{\exp\left(e_{ij}\right)}{\sum_{l \in N(i)} \exp\left(e_{il}\right)}. \tag{3.7}$$

Podľa práce [48] je teda možné následne zadefinovať výpočet attention koeficientov ako:

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(W_2\left[WH_i^{k-1} \| WH_j^{k-1}\right]\right)\right)}{\sum_{l \in N(i)} \exp\left(\text{LeakyReLU}\left(W_2\left[WH_i^{k-1} \| WH_l^{k-1}\right]\right)\right)}, \tag{3.8}$$

kde $\|$ reprezentuje operáciu konkatenácie dvoch vektorov. Nová reprezentácia uzla je lineárna kombinácia reprezentácií susedných uzlov, s váhami určenými koeficientami pozornosti (s potenciálnou nelineárnou transformáciou):

$$H_i^k = \sigma\left(\sum_{j \in N(i)} \alpha_{ij} W H_j^{k-1}\right). \tag{3.9}$$

GraphSAGE neurónová sieť

Na GraphSAGE, *Hamilton et al.* [19], je možné pozerať ako na rozšírenie pôvodnej grafovej konvolučnej siete (GCN). Prvou zmenou je generalizácia agregačnej funkcie. Nech $\mathscr{G}(\mathscr{V}, \mathscr{E}), \mathscr{N}(v)$ je okolie $v, \mathbf{h}$ je reprezentácia uzla. Výpočet *embedding* vektoru na vrstve $(l+1)$ od cieľového uzla $v \in \mathscr{V}$ je definovaná ako [19, 57]:

$$\mathbf{h}_{\mathscr{N}(v)}^{(l+1)} = \text{AGGREGATE}_l\left(\left\{\mathbf{h}_u^{(l)}, \forall u \in \mathscr{N}(v)\right\}\right), \tag{3.10}$$

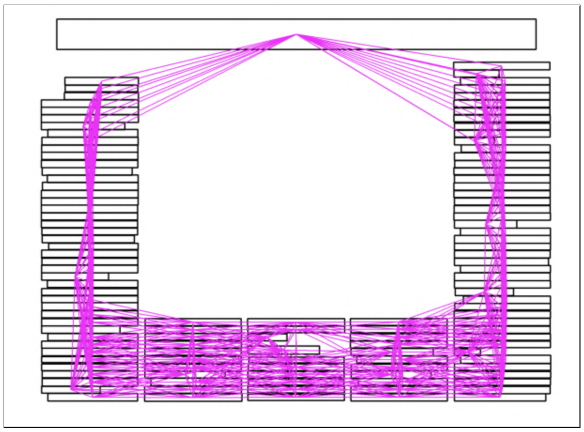
V porovnaní s GCN agregačnou funkciou je GraphSAGE založená na LSTM (*Long short-term memory*) [20] a pooling agregácií na zagregovanie informácií od svojho okolia. Ďalšou zmenou oproti GCN je, že sa aplikuje funkcia konkatenácie na skombinovanie informácií z cieľového uzlu a jeho okolia, kdežto pri GCN sa využívala funkcia sumy. Z popisu teda vyplýva rovnica:

$$\mathbf{h}_v^{(l+1)} = \sigma\left(W^{(l+1)} \cdot \text{CONCAT}\left(\mathbf{h}_v^{(l)}, \mathbf{h}_{\mathscr{N}(v)}^{(l+1)}\right)\right), \tag{3.11}$$

kde $W^{(l+1)}$ sú matice váh a σ je aktivačná funkcia. Na prispôsobenie sa k veľkým grafom, GraphSAGE implementuje techniku trénovania pomocou tzv. *mini-batchov* na redukciu výpočtov počas trénovacej fázy. Viac špecificky, v každej trénovacej iterácii sú brané do úvahy iba uzly, ktoré sa používajú pri výpočte uzlových reprezentácií v *batchi*, čo výrazne znižuje počet vzorkovaných uzlov.

3.2 Tvorba grafu pomocou metódy k-NN

Jeden z najdôležitejších krokov pre to, aby bolo možné využiť grafové neurónové siete korektne, je správna reprezentácia zvoleného problému. Na základe vstupných dát nie je možné vytvoriť zakaždým uniformný typ grafu, ktorý by bolo možné aplikovať na každý typ problému. Preto je veľmi dôležité pochopiť sémantický význam spracovávaných informácií. Uzly



Obr. 3.1: Graf vytvorený pomocou metódy k-NN pre ($k = 20$) nad článkom z datasetu Lidové noviny.

a hrany grafov reprezentujú rôzne druhy vzťahov, ktoré medzi dátami vznikajú. Niektoré dáta majú svoju štruktúru pomerne zjavnú a častokrát hovoria implicitne samé za seba, ako napríklad štruktúry molekúl či proteínov, ktoré sú zo svojej podstaty prirodzenou reprezentáciou grafu [57, 58]. Napriek tomu nie všetky druhy dát majú takúto zjavnú reprezentáciu, a je dôležité dáta dôkladne preskúmať a zvoliť vhodný spôsob reprezentácie.

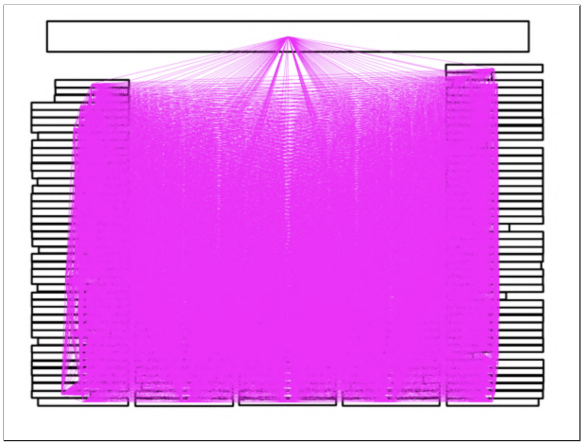
Metóda k-najbližších susedov je pomerne bežnou a má mnoho aplikácií v praxi vďaka svojej jednoduchosti [6, 60]. Na druhú stranu jednoduchosť tejto metódy ľahko naráža na viaceré úskalia, kvôli ktorým je táto metóda pomerne limitujúca pri komplikovanejších štruktúrach dát. Obecne je prepojenie v grafe realizované pomocou k najbližších susedov (uzlov) [56]. Jedná sa teda o metódu, ktorá využíva vzdialenosť ako budovacie kritérium. Hodnota k predstavuje kladné nenulové číslo, ktoré je volené empiricky a mení sa podľa toho, ako veľmi postačujúci výstup dostaneme. Vzdialenosť medzi dvomi uzlami je v tejto práci uvažovaná v euklidovskom priestore, kde v prípade 2D priestoru ide o vzdialenosť medzi dvomi bodmi ktoré sú reprezentované v súradnicovej sústave dvojicou (x, y). Výpočet samotnej vzdialenosti je realizovaný ako bežná euklidovská vzdialenosť dvoch bodov [46]

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}, \tag{3.12}$$

kde p a q reprezentujú body stredov riadkov. Na záver je teda vytvorená hrana medzi k najbližšími susedmi uzlu v. Výsledný graf je vždy orientovaný. Vizualizácia takéhoto grafu, vytvoreného na článku z novín datasetu Lidové noviny je na obrázku 3.1. V prípade, že by bola zvolená hodnota k rovnaká, ako je celkový počet uzlov, tak by sa vždy jednalo o plne prepojený graf, čo je v tejto práci nevyhovujúce kritérium kvôli svojej vysokej výpočetnej a pamäťovej náročnosti, keďže stránky obsahujú mnohokrát veľké množstvo riadkov. Príklad výstupu tvorby grafu s plnou prepojenosťou je na obrázku 3.2.

3.3 Typy riešených úloh nad grafmi

Existuje niekoľko typov úloh, ktoré je možné vykonávať pri používaní grafových neurónových sietí. V tejto sekcii si ale predstavíme len tie, ktoré sú ďalej predmetom skúmania tejto práce. Prvou z úloh, ktorá je jednou z najčastejších aplikácií GNN, je klasifikácia uzlov do svojich tried. Podobne je možné sa zamerať aj na klasifikáciu hrán do tried a vo svojej podstate sa tieto dve úlohy od seba až tak nelíšia.



Obr. 3.2: Graf vytvorený pomocou metódy k-NN pre ($k = 121$) nad článkom z datasetu Lidové noviny. Tento graf je vo svojej podstate plne prepojeným, keďže je hodnota k rovná počtu riadkov na stránke. Ako je možné pozorovať, už len z vizuálneho hľadiska sa graf stáva nečitateľným a jeho počet hrán predstavuje výpočetné komplikácie v priebehu trénovania.

Klasifikácia uzlov a hrán

Klasifikácia uzlov v GNN je úlohou predikcie tried individuálnych uzlov v grafe. Tento typ úlohy má pomerne široké zameranie [61, 57], pričom je možné ho využiť napríklad pri detekcii podvodov vo finančnom sektore, identifikácii určitých abnormalít vo vzťahoch medzi uzlami a podobne. Jedná sa teda o úlohu, kde GNN využíva grafovú štruktúru a príznaky uzlov na predikciu tried nových neznámych uzlov.

Formálne definujme problém učenia reprezentácií uzlov na ich klasifikáciu pomocou grafových neurónových sietí [57]:

Definícia 3.3.1 *Nech $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ predstavuje graf, kde $\mathscr{V}$ je množina uzlov a $\mathscr{E}$ je množina hrán. $A \in \mathbb{R}^{N \times N}$ reprezentuje maticu susednosti, kde N je celkový počet uzlov, a $X \in \mathbb{R}^{N \times C}$ reprezentuje maticu atribútov uzlov, kde C je počet príznakov pre každý uzol. Cieľom grafových neurónových sietí je efektívne naučiť uzlové reprezentácie (označené ako $H \in \mathbb{R}^{N \times F}$, kde F je dimenzia reprezentácií uzlov), kombinovaním informácií o štruktúre grafu a atribútoch uzlov, ktoré sú ďalej použité na klasifikáciu uzlov.*

Hlavným problémom pri uzlovej klasifikácii je prísť na spôsob ako efektívne propagovať informáciu zo susedných uzlov pre daný uzol. Typicky sa na tento problém využívajú GCN architektúry, ktoré sú navrhnuté tak, aby efektívne propagovali informácie v grafe aplikovaním konvolučnej operácie na maticu susednosti, ktorá bola pre daný graf vytvorená. Pri klasifikácii uzlov sa GNN typicky snaží natrénovať tak, aby minimalizovalo chybu medzi predikovanými a skutočnými triedami uzlov. GNN je trénovaná, podobne ako aj iné neurónové siete, na množine trénovacích prvkov a následne vyhodnotená na množine testovacích. Vo výsledku sa teda jedná o pomerne klasický postup pri trénovaní klasifikačného modelu.

Úloha klasifikácie hrán je len veľmi mierna úprava oproti klasifikácii uzlov, kde sa GNN usiluje o predikciu triedy hrany, ktorá existuje medzi dvomi uzlami. Podobne, ako pri klasifikácii uzlov, môžeme formálne zadefinovať klasifikáciu hrán. Pre jednoduchosť bude problém definície uvažovaný ako binárna klasifikácia [1]:

Definícia 3.3.2 *Nech je daný graf* $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ *a množina* $\mathscr{E}_l \subseteq \mathscr{E}$ *olabelovaných hrán, kde každá hrana* $(u, v) \in \mathscr{E}_l$ *má binárnu označenie* $l_{uv} \in \{0, 1\}$*. Cieľom problému klasifikácie hrán je určiť triedy pre hrany v* $\mathscr{E}_u = \mathscr{E} \setminus \mathscr{E}_l$*.*

Pri tomto type úlohy je vhodné využiť taktiež hranové príznaky, ktoré pri vhodnej reprezentácii efektívne dopomáhajú zlepšiť kvalitu modelu a výslednú predikciu. Je však potrebné počítať aj s tým, že nie všetky GNN podporujú vo svojej architektúre hranové príznaky.

Kapitola 4

Dátové sady pre analýzu rozloženia

Existuje pomerne veľké množstvo dátových sád, ktoré sú vhodné pri dokument layout analýze (DLA). Je však veľmi dôležité predom stanoviť úlohy, na ktoré budú dátové sady používané, pretože analýza rozloženia je pomerne široká sféra a existuje množstvo úloh, ktoré sa dajú riešiť. V prípade viacerých typov zvolených úloh sa nedá predpokladať, že bude existovať univerzálny dataset, ktorý by obsahoval všetky potrebné informácie na ich splnenie. Preto je niekedy potrebné nájsť väčší počet rôznych datasetov. Výsledná sumarizácia všetkých datasetov, ktoré boli skúmané, alebo sú v práci priamo použité je v tabuľke 4.1.

Aby bolo možné jednotlivé datasety analyzovať, je dôležité si na úvod predstaviť niektoré z najpoužívanejších formátov ukladania dát pri úlohe DLA. Formáty typu JSON alebo CSV je možné považovať za triviálne, a preto sa ich popis zanedbá. Podobne by tomu mohlo byť aj pri generickom formáte XML, avšak existujú niektoré varianty, ktoré je dôležité si predstaviť z toho dôvodu, že ich používanie nie je vo všeobecnosti až tak bežné.

Formát Page XML [41] je v súčasnosti jeden z najpopulárnejších formátov pri zachytení a vhodnom zakódovaní informácií o rozložení stránky. Tento formát je rovnako vhodný pri ukladaní informácií z výstupu OCR. Vo formáte Page XML je možné zaznamenávať informácie o charakteristikách obrazu (ohraničenia obrázkov, geometrické deformácie a ich korekcie, binarizácia atď.), štruktúru rozloženia fyzických komponent a ich logický význam. Formát má hierarchickú štruktúru, tvorenú elementami, ktoré reprezentujú rozličné komponenty stránky, ako napríklad riadky, slová, znaky a obrázky. Každý z týchto elementov môže mať priradený atribút, ktorý popisuje jeho vlastnosti, ako napríklad pozícia na stránke, veľkosť, font a štýl. Page XML súbory sú typicky vygenerované pomocou spomínaného OCR nástroja alebo pomocou nástrojov na analýzu rozloženia.

Dataset ReadingBank

Dataset ReadingBank je vytvoreným datasetom, ktorý je vhodný na trénovanie a vyhodnotenie postupnosti čítania. Anotovanými sú sekvencie slov s korešpondujúcimi súradnicami, ktoré ohraničujú takéto slovo. Dataset obsahuje 500,000 dokumentov stránok, ktoré sú rozdelené na 400,000 trénovacích, 50,000 evaluačných a 50,000 testovacích vzorkov. Každá zo stránok má taktiež uvedenú hodnotu metriky BLEU [55]. Táto metrika bola použitá autormi pri vyhodnotení ich vlastného modelu, ktorý je založený na technike sequence-to-sequence [53]. Dataset je uložený vo formáte JSON a príklad obrazu dokumentu je zobrazený na obrázku 4.1b.

Dataset cBAD

Dataset cBAD (*angl.* Competition on Baseline Detection) [13] je dataset vytvorený pri súťaži na detekciu baseline. Obsahuje 3,021 obrázkov dokumentov, ktoré sú rozdelené v pomere 50:25:25 na trénovaciu, evaluačnú a testovaciu sadu. Obsah dokumentov je tvorený z historických spisov a kníh v latinke a rôznych typov jazykov. Formát datasetu je Page XML a anotácie sú na úrovni blokov textu, kde každý tento blok je zložený z riadkov. Jednotlivé textové elementy majú uložené informácie o geometrickom rozložení (pozície súradníc) a taktiež obsahujú informácie o poradí blokov a samotných riadkov. Informácie o postupnosti čítania nie sú uložené v hierarchickej štruktúre Page XML, ale sú uložené vo voliteľnom atribúte každého elementu. Empirickým otestovaním v nástroji Aletheia[1] bolo ale zistené, že samotné poradie riadkov je na viacerých stránkach chybné. Príkladom bola stránka, ktorej text bol v latinke a bol fyzicky formátovaný do separátnych stĺpcov, pričom poradie čítania bolo nezmyselne anotované zprava doľava. Pri jednoduchších rozloženiach bolo poradie čítania anotované správne. Treba však podotknúť, že tento dataset nevznikol so zámerom otestovať reading order. Nedostatkom sú taktiež verejne nedostupné anotácie testovacej sady. Príklad datasetu je na obrázku 2.4b.

Dataset DocLayNet

Dataset DocLayNet [40] obsahuje 80,863 PDF stránok, z ktorých 7,059 má dve ručné anotácie a 1591 tri, kde je teda dohromady 91,104 anotovaných instancií. Tieto anotácie sú vo forme označených bounding boxov, ktoré obsahujú informácie o rozložení. Celkovo má dataset 11 druhov tried charakterizujúcich rozloženie, ako napríklad obrázok, nadpis, titulok, paragraf, tabuľka a pod. Príklad z takéhoto datasetu je na obrázku 4.1a.

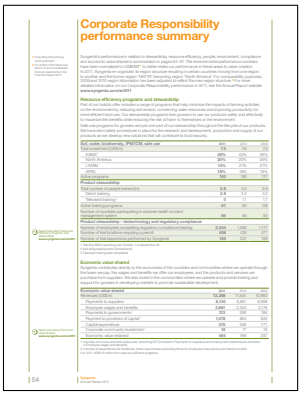
Dataset DocBank

Dataset DocBank [35] obsahuje 500,000 stránok dokumentov s 12 rôznymi logickými typmi (autor, titulok, rovnica, obrázok, nadpis atď.). Tieto stránky sú organizované na základe poradia čítania, čo znamená, že textové bloky a netextové elementy sú zoradené od vrchu po spodok podľa ich pozícií vrchného ohraničenia (vrch bounding-boxu). V textových blokoch sú textové riadky znovu zoradené od vrchu po spodok. Všetky textové riadky sú tokenizované a anotované zľava doprava, a teda všetky tokeny sú usporiadané od vrchu po spodok a zľava doprava, vrátane dokumentov, ktorých štruktúra je rozdelená do viacerých stĺpcov. Príklad z tohto datasetu je na obrázku 2.4c. Dataset používa na vyhodnotenie metriku skoré F_1.

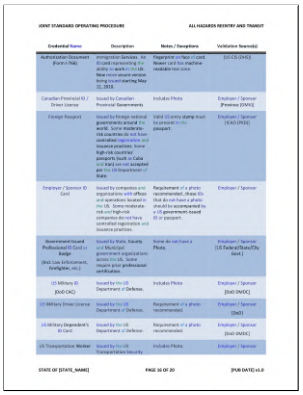
Datasety PRImA

Dataset IMPACT [39] obsahuje viac ako pol milióna historických obrázkov s textom, zložených prevažne z kníh a novín, rôznych druhov jazyka písaného v latinke alebo cyrilike. 300 obrázkov má veľmi detailnú ground truth, ktorá vytvára dohromady 5,000 textových riadkov a 70,000 slov. K tomu má 45,000 obrázkov anotované reading order a textové regióny čo vytvára dohromady 573,725 textových regiónov, ktoré sú zložené z paragrafov, nadpisov, titulkov. Dataset je uložený vo formáte Page XML [39]. Príklad z datasetu IMPACT je na obrázku 4.1c. Napriek snahe získať celý dataset a využiť jeho plný potenciál

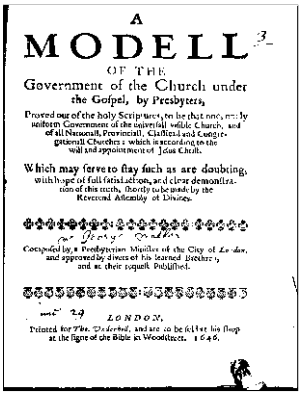
[1] https://www.primaresearch.org/tools/Aletheia



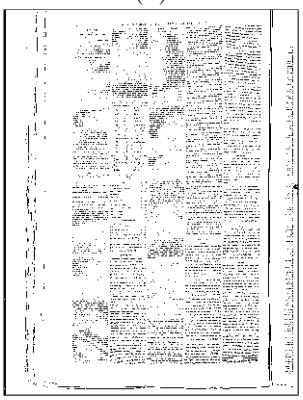
(a)



(b)



(c)



(d)

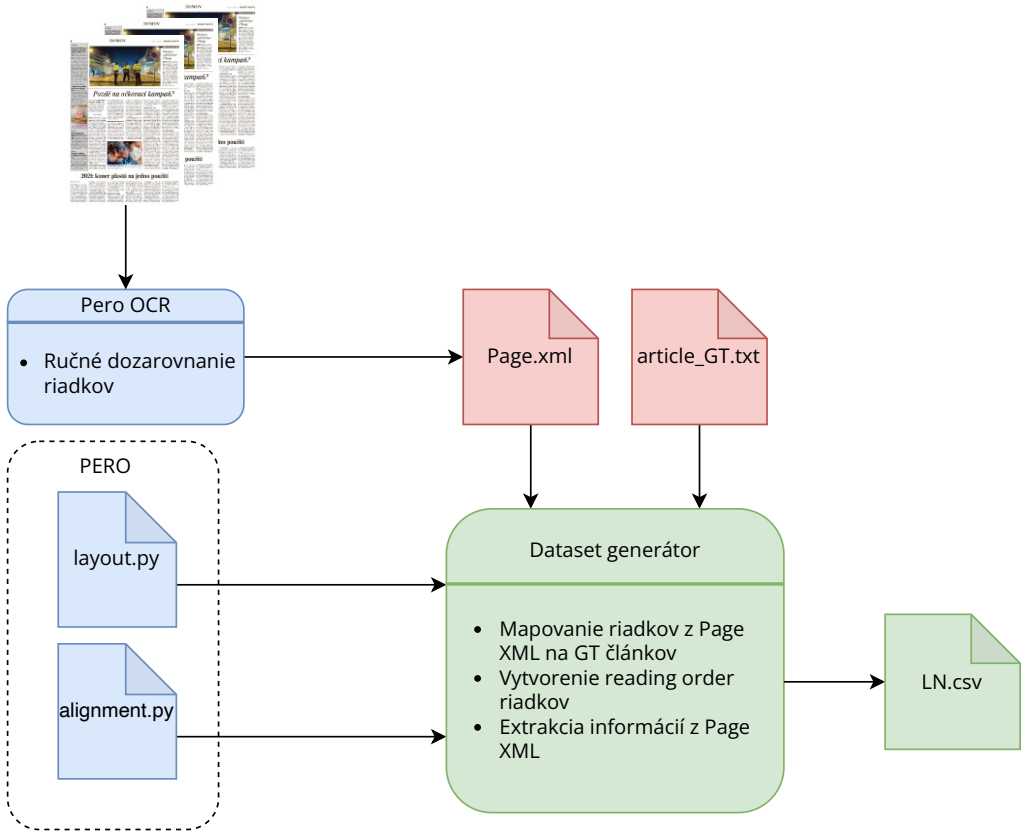
Obr. 4.1: Kolekcia vzorkov dokumentov z jednotlivých datasetov. (a) Dokument obrázku z datasetu DocLayNet [40]. (b) Obrázok z datasetu ReadingBank [53]. (c) Archívny dokument z datasetu IMPACT [39]. (d) Dokument obrázku z datasetu Europana [9].

sa toto počas tvorby práce nepodarilo. Bolo však možné získať istú podmnožinu tohto datasetu vďaka výskumnému tímu z Fakulty Informačných Technológií v Brne (FIT VUT). Detailnejší popis prípravy dát a ich použitie je popísaný v sekcii 4.2.

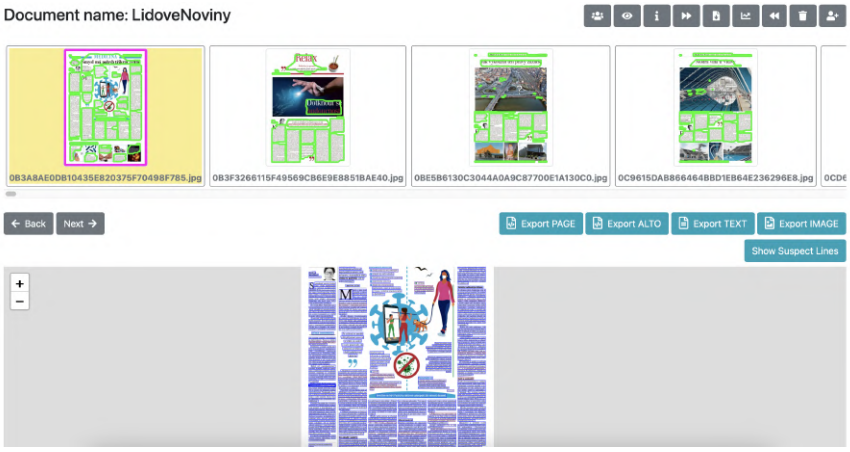
Dataset Europana [9] je charakterovo veľmi podobný datasetu IMPACT, ale jeho rozsah je oveľa menší. Je zložený z 528 obrázkov, ktoré majú anotované rôzne typy regiónov, ako napríklad obrázky, textové regióny a 202,524 riadkov. Taktiež je v ground truth uložený reading order regiónov. Formát je, podobne ako pri IMPACT, Page XML [9]. Príklad z datasetu Europana je na obrázku 4.1d.

4.1 Datová sada Lidové noviny

Súčasťou tejto práce je aj vytvorený dataset Lidové noviny. Je to automaticky vytvorený dataset, ktorý je určený primárne na otestovanie schopnosti modelu určiť správne poradie čítania. Tento dataset sa skladá z 493 digitalizovaných novinových stránok. Každá stránka má pomerne komplexné rozloženie, ktoré je zložené z množstva riadkov, odstavcov, článkov, obrázkov a iných fyzických komponent. Dataset je uložený vo formáte CSV. Obrázok 4.2 vykresľuje základný postup vytvorenia datasetu.



Obr. 4.2: Základná schéma vytvorenia datasetu Lidové noviny.



Obr. 4.3: Snímka z anotovacieho nástroja Pero OCR počas procesu vytvárania datasetu Lidové noviny.

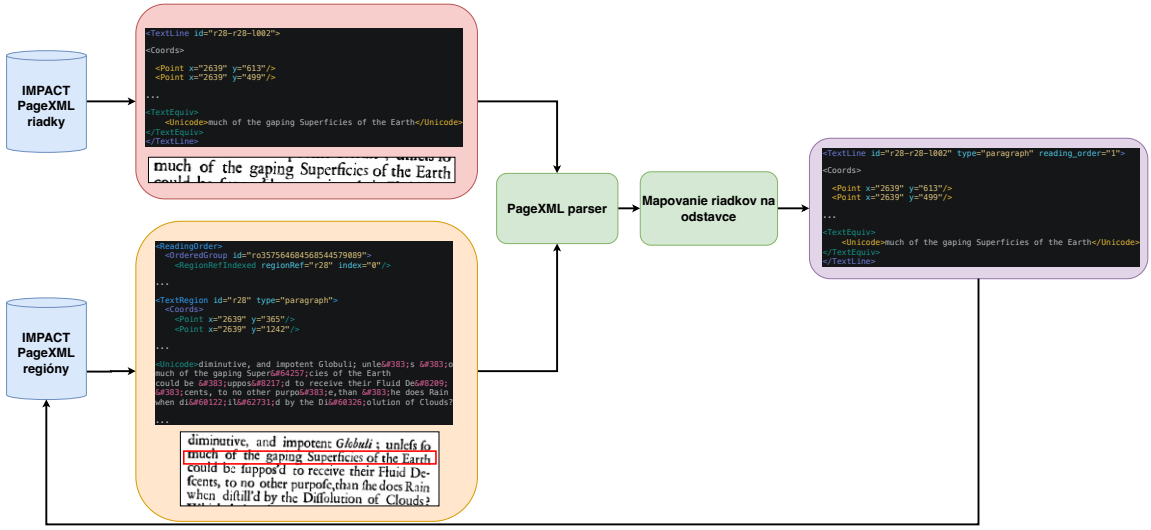
Prvým krokom bol prevod obrázkov do formátu JPG. Tieto obrázky boli následne odoslané na vstup do nástroja Pero OCR[2], kde boli jeho výstupy znovu analyzované a prípadne upravené. Prostredie z nástroja počas validácie výstupov je na obrázku 4.3.

[2] https://pero-ocr.fit.vutbr.cz/index

Obr. 4.4: Grafická reprezentácia zarovnania viacerých článkov na jednej stránke, kde je poradie riadkov postupne spojené čiarou (poradie sa vzťahuje na každý článok zvlášť).

Výsledkom tohto nástroja boli vysegmentované oblasti textu na úrovni textových blokov, kde každý blok je zložený z textových riadkov. Segmentácia nie je bezchybná, a preto bolo potrebné niektoré riadky na stránkach ručne zarovnať, prípadne doplniť. Všeobecne sa však nástroj OCR s formátom novodobých novín vysporiadal veľmi dobre nielen z pohľadu segmentácie oblastí, ale aj z pohľadu rozpoznávania znakov, ktoré sú podstatné pri samotnom zarovnávaní. Následným krokom je vytvorenie korektnej ground truth z pohľadu reading order, k čomu slúži vytvorený Python skript. Ten má na vstupe vytvorené Page XML súbory z nástroja OCR a textové súbory, ktoré boli softwarovo vygenerované a dodané k obrázkom. Každý z týchto textových súborov obsahuje informáciu o tom, ku ktorému obrázku sa vzťahuje a obsahuje samotný text, ktorý sa na obrázku nachádza, kde však nastáva problém, že text je odlíšený len z pohľadu väčších odstavcov, nadpisov, titulkov a pod., čo značne komplikuje zarovnanie na úrovni riadkov, ktoré je potrebné vo výsledku získať. Taktiež je dôležité dodať, že doložené textové súbory sú organizované podľa novinových článkov, čo znamená, že sa môže vzťahovať aj viac textových súborov na jeden obrázok, a teda jeden Page XML súbor. Táto skutočnosť taktiež obmedzila vytvorenie ground truth postupnosti čítania pre celú stránku.

Na samotné zarovnanie boli využité skripty z projektu PERO, menovite `layout.py`, ktorý slúži na spracovanie súboru formátu PAGE XML a `alignment.py`, ktorý poskytoval vhodný algoritmus na hľadania podobných podreťazcov vo väčšom reťazci, vďaka čomu bolo možné hľadať zhody riadkov z Page XML v súvetiach z textového súboru. Pomocou tohto spôsobu, a s vhodným využitím geometrických informácií (pozície na stránke) bolo zostavené poradie riadkov pre každý z článkov na všetkých stránkach. Príklad takéhoto výstupu v grafickej podobe je možné vidieť na obrázku 4.4. Samotný výstup CSV obsahuje informácie o poradí riadkov podľa článkov, text získaný z výstupu OCR a geometrické informácie riadkov.



Obr. 4.5: Základný návrh vytvárania nových ground truth z regiónov na riadky.

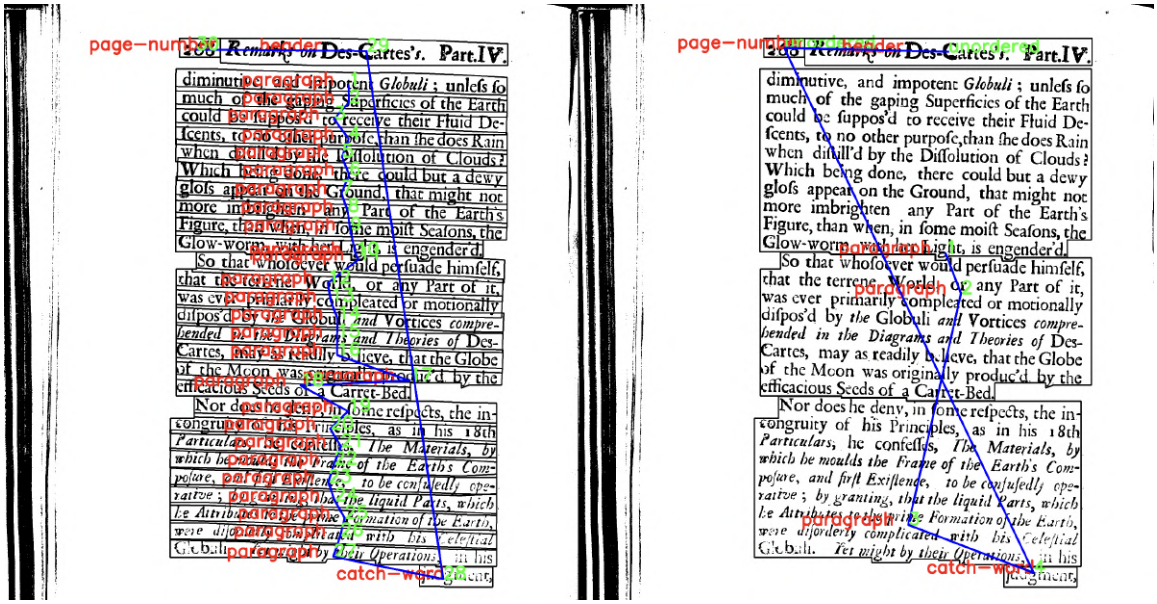
4.2 Datová sada IMPACT

Keďže sa práca zameriava na rekonštrukciu poradia čítania a súčasne aj na klasifikáciu riadkov do tried, vytvorený dataset Lidové noviny nebol postačujúci, či už kvôli malému množstvu stránok, alebo chýbajúcim anotáciam typov entít. Z tohto dôvodu bol v tejto práci vytvorený dataset IMPACT, ktorý je avšak iba podmnožinou originálneho datasetu. Povôdný dataset je obsahom anotácií a typom dokumentov ideálnym prípadom na otestovanie zvoleného prístupu. Dataset obsahuje anotácie postupnosti čítania a celkovo 15 tried identifikujúcich typ regiónu, menovite *paragraf, nadpis, hlavička, pätička, číslo strany, catch-word, marginália, kredit, iniciály, obsah, poznámka pod čiarou, podpis, pokračovanie poznámky pod čiarou, titulok* a *nepriradený text*. Avšak primárnym nedostatkom pri tejto dátovej sade sú iba anotácie na úrovni celých odstavcov, pričom cieľom je vykonávať analýzu dokumentov na úrovni riadkov.

Z tohto dôvodu bol vytvorený prístup, ktorý automaticky transformuje pôvodné anotácie z vyššej granularity na nižšiu pre anotácie poradia čítania a tried komponent v dokumente. Pôvodný dataset je počtom stránok pomerne veľký, no napriek všetkým snahám nebolo možné získať všetky stránky, ale podmnožinu približne 27 tisíc anotovaných stránok. Tieto dokumenty už boli predom spracované pomocou nástroja Pero OCR, ktorý vygeneroval nové Page XML na úrovni riadkov (bez anotácií).

Navrhnutý prístup uvažuje ako prerekvizitu to, že OCR nástroj ukladá svoje výsledky spracovaných riadkov do istej miery sekvenčne, aspoň z pohľadu úrovne odstavcov. Na obrázku 4.5 je základná schéma automatického pretransformovania úrovne anotácií.

Na vstupe sú samotné súbory Page XML pôvodnej dátovej sady obashujúce regióny a Page XML riadkov vytvorené z OCR. Tieto súbory sú následne parsované a výsledky sú ukladané vo vhodných dátových štruktúrach pre navhrnutý prístup, ktoré obsahujú informácie ako napríklad súradnice konvexných obalov, unikátne identifikátory komponent a samotné anotácie pre reading order a typ odstavca. V kroku mapovania riadkov na odstavce sa na úvod zoradia odstavce podľa daného reading order poradia. Ani v tomto prípade nie je možné uvažovať s úplným zoradením odstavcov, keďže v niektorých prípadoch sa táto



Obr. 4.6: Na obrázku (vpravo) je pôvodná ground truth datasetu IMPACT pre celé odstavce. Je vidieť, že niektoré odstavce nemajú dané poradie čítania, tzv. `unordered` anotácie. Príklad konečného výstupu (vľavo) vizualizuje každý riadok s novo pridelenou triedou a poradím čítania. Z dôvodu nezoradených regiónov sa výsledok odzrkadľuje aj pri samotných riadkoch, preto je nutné počítať s tým, že výstupy nie sú bezchybné.

Tabuľka 4.1: Prehľad datasetov, ktoré práca skúmala.

Datasety určené na analýzu rozloženia dokumentov					
Dataset	*Počet obrázkov*	*Typy dokumentov*	*Typy anotácií*	*Úroveň anotácií*	*Metrika*
ReadingBank	500k	WORD dokumenty	Reading order	Slová	BLEU [55]
cBAD	3021	Historické	Baseline riadkov, Reading order	Riadky	F-1 skóre
DocLayNet	80K	Rôzne	Typy textu	Odstavce	mAP@0.5-0.95 [38]
DocBank	500K	arXiv dokumenty	Typy textu, Reading order	Slová	F-1 skóre
Europana	528	Historické	Typy textu, Reading order	Odstavce	Vlastné[8]
IMPACT	500K	Historické	Typy textu, Reading order	Odstavce	Vlastné[8]
IMPACT (vlastný)	27K	Historické	Typy textu, Reading order	Riadky	Vlastné (sekcia 2.7)
Lidové noviny	493	Noviny	Reading order	Riadky	Vlastné (sekcia 2.7)

informácia v ground truth nenachádza, a teda takýto odstavec je zaradený do skupiny `UnorderedGroup`. V takýchto prípadoch v algoritme nie je implementovaná žiadna heuristika, ktorá by vedela chýbajúcu anotáciu spätne vytvoriť, a teda po lineárnom zoradení všetkých regiónov, ktoré majú dané poradie čítania sa následne pripájajú nezoradené regióny sekvenčne. Následne sa vyberajú postupne zoradené regióny a všetky riadky, ktoré na danej stránke sú, a pre každý riadok sa spočíta prienik medzi konvexným obalom regiónu a daného riadku, pričom sa zohľadňuje aj samotná pozícia útvarov. Ak je výsledok prieniku vyšší ako zvolený prah, tak sa riadok nachádza v danom regióne a je mu priradená trieda regiónu, a sekvenčne aj poradové číslo. Ak sa posunieme na ďalší odstavec v poradí, tak sú poradové čísla riadkov prideľované s offsetom posledného riadku z predošlého odstavca. Na záver sa takto zozbierané informácie o nových labeloch spätne priradia do pôvodného Page XML regiónov, čím sa vytvoria nové entity `TextLine`. Príklad konečného výstupu je na obrázku 4.6.

Kapitola 5

Návrh a implementácia riešenia

Cieľom tejto práce je navrhnúť a experimentálne overiť kvalitu a úspešnosť grafových neurónových sietí pri riešení problému analýzy rozloženia dokumentu, a to z pohľadu fyzickej aj logickej analýzy. Na základe predošlých štúdií bol navrhnutý systém, ktorý je efektívny z pohľadu výpočetnej náročnosti s ohľadom na kvalitu výstupu. Systém rieši dva konkrétne problémy z pohľadu analýzy dokumentu. Prvým typom úlohy je správna klasifikácia triedy homogénnych komponent na úrovni riadkov, akými sú nadpis, záhlavie, poznámka pod čiarou, paragraf, marginálie a iné. Celkovo sa práca venuje klasifikácií riadkov do 15 rôznych tried. Zároveň je samotný model vytvorený tak, že je schopný zvládnuť aj druhý typ úlohy, a to je identifikácia postupnosti čítania, ktorá je veľmi dôležitá pre celkové pochopenie kontextu obsahu. Postupnosť čítania je identifikovaná podobne ako pri klasifikácií textu na úrovni samotných riadkov.

Na obrázku 5.1 je základná schéma navrhnutého systému na analýzu rozloženia, ktorý začleňuje v sebe riešenie úloh, na ktoré sa táto práca zameriava. Tento systém sa skladá z viacerých komponent, ale celok je možné rozdeliť na dve hlavné časti.

Prvá je dátová časť, v ktorej sa načítajú vstupné dáta. Datasety Lidové noviny a upravený IMPACT boli vytvorené špecificky pre túto prácu a sú použité počas celej implementačnej a experimentálnej fázy. Konečný formát dát je v štruktúrovanom formáte CSV, kde jeden CSV súbor korešponduje k jednej dátovej sade, a teda vo výsledku sú informácie uložené v dvoch súboroch, pričom pôvodne bol pre každú stránku jeden samostatný Page XML súbor. Táto konverzia bola vykonaná z dôvodu zjednodušenia samotného vstupu, keďže CSV je svojou jednoduchosťou veľmi ľahko spravovateľné, oproti komplikovanejšej štruktúre Page XML. Na vstupe do programu je vždy iba jeden z datasetov. Z týchto dát sa extrahujú dôležité príznaky pre uzly grafu, kde každý uzol reprezentuje riadok na stránke. Pre uzly sa získajú dôležité črty, ktorými sú *bounding box* (dva body definujúce ohraničujúci obdĺžnik), stred daného bounding boxu, ako aj jeho výška a šírka, podobne ako v prácach [50, 51]. Dohromady je to 8 geometrickým príznakov. Tieto príznaky sa na záver normalizujú, a následne zakódujú ako uzlové embeddingy.

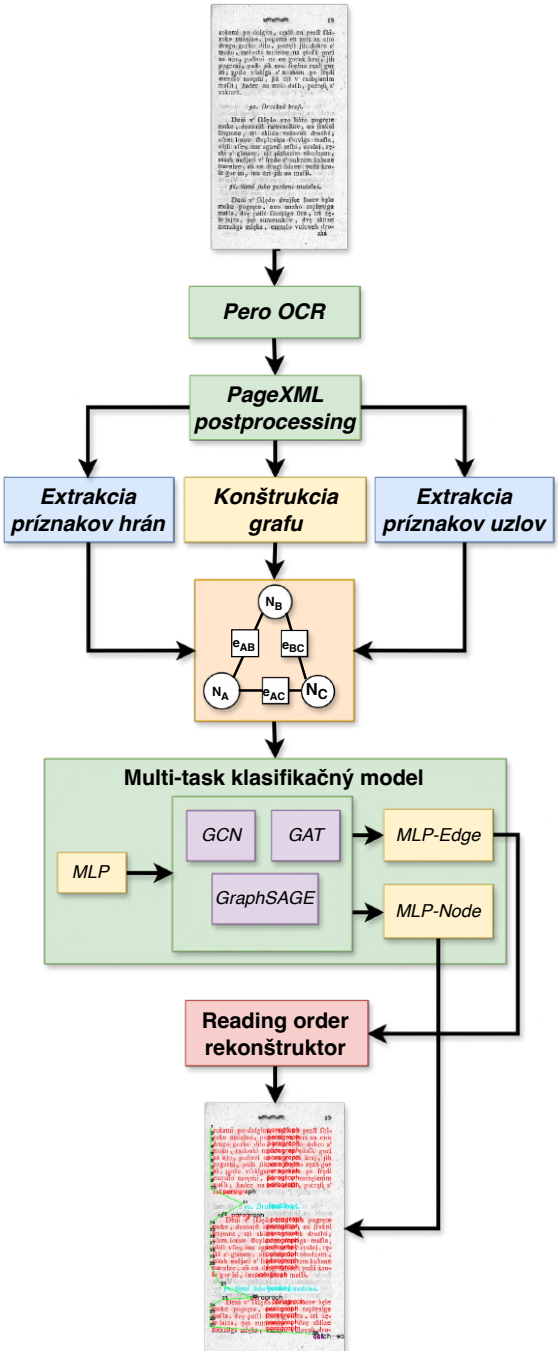
Následne je na rade extrakcia príznakov pre hrany, ktorými je vzdialenosť dvoch komponent (riadkov) v euklidovskom priestore. V ďalšom kroku sa aplikuje metóda, ktorá vytvorí samotný graf, tým že prepojí uzly hranami. Na tvorbu grafu je použitá metóda k-NN, podobne ako v práci [6], kde voľba parametru k (počtu susedov) je volená experimentálne na základe najlepších výsledkov. Pri voľbe väčšieho parametru môžu tým pádom vzniknúť plne prepojené grafy. To nastáva v prípadoch, že stránka obsahuje menší počet riadkov, čo však z výpočetného hľadiska netvorilo problémy, a preto sa takéto husté grafy ďalej neprerezávajú.

Druhou časťou systému je modelová. Tá ma na vstupe samotný graf s definovanými uzlami a hranami vrátane ich príznakov. Graf vstupuje na úvod do vstupnej neurónovej siete, ktorou je jednoduchý (viacvrstvový) perceptron. Hlavnú časť (tzv. *backbone*) tvorí grafová neurónová sieť, pričom podobne ako pri tvorbe grafu, bolo experimentované s viacerými typmi sietí. Výstup z tejto siete je potom vstupom znovu do n-vrstvovej plne prepojenej siete. V prípade, že sa jedná o úlohu, ktorej cieľom je vytvoriť model, schopný klasifikovať uzly aj hrany, tak je pre oba vytvorená samostatná MLP sieť, ktorá má na konci definovaný iný počet výstupov. V prípade, že sa jedná o klasifikáciu uzlov, a teda predikciu triedy daného riadku, je výstupov 15 (multiclass klasifikácia). Ak sa jedná o klasifikáciu hrán, tak sú definované práve dva výstupy, ktoré hovoria či riadok j nasleduje pred riadkom i, a teda jedná sa o binárnu klasifikáciu. Klasifikácia hrán umožňuje získať informácie o lokálnom poradí čítania, avšak nedáva nám prehľad o globálnom poradí. Z toho dôvodu je potrebné tieto lokálne zoradenia usporiadať tak, aby tvorili globálne. Tento krok sa nazýva dekódovanie globálneho reading order, a z pohľadu implementácie je popísaný v sekcií 5.2.

5.1 Popis modelu

Táto práca implementuje 2 varianty modelov. Keďže cieľom je vytvoriť prístup, ktorý bude schopný určiť sémantický význam riadku, a súčasne zrekonštruovať poradie čítania, jedná sa teda typovo o dve rozdielne úlohy, ktoré si vyžadujú dva klasifikátory, a to pre uzly a hrany zvlášť. Napriek tomuto cieľu je prvou variantou model, ktorý sa bude zaoberať iba určením poradia čítania. Je to z dôvodu, že dátová sada Lidové noviny, na ktorej sa prevádzajú experimenty, obsahuje iba anotácie poradia čítania riadkov. Druhou variantou je teda model, ktorý bude schopný určiť poradie čítania riadkov a súčasne týmto riadkom priradiť správny sémantický význam. Obe varianty sú založené na grafovej neurónovej sieti, pričom ale majú rozdielne výstupné klasifikátory. Navrhnutá architektúra modelu na obrázku 5.2 zobrazuje generickú štruktúru, kde model na rekonštrukciu poradia čítania zdieľa celkovú architektúru, ale nepoužíva klasifikačnú hlavu pre uzly, zatiaľ čo druhý model, určený pre oba typy úloh, používa celú architektúru. Vo všeobecnosti sa architektúra modelov rozdeľuje na niekoľko častí:

- **Vstupný graf** so zakódovanými príznakmi uzlov a hrán a uloženou ground truth, potrebnú na spätný prechod.
- **Vstupnej MLP** (*angl. Multi layer perceptron*) neurónovej siete, ktorej cieľom je transformovať vstupné uzlové embeddingy. Tento krok bol mnou navrhnutý a neopiera sa o iné práce. Cieľom je skúmať a experimentovať, či transformácia uzlových embeddingov pomocou vstupnej MLP siete dokáže pozitívne ovplyvniť celkovú úspešnosť modelu, alebo naopak zhoršiť. Tento krok je v danej architektúre voliteľný, pričom sa v kapitole experimentov 6 vyhodnocujú architektúry modelov pre obe varianty.
- **Grafová neurónová sieť** (GNN), kde typ vrstvy môže byť GCN, GAT alebo GraphSAGE, pričom ich využitie sa opiera o práce [16, 49, 6]. Táto sieť má na vstupe transformované výstupy z MLP a hranové príznaky. Obecne sa táto časť, podobne ako pri vstupnej MLP, môže skladať z niekoľkých vrstiev, kde po každej GNN vrstve nasleduje batch normalizácia, podobne ako to robili autori v práci [43].
- **Node-to-edge** konkatenácia, inšpirovaná prácami [51, 49]. Cieľom tohto kroku je previesť uzlové *feature* vektory, ktoré sú výstupom poslednej vrstvy v GNN na vektory



Obr. 5.1: Návrh systému analýzy rozloženia. Obrázok ilustruje všeobecný návrh systému na analýzu rozloženia stránky, ktorý rieši dva typy úloh: klasifikáciu hrán na zostavenie poradia čítania riadkov a klasifikáciu uzlov na priradenie triedy riadku do jednej zo špecifikovaných tried popísaných v sekcií 4.2. Toto generické nastavenie pipeline umožňuje experimentovanie s rôznymi typmi grafových neurónových sietí, nastavovaním parametrov a samotných datasetov, ktoré sú vstupom do pipeline. Výstupom je grafická vizualizácia predikcií natrénovaného modelu.

hrán, aby bolo možné v poslednej klasifikačnej vrstve vypočítať po aplikácií funkcie *Softmax* pravdepodobnosť výstupu pre každú hranu. Matematicky funkciu konkatenácie uzlov vyjadríme ako $h_e = [h_s^L \| h_d^L]$, kde L značí poslednú vrstvu v GNN, h_s skryté vektory uzlov, z ktorých hrany vychádzajú, h_d skryté vektory uzlov, do ktorých hrany vstupujú a $\|$ je operátor konkatenácie.

- Výstupná vrstva **MLP-edge** sa podobne, ako vstupná MLP sieť, môže skladať z niekoľkých vrstiev, pričom jej posledná vrstva má práve dva výstupy, ktoré produkujú pre každú hranu jej nenormalizované hodnoty. Táto sieť je určená práve pre úlohu rekonštrukcie poradia čítania.

- Výstupná vrstva **MLP-node** je veľmi podobná sieti MLP-edge a rozdiel spočíva v tom, že táto sieť má na vstupe priamo výstup z poslednej vrstvy z GNN, bez nutnej transformácie feature vektorov uzlov na hrany. Výstupom je klasifikácia uzlov do jednej z 15 zadefinovaných tried, a teda pre každý uzol 15 nenormalizovaných hodnôt z poslednej vrstvy. Táto sieť je určená na priradenie sémantického významu každému riadku.

- Pre obe MLP klasifikačné hlavy platí, že sa nad ich výstupmi aplikuje funkcia **Softmax**, ktorej cieľom je normalizovať výstupné hodnoty na pravdepodobnosti, pomocou čoho je možné na záver získať požadované výstupy. V prípade klasifikácie uzlov sa týmto krokom úloha končí. Pri klasifikovaných hranách sa výstup ďalej používa v algoritme na rekonštrukciu globálneho poradia čítania.

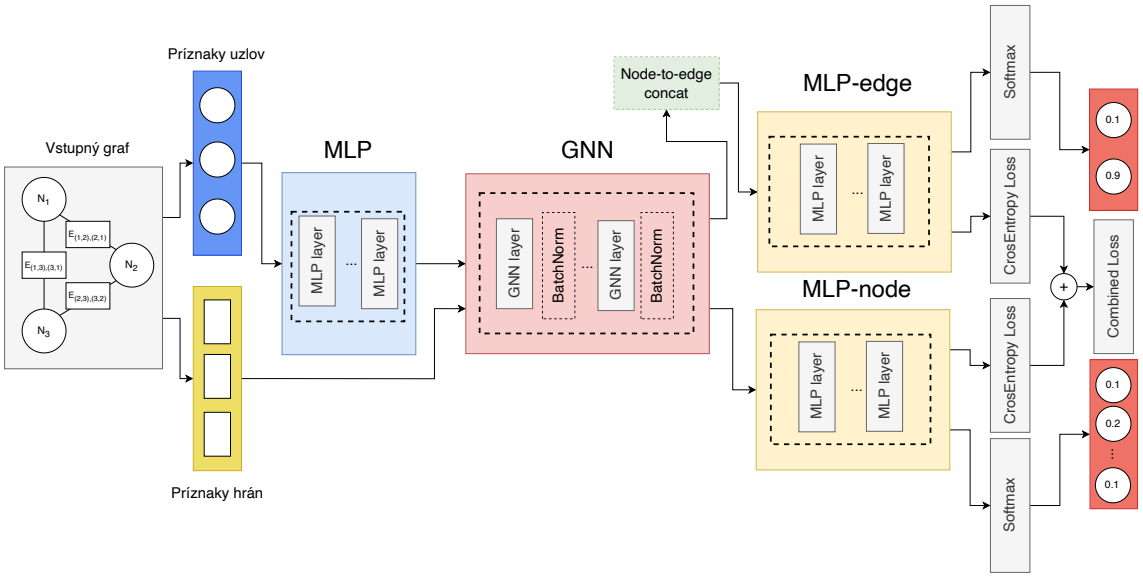
- Okrem samotnej klasifikácie je samozrejme nutné použiť aj vhodnú *loss* funkciu, aby sa sieť mohla učiť. V oboch prípadoch je použitá funkcia **CrossEntropy Loss**. Ak je cieľom vytvoriť model, ktorý vie vykonať oba typy úloh súčasne, tak sa jednotlivé hodnoty z loss funkcie sčítajú [25, 49] do tzv. kombinovanej loss hodnoty. Zároveň sa po každej vrstve (GNN aj MLP), okrem poslednej klasifikačnej vrstvy, aplikuje **aktivačná funkcia** $ReLU$.

Prvá varianta modelu je určená iba pre klasifikáciu hrán na rekonštrukciu poradia čítania. Tento model z predstavenej architektúry používa všetky časti, okrem klasifikačnej hlavy MLP-node, ktorá sa v tomto prípade vôbec nevykoná. Z toho taktiež vyplýva, že nedochádza ku kombinovaniu výstupných loss funkcií, ale je použitá iba jedna.

Druhá varianta multi-task model využíva klasifikačné hlavy MLP-edge a MLP-node dohromady podobne, ako aj v práci [49], kde bolo cieľom klasifikovať uzly aj hrany pomocou jedného modelu. Pochopiteľne, cieľom je nájsť model, ktorý bude dosahovať najlepšie výsledky pri daných úlohách. Samotnej sumarizácii a určeniu najoptimálnejšieho modelu sa venuje kapitola experimentov 6, ktorá testuje oba zvolené prístupy, experimentuje s viacerými typmi GNN vrstiev a nastavením parametrov modelu pri trénovaní.

Konštrukcia grafu

Vytvorenie konečného grafu je implementačne prakticky úplne prepojené s extrakciou príznakov a zakódovaním ground truth do grafu, ale z významového hľadiska sa jedná o dve separátne časti. Vhodná tvorba grafu je jednou z najdôležitejších súčastí pri GNN. Jej cieľom je zachytiť dôležité prepojenia hranami medzi susednými uzlami. V práci je použitá tvorba metódy grafu pomocou k-NN. Podobne ako v práci [6], parameter k je volený

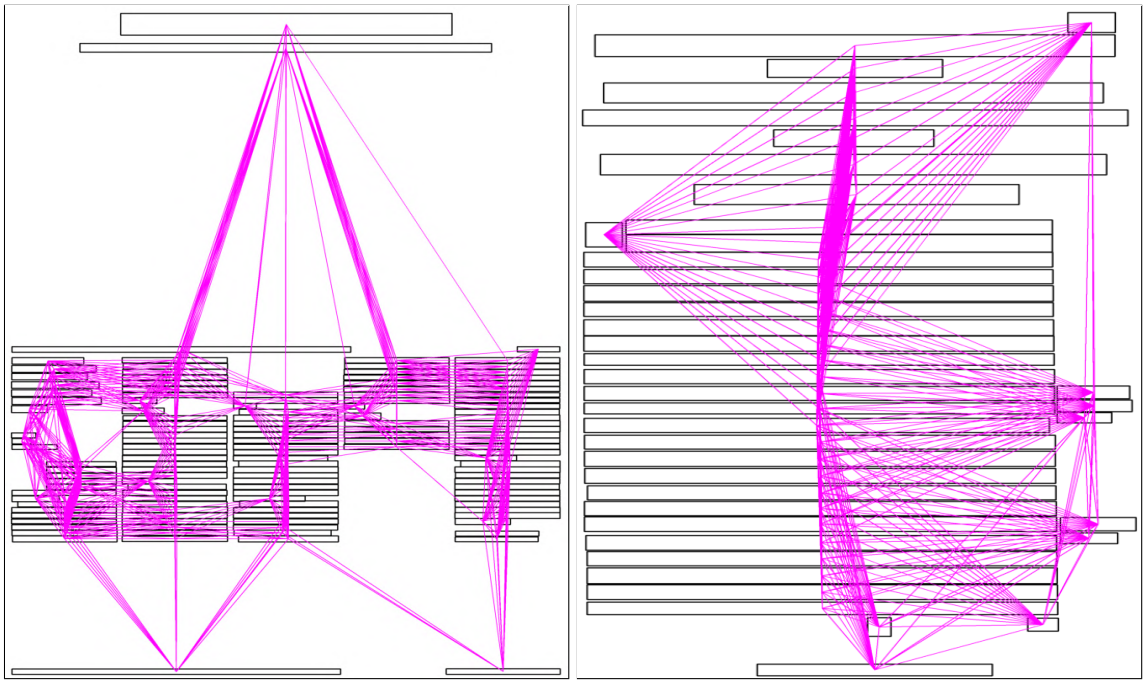


Obr. 5.2: Architektúra vytvoreného modelu, ktorá je schopná vykonávať klasifikáciu uzlov do tried, a tým priradiť sémantický význam riadku, a súčasne klasifikovať hrany do binárnych tried. Hranovou klasifikáciou získame prehľad o tom, či dva uzly, a teda riadky, za sebou nasledujú.

experimentálne a nedá sa predom povedať, aká hodnota parametru je najdôležitejšia, avšak je potrebné ho voliť tak, aby nevznikol graf s nadmerným množstvom prepojení, čo je výpočetne náročné a negatívne ovplyvňuje samotné učenie siete. Je teda dôležité mať samotný prehľad o dátach, na ktorých sa sieť bude učiť, a to najmä z pohľadu množstva uzlov (riadkov v texte). Pre zvolené dátové sady IMPACT a Lidové noviny platí teda následovne:

- **IMPACT**: Priemerný počet riadkov na stránku je 53, pričom minumum sú dva riadky a maximum 1736.
- **Lidové noviny**: Priemerný počet riadkov na článok je 89, minimum sú 2 riadky a maximum 795.

Voľba vhodného parametru zohľadňuje tieto štatistiky a snaží sa neprekročiť priemerný počet riadkov, čím bolo vyskúmané, že $k = 20$ dosahuje najlepšie výsledky pre obe dátové sady. Vzdialenosť dvoch riadkov, od ktorej je k-NN závislý, sa počíta ze stredu bounding boxu riadku. Príklady z výsledných grafov pre oba datasety sú na obrázku 5.3. Ešte predtým, ako sa vytvorí graf, pre každú stránku prebieha validácia samotnej stránky, kde je podmienkou to, že musí obsahovať aspoň dva riadky, medzi ktorými môže existovať prepojenie. Výsledné grafy sú na záver serializované a uložené na disk z toho dôvodu, že konštrukcia grafu a zakódovanie jeho vlastností je pomerne časovo náročné ak uvážime, že pri datasete IMPACT sa jedná o približne 26 tisíc grafov po odstránení nevalidných stránok. Metóda k-NN sa svojou jednoduchosťou osvedčila veľmi pozitívne pri jednoduchších typoch stránok. V prípade, že je text stránky rozdelený napríklad do viacerých stĺpcov, ako tomu býva najmä pri novinových článkoch, tak tento algoritmus začne zlyhávať pri rekonštrukcii poradia čítania. Na obrázku 5.3a je novinová stránka, ktorá pozostáva z viacerých oddelených stĺpcov textu. Ako je možné vidieť, spodné riadky nemajú prakticky nikdy prepojenie s počiatočnými riadkami nadväzných stĺpcov textu, ale z pohľadu postupnosti čítania sú práve



(a) (b)

Obr. 5.3: Vzorky výstupov z tvorby grafu pomocou k-NN nad stránkami z datasetov Lidové noviny a IMPACT. (a) Príklad grafu nad stránkou z novín datasetu LN. Je možné vidieť limitáciu prepojení vzdialenejších uzlov v členitom texte s veľkým množstvom riadkov. (b) Graf vytvorený nad historickou stránkou z datasetu IMPACT. V tomto prípade sú prepojenia medzi uzlami vhodnou reprezentáciou najmä vďaka tomu, že je text málo členitý.

tieto prepojenia esenciálne na naučenie nadväznosti. Ako už bolo skôr spomenuté, zvyšovanie parametru k by neviedlo k zlepšeniu efektívnosti neurónovej siete, a preto je potrebné tento problém vyriešiť inak, čomu sa detailne venuje sekcia 5.2.

Extrakcia príznakov a tvorba ground truth

Ako už bolo uvedené, prvotným vstupom do navrhnutého systému je CSV súbor, z ktorého sa získajú pre každú stránku (prípadne článok pri LN) potrebné informácie, ako je poradie riadku, súradnice riadku alebo jeho typ. Extrakcia príznakov sa deje pre uzly a hrany separátne. Pri uzloch sú základom 4 hodnoty identifikujúce bounding box riadku. Z týchto súradníc sa následne dopočíta výška, dĺžka a stred riadku, čím sa vo výsledku extrahuje 8 príznakov, ktoré sa na záver normalizujú. Hranové príznaky sa skladajú z jednej hodnoty, ktorá udáva vzdialenosť stredov dvoch riadkov.

Tvorba a zakódovanie ground truth sa deje pre uzly a hrany znovu oddelene. V prípade, že je vstupom dataset Lidových novín, tak sa ground truth zakóduje iba pri hranách, keďže na tomto datasete nie je možné vykonať predikciu sémantického významu riadku. Pri datasete IMPACT je ground truth pre uzly daná priamo, a nie je potrebné vykonávať akékoľvek nadväzné transformácie, a teda vo výsledku vieme, ktorému uzlu prislúcha daná trieda. V prípade vytvorenia ground truth pre hrany je potrebné prideliť každej hrane binárny label, kde hodnota 1 nám hovorí, že uzol (resp. riadok) i je pred riadkom j a hodnota

0 presný opak. Ku každému riadku máme pridelené jeho správne poradie. Ak teda vezmeme dva riadky i, j, orientovanú hranu $e_{i,j}$ a ground truth poradia riadkov r_i, r_j, potom platí:

- $(e_{i,j}, 1) \iff r_i + 1 = r_j$.
- Inak $(e_{i,j}, 0)$.

Takto sa vytvoria binárne označenia hrán, kde ak by sme zobrali iba orientované hrany s hodnotou 1, dostaneme naspäť súvislú, acyklickú cestu v grafe, ktorý bude mať práve jeden začiatočný a jeden koncový uzol. Táto cesta udáva globálne poradie čítania riadkov. Po konečnom zakódovaní labelov pre uzlu a hrany sa na záver vytvorí dátová štruktúra `Data`, vytvorená pomocou knižnice *PyTorch Geometric*[1], ktorá v sebe obsahuje všetky informácie o grafe.

5.2 Rekonštrukcia poradia čítania

Cieľom rekonštrukcie poradia čítania je previesť lokálne prepojenia uzlov, ktoré sú výstupom hranového klasifikátoru, do globálneho usporiadania, v ktorom sa usiluje o priradenie čo najviac riadkom ich korešpondujúce poradie v texte. Keďže samotný výstup z hranového klasifikátora sú binárne predikcie, kde pozitívna predikcia signalizuje, že dva riadky za sebou nasledujú, je nutné počítať s tým, že nie vždy existujú prepojenia medzi všetkými uzlami. Z toho potom vyplýva, že vo výsledku strácame informáciu o prepojení dvoch riadkov. Ak sa teda pozrieme na výstup klasifikátora detailnejšie, tak ten obsahuje po aplikácií funkcie Softmax pravdepodobnosti, ktoré nám hovoria, aká je pravdepodobnosť, že dva riadky za sebou nasledujú, resp. nenasledujú.

Algoritmus rekonštrukcie je teda nasledovný. Obrázok 5.4 zobrazuje jeden z možných výstupných grafov po klasifikácii hrán, kde zelené orientované hrany hovoria, že medzi dvojicou riadkov existuje lokálne zoradenie. Červené orientované hrany hovoria, že medzi dvojicou riadkov neexistuje lokálne zoradenie, resp. že riadok i nepredchádza riadku j. Vo výsledku nás teda negatívne (červené) hrany nezaujímajú, čím ich môžeme z grafu odstrániť. Tento krok je zobrazený na obrázku 5.5. V ďalšom kroku je potrebné v preriedenom grafe identifikovať takzvané štartovacie uzly. Tieto uzly majú vlastnosť, že počet hrán do nich vstupujúcich je rovný 0 a počet hrán, ktoré z takéhoto uzlu vychádzajú je aspoň 1. Detekcia takýchto uzlov bola implementovaná pomocou knižnice *NetworkX*[2]. V prípade, že po transformácii grafu iba pre pozitívne hrany neexistuje žiadny takýto štartovací uzol, tak je jeho voľba náhodná. Výstup tohto kroku je na obrázku 5.6. Po tomto kroku nasleduje hľadanie najdlhšej cesty podgrafu v danom grafe, kde každý podgraf má ako svoj počiatočný uzol práve jeden z nami nájdených štartovacích uzlov. Tento krok môžeme nazvať aj ako vytvorenie tzv. spojených komponent (*angl. connected components*) v grafe, a bol inšpirovaný prácou [43], kde súčasťou ich prístupu bola práve identifikácia takýchto komponent z výstupu GNN, aj keď boli následne tieto komponenty použité v inom kontexte. Ako ale ďalej pochopíme, koncept vytvárania podgrafov je esenciálny pre rekonštrukciu poradia. Pre zvolený počiatočný uzol sa následne určí jeho následník. Ak je počet výstupných hrán pre počiatočný uzol väčší ako 1, a teda počet susedných uzlov je väčší ako 1, potom sa zvolí ten, pre ktorý má daná hrana najväčšiu váhu. Tieto váhy predstavujú získané pravdepodobnosti po funkcií Softmax z výstupnej klasifikačnej vrstvy siete. Po získaní následníka počiatočného uzlu sa tento uzol určí ako počiatočný a celý proces sa vykonáva odznovu. Podmienky

[1] https://pytorch-geometric.readthedocs.io/en/latest/
[2] https://networkx.org

Algoritmus 1 Hľadanie podgrafov s najdlhšou cestou

Vstup: Zoznam hrán E s váhami $w_{uv} > 0$
Výstup: Zoznam podgrafov L

Inicializuj prázdny orientovaný graf G
Pridaj hrany z E do G s príslušnými váhami
Inicializuj prázdnu množinu $Q = \{\}$ začiatočných uzlov
Definuj $\delta^+(v)$ ako množinu odchádzajúcich hrán z uzlu v a $\delta^-(v)$ ako množinu hrán vstupujúcich do v
Priraď do množiny Q všetky začiatočné uzly $q \in G$ tak, že:

$$\forall q \in G : \{q\} \cup Q \iff \delta^+(q) \neq \emptyset \wedge \delta^-(q) = \emptyset$$

if $Q = \emptyset$ **then**
    Náhodne vyber jeden uzol $q \in G$ ako začiatočný uzol
    $Q \cup \{q\}$
end if
Inicializuj prázdny zoznam L podgrafov
Inicializuj prázdnu množinu $V_{\text{global}} = \{\}$ navštívených uzlov
for každý začiatočný uzol $q \in Q$ **do**
    Inicializuj prázdny zoznam L_{cesta} a nastav $c = q$
    while True **do**
        if $c \notin V_{\text{global}}$ **then**
            $V_{\text{global}} \cup \{c\}$
            $L_{\text{cesta}} \cup \{c\}$
        else
            L.append(L_{cesta})
            break
        end if
        Získaj susedov N uzla c: $N = \{v \mid (c, v) \in E\}$
        if $N = \{\}$ **then**
            L.append(L_{cesta})
            break
        end if
        Nájdi ďalší uzol c' s najväčšou váhou: $c' = \arg\max_{v \in N} w_{cv}$
        Nastav $c = c'$
    end while
end for

na ukončenie hľadania podgrafu nastanú vtedy, ak posledný uzol už nemá ďalších susedov, alebo ak bol detekovaný cyklus, a teda daná hrana vedie do uzlu, ktorý už bol do cesty zahrnutý. Zároveň pre všetky podgrafy platí, že nezdieľajú rovnaké uzly, a teda každý podgraf sa skladá z unikátnej sekvencie uzlov. Výstup hľadania podgrafov je na obrázku 5.7. Všetky kroky potrebné na vytvorenie podgrafov sú popísané v algoritme 1.

Zoradenie podgrafov

Vytvorenie podgrafov nám dáva už pomerne dobrú predstavu o tom, v akom poradí sa jednotlivé riadky vyskytujú, avšak stále to platí iba pre jednotlivé spojené komponenty, ktorých

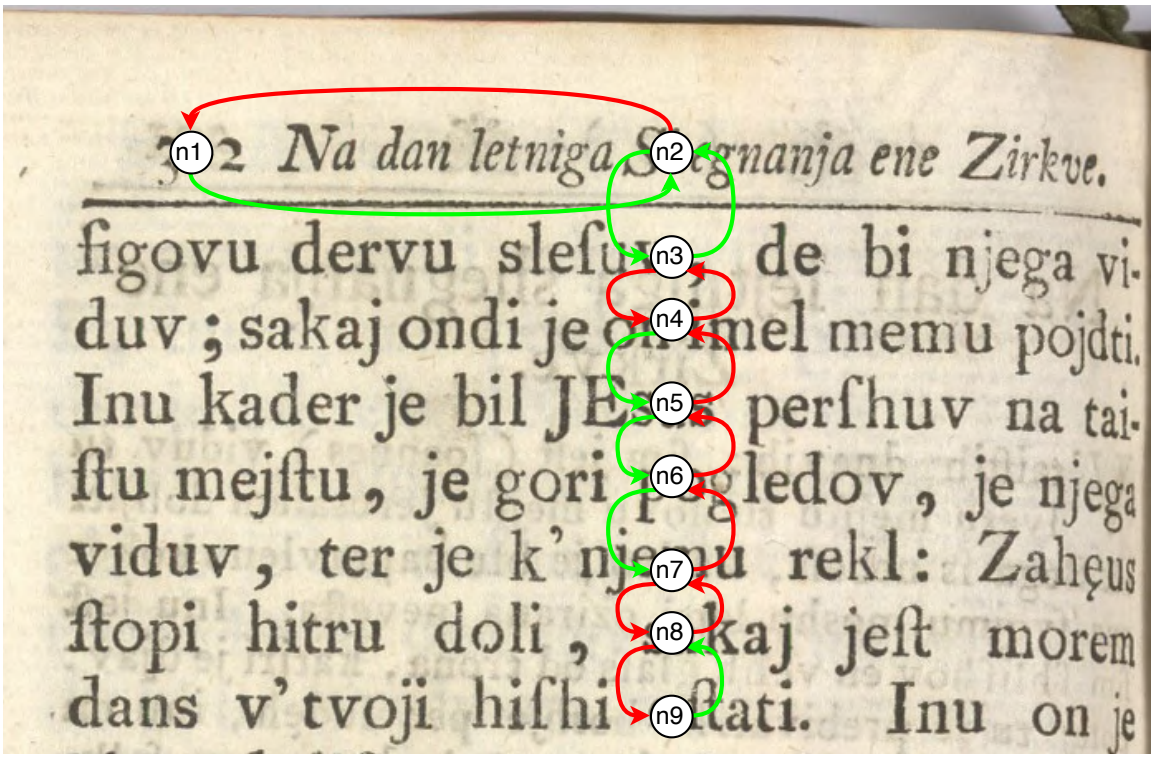
Algoritmus 2 Cluster precedence algoritmus

Vstup: Zoznam podgrafov C so zoradenými riadkami a ich bounding box súradnicami
Výstup: Zoradenie podgrafov od prvého po posledný

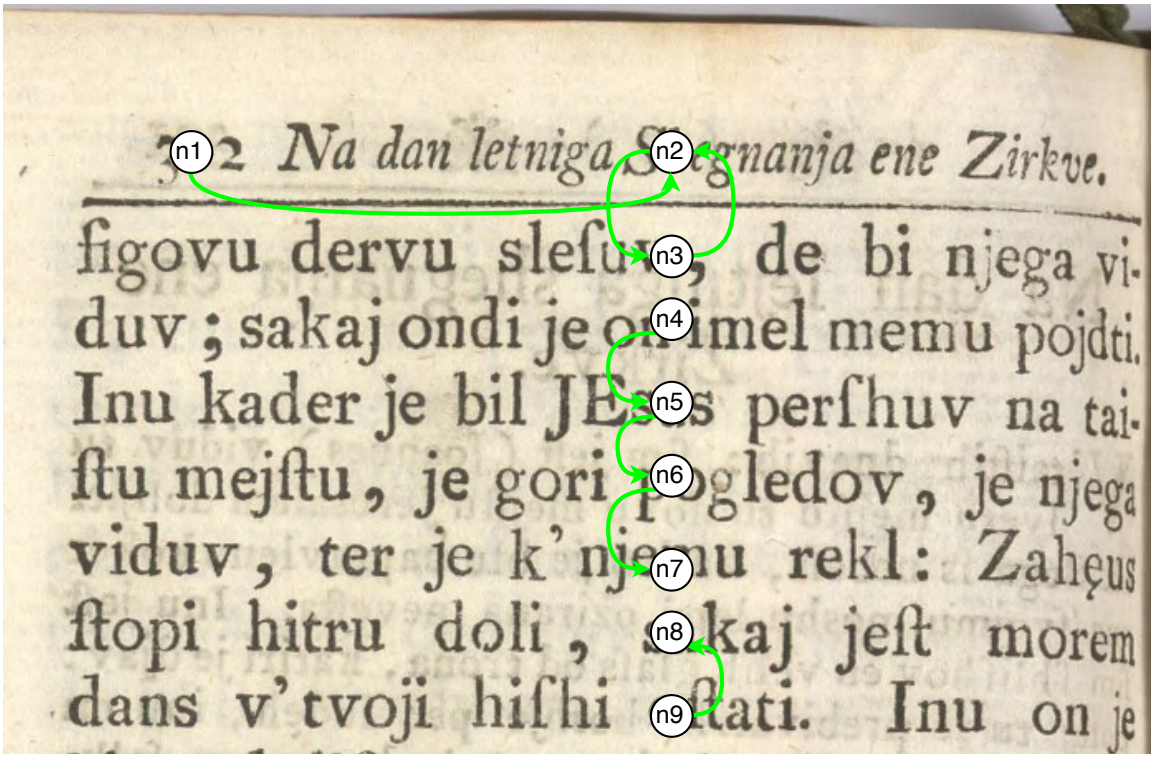
1: Inicializujme pre každý podgraf C_i z množiny C, kde $0 < i \leq n$, n je počet podgrafov, aj pomocný prázdny zoznam Children_{C_i}
2: Majme definovanú funkciu `precedes(a,b)`, ktorá pomocou pravidiel určí, či riadok a nasleduje pred riadkom b:
3: **for** každý podgraf C_a v zozname podgrafov C **do**
4: $a = C_{a1}$ je prvý riadok podgrafu C_a
5: **for** každý podgraf C_b v zozname podgrafov C **do**
6: $b = C_{b1}$ je prvý riadok podgrafu C_b
7: **if** $C_a \neq C_b \wedge \texttt{precedes}(a, b)$ **then**
8: priraď C_b ako dieťa podgrafu C_a do zoznamu Children_{C_a}
9: **end if**
10: **end for**
11: **end for**
12: Zoraď od najväčšieho $[|\text{Children}_{C_1}|, |\text{Children}_{C_2}|, \ldots, |\text{Children}_{C_n}|]$, kde $|\text{Children}_{C_i}|$ je výpočet veľkosti zoznamu detí pre cluster i a vráť výsledné poradie clustrov.

môže byť obecne niekoľko, v závislosti od presnosti predikovaných hrán. Preto je potrebné aplikovať posledný krok, ktorý nám presne určí, ktorý podgraf so svojou unikátnou postupnosťou uzlov je na stránke ako prvý, ktorý podgraf je ako druhý, atď. Na priradenie poradia podgrafom je použitý prístup pomocou definovaných pravidiel (*angl. rule based*). Na inšpiráciu v tomto kroku slúži práca [50], ktorej cieľom bolo pomocou GNN zostaviť postupnosť čítania, kde zvolili prístup *cluster-and-sort*, kde clustre boli vytvorené pomocou GNN výstupu, a následne nad týmito clustrami aplikovali pravidlá definované v práci [5], ktorá riešila rekonštrukciu poradia pomocou topologické zoraďovania. V tomto prípade sa budeme teda pozerať na jednotlivé podgrafy ako na clustre.

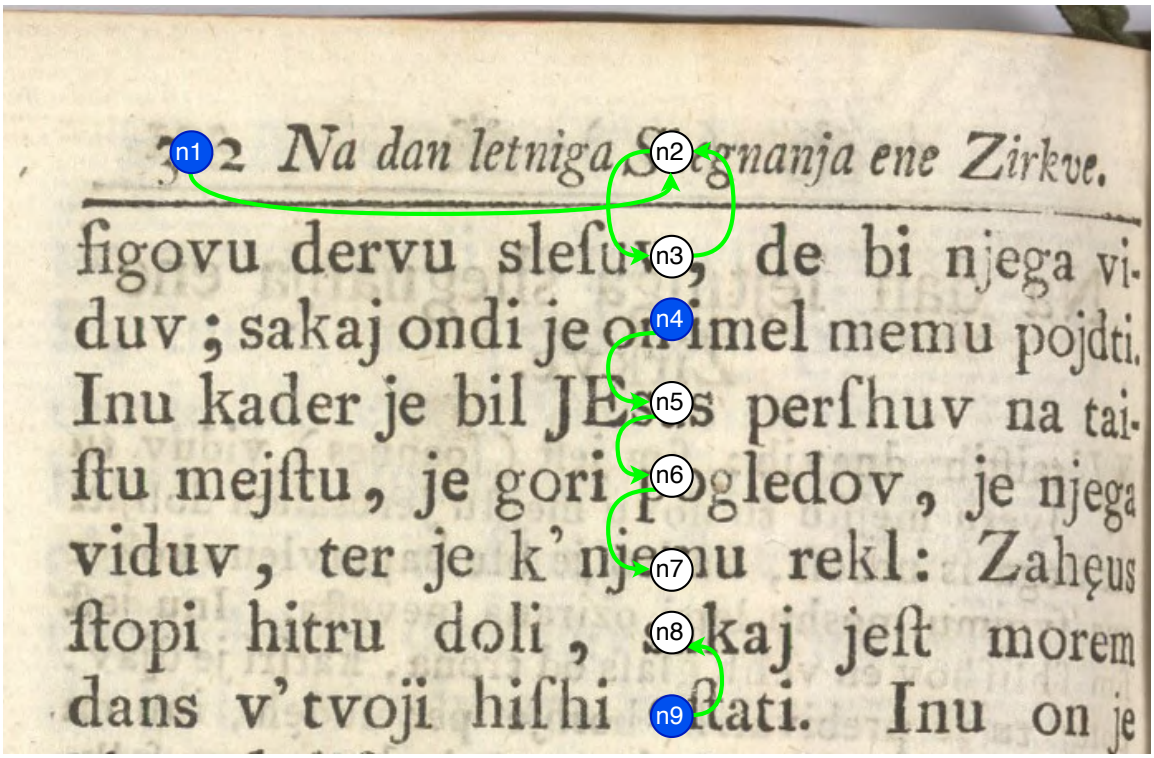
Každý z týchto clustrov sa skladá z už zoradených riadkov, čo je pre nás dôležitá informácia. Ak teda uvážime, že vieme vždy povedať, aké pozičné súradnice má prvý riadok v každom clustri, vieme veľmi jednoducho pomocou definovaných pravidiel určiť poradie medzi všetkými prvými riadkami v každom clustri, a tým pádom ďalej implikujeme aj celkové poradie clustrov voči sebe. Obrázok 5.8 obsahuje identifikované prvé riadky pre každý cluster, ktoré sa voči sebe porovnajú pomocou pravidiel. Algoritmus *Cluster precedence* 2, ktorý je doimplementovaným rozšírením rule based prístupu, priraďuje konečné poradie clustrom pomocou definovaných pravidiel a aplikuje tzv. *parent-child* hierarchiu. Každý cluster sa porovná so všetkými ostatnými a v prípade, že sa daný prvý riadok clustru i nachádza pred prvým riadkom clustru j, potom sa cluster j priradí ako dieťa clustru i. Na záver sa vyhodnotí počet detí pre každý cluster, a ten ktorý má najväčší počet detí, je identifikovaný ako prvý v celkovom poradí. Postupne sa teda týmto spôsobom určí poradie jednotlivých clustrov, resp. podgrafov. Finálne zoradenie riadkov je už iba jednoduchým dopočítaním offsetov. Platí, že každý ďalší cluster v poradí má poradie prvého riadku určené podľa poradia posledného riadku z predošlého clustru o jedna väčšie. Každý ďalší riadok v clustri má svoje poradie určené od predošlého riadku o jedna väčšie. Voľnou definíciou môžeme zapísať ako: $C_i = [r_1, r_2, ..., r_n]$ a $C_j = [r_{n+1}, r_{n+2}, ..., r_{n+m}]$, kde C_i je pred C_j. Príkladným výstupom takéhoto zoradenia je na obrázku 5.9.



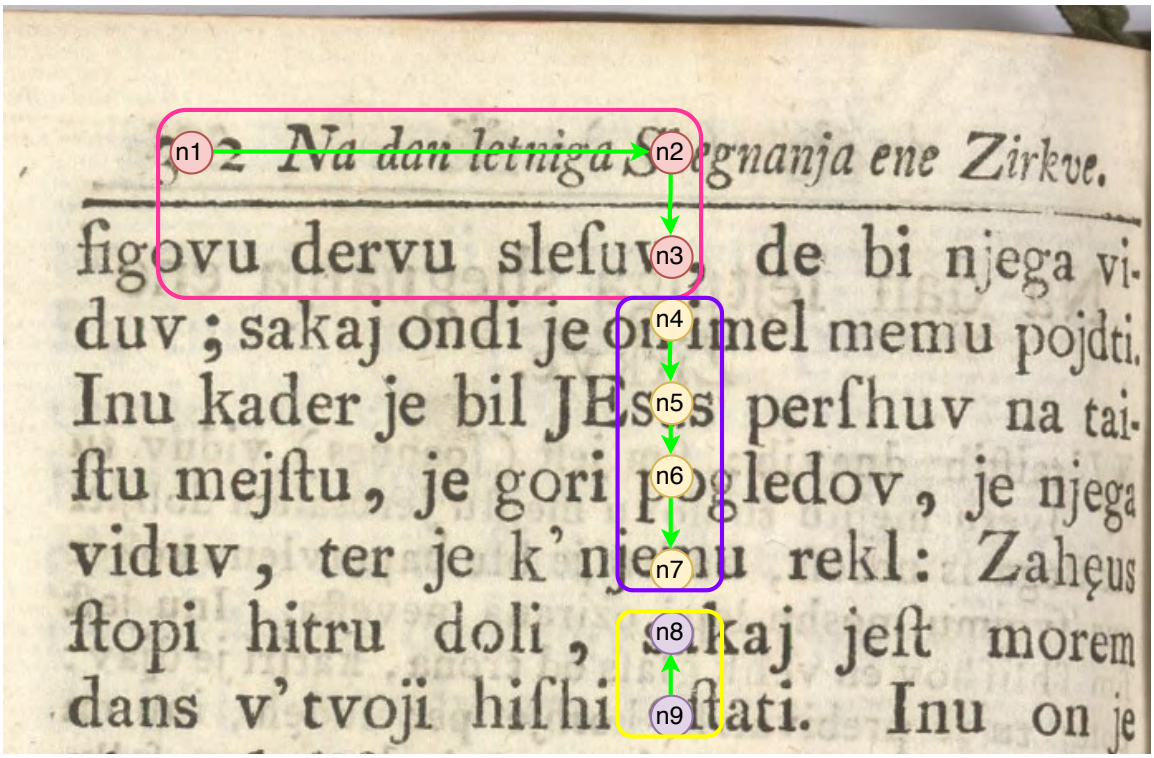
Obr. 5.4: Výstupný graf z modelu obsahujúci klasifikované hrany. Dôležité je si povšimnúť, že spodné dva riadky majú postupnosť čítania zamenenú, čo je zámerne zdôraznená chyba výstupu z modelu klasifikácie hrán, ktorá vedie aj k chybnému zoradeniu.



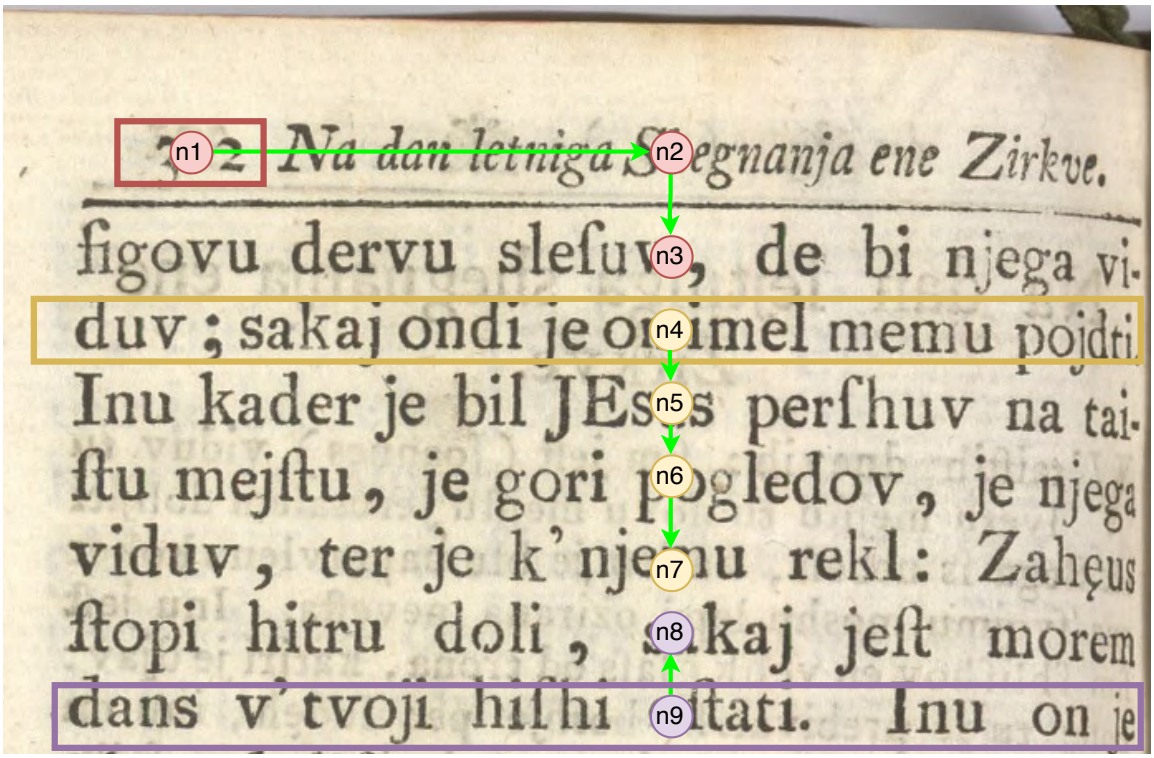
Obr. 5.5: Odstránenie negatívnych hrán z grafu.



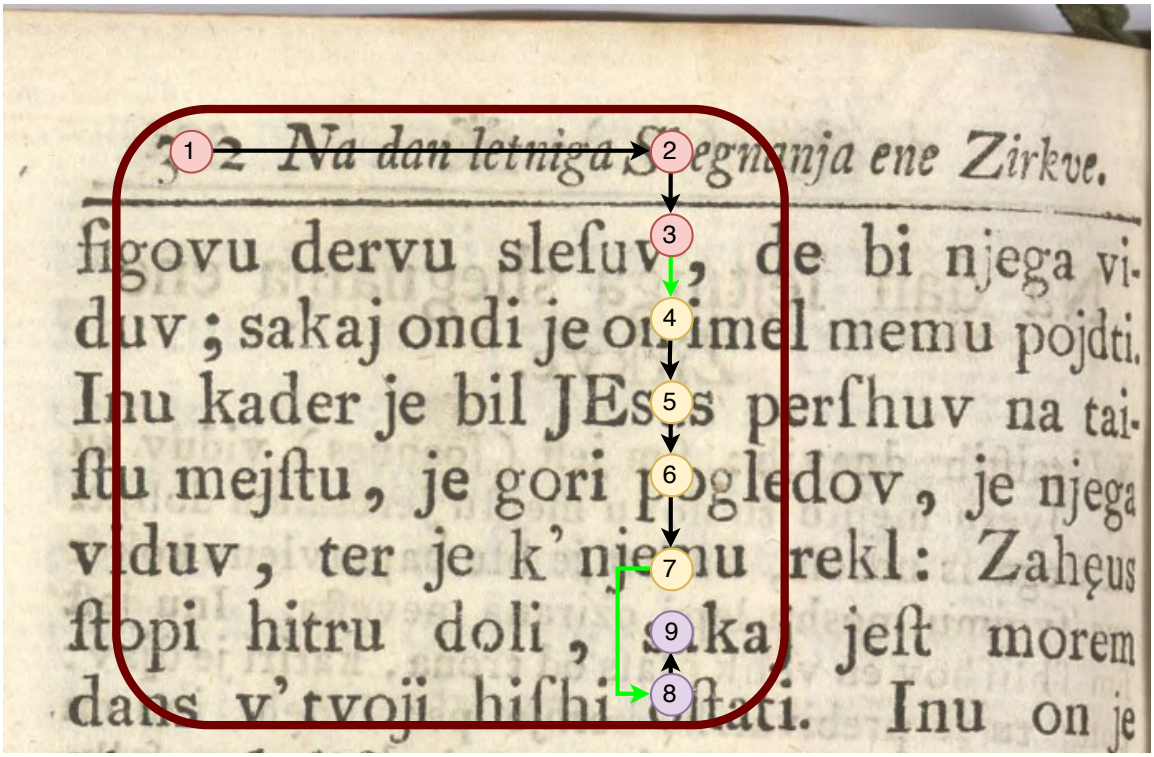
Obr. 5.6: Identifikácia štartovacích uzlov.



Obr. 5.7: Vytvorenie podgrafov (spojených komponent), spĺňajúce podmienky acyklickosti, unikátnosti uzlov, skladajúcich sa z jedného počiatočného a jedného koncového uzlu, pričom všetky ostatné uzly majú práve jednu vstupnú a jednu výstupnú hranu.



Obr. 5.8: Identifikácia prvých riadkov v podgrafoch.



Obr. 5.9: Výsledné zoradenie riadkov pomocou algoritmu Cluster precedence. Zaujímavosťou je, že pomocou tohto algoritmu sa nakoniec dotvoria chýbajúce prepojenia medzi jednotlivými podgrafmi, čím dostaneme jeden súvislý graf.

Kapitola 6

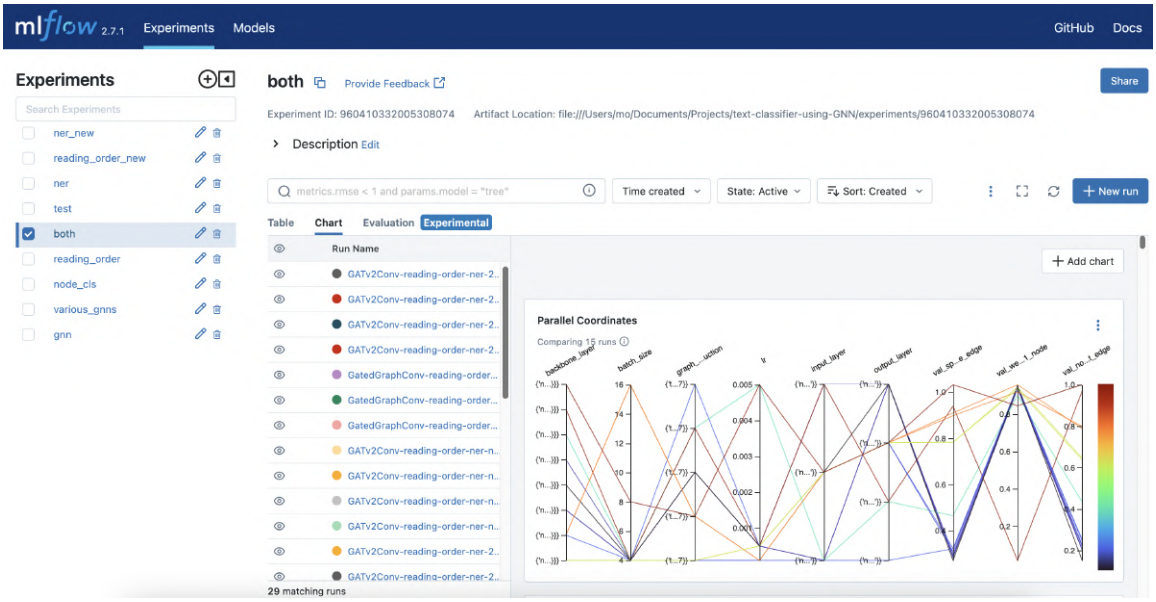
Experimenty a vyhodnotenia

V tejto kapitole je venovaná pozornosť experimentom s navrhnutými modelmi na mnou vytvorených dátových sadách, menovite IMPACT a Lidové Noviny. Cieľom experimentov je zistiť, ktorý z modelov a s akými parametrami dosahuje najlepšie výsledky na zadefinovaných metrikách zo sekcie 2.7.

6.1 Trénovanie modelov

Trénovanie modelov som prevádzal na svojom lokálnom zariadení, bez podpory výkonnejšej grafickej karty, a preto som sa snažil aj samotné parametre modelov nastavovať s ohľadom na tento fakt. Každopádne, výsledné modely aj vďaka použitiu iba pozičných príznakov nie sú veľmi veľké (maximálne v jednotkách MB), a preto nebol problém previesť väčší počet experimentov. Experimenty som vykonával pre 2 typy modelov, kde prvý typ bol na rekonštrukciu poradia čítania a druhý typ bol multi-task model, ktorý rekonštruuje poradie čítania riadkov, a súčasne týmto riadkom priraďuje ich sémantický význam. Samotný proces trénovania bol však pre oba modely takmer identický, ak nerátame zmeny datasetov a parametrov modelu. V prvom kroku sa vytvoria jednotlivé grafy zo vstupného CSV súboru, pričom sa výsledné grafy serializujú vo formáte `.pt`. Tento krok je opodstatnený, keďže sa tvorí veľké množstvo grafov, tak sa predíde tomu, aby sa museli jednotlivé grafy neustále vytvárať nanovo. Po vytvorení grafov začína trénovanie modelu. Na trénovanie modelov sa používa *CrossEntropy* loss funkcia, pričom v prípade, že sa jedná o multi-task model, tak je výsledná loss súčtom loss pre hranový a uzlový klasifikátor. Ako optimalizačná funkcia bola pri všetkých modeloch zvolená už notorická *Adam* [26]. Štandardne bol zvolený počet epoch na maximálne 200, pričom bolo experimentálne zistené, že žiadny z modelov sa po viac ako 200 epochách nezlepšoval. K tomu bol ešte pridaný algoritmus *Early stopping*, kde ak pri viac ako 20 epochách sa model nezlepšil, tak sa trénovanie ukončilo. Tento krok bol zvolený, aby sa zabránilo silnému pretrénovaniu modelu [3]. Prechod validačným datasetom sa dial po každých 5 epochách.

V prípade, že bol na vstupe dataset Lidové noviny, tak výsledný model bol natrénovaný iba na klasifikáciu hrán, z ktorej sa následne zostavil reading order, keďže tento dataset neobsahoval iný typ anotácií. Pri tomto datasete je graf jeden článok na stránke. Dataset obsahuje celkovo 493 stránok a 1569 článkov. Dataset bol rozdelený na trénovací a validačný v pomere 70/30, a teda 1098 trénovacích a 471 validačných grafov. Pri datasete IMPACT je výstupom natrénovaný multi-task model, keďže má tento dataset anotovanú postupnosť čítania riadkov aj triedy riadkov. Dataset obsahuje celkovo približne 27 tisíc stránok. Tieto



Obr. 6.1: Základné rozhranie nástroja MLFlow na ukladanie modelov a porovnávanie experimentov.

stránky sú ale predtým, než sa vytvoria dané grafy ešte znovu validované, a ak obsahujú jeden riadok, prípadne neobsahujú žiadny riadok, iba obrázky, tak sa tieto stránky ďalej nepropagujú do trénovania modelu. Po tejto validácii je vo výsledku 25614 stránok, kde teda pri datasete IMPACT predstavuje jedna stránka jeden graf. Tieto grafy sú potom znovu rozdelené v pomere 70/30 na trénovacie a validačné. To viedlo ku konečnému rozdeleniu 17929 trénovacích grafov a 7685 validačných.

Všetky experimenty a modely boli ukladané a porovnávané pomocou *open source* nástroja MLFlow[1], ktorý je účelom prispôsobený práve na experimentovanie a monitorovanie *machine learning* modelov. Základné rozhranie tohto nástroja je na obrázku 6.1.

6.2 Analýza výsledkov

Na konečnú evaluáciu bolo použitých niekoľko metrík, ktoré boli detailne popísané v sekcii 2.7. Pri vyhodnotení postupnosti čítania sú použité metriky *Spearman footrule distance* (skr. **SFD**), *Normalizovaný počet výmen* (skr. **NPV**) a *Normalizovaný počet prerušení* (skr. **NPP**). Implementácia metriky Spearman footrule distance bola prevzatá priamo zo zdrojového kódu[2] práce [44]. Pri všetkých metrikách na vyhodnotenie postupnosti čítania je nižšia hodnota lepšia. Na vyhodnotenie sémantiky riadkov je použitá metrika *váhovaného F1* (skr. **wF1**). Zároveň sa pri vyhodnotení sémantiky riadkov počítala aj *Accuracy* pre každú triedu, ktorá však z dôvodu prehľadnosti nebude uvedená v tabuľke 6.1, ale je uvedená ako zoznam v prílohe A pre najlepšie vytvorený multi-task model. Pri metrikách na vyhodnotenie klasifikácie textu je vyššia hodnota lepšia.

V práci som experimentoval s GAT, GCN a GraphSAGE grafovými neurónovými sieťami, pričom som testoval aj vplyv použitia vstupnej MLP siete, predtým ako sa aplikuje samotná backbone GNN sieť. Na základe experimentov som zistil, že najlepšie výsledky

[1] https://mlflow.org
[2] https://github.com/lquirosd/Order_Relation_Operator/blob/master/src/metrics.py

sa dosahovali pri použití batch size o veľkosti 8 pri oboch datasetoch, pričom som testoval veľkosti mocnín čísla 2, od 4 po 32. Počas týchto experimentov som musel hľadať aj vhodný učiaci koeficient α pri použití optimalizačnej funkcie Adam, ktorým je vo výsledku hodnota 5×10^{-4}. Nastavovanie parametrov na trénovanie som zároveň efektívne kombinoval s informáciami z práce [3], ktorá sa detailne venuje voľbe vhodných parametrov pri učení neurónových sietí. Keďže mnou navrhnuté modely používajú na tvorbu grafu k-NN algoritmus, musel som taktiež empiricky nájsť vhodný parameter k na vytvorenie prepojení v grafe, kde jeho hodnota je pri oboch datasetoch $k = 20$. Pri takto nastavenom parametri k dosahovali modely najlepšie výsledky, a zároveň nevytvárali príliš veľké grafy, ktoré by samotný proces trénovania robili časovo a pamäťovo neúnosným.

Tabuľky 6.1 a 6.2 obsahujú porovnanie medzi najlepšími modelmi, ktoré boli vytvorené pre úlohy reading order a multi-task, a pre každý typ siete, s ktorými práca experimentovala. Najlepšie výsledky dosahuje pri riešení reading order a multi-task úlohe grafová neurónová sieť založená na attention mechanizme (GAT), a táto skutočnosť bola pozorovaná počas celej fázy experimentovania. V prípade datasetu IMPACT bol najlepší výsledok dosiahnutý pomocou kombinácie vstupnej siete MLP a backbone siete GAT, kde sieť MLP sa skladala z 2 plne prepojených vrstiev. Dimenzia skrytých vrstiev bola zvolená uniformne pre obe siete 256. Sieť GAT sa skladala z 8 vrstiev a 16 attention hláv. Výstupná MLP sieť sa skladala zo 4 vrstiev. Paradoxne, pri datasete Lidové noviny, kde úlohou bolo iba určenie reading order, bol najlepší výsledok dosiahnutý modelom, ktorý nevyužíva vstupnú MLP sieť. Táto sieť bola zložená zo 4 skrytých vrstiev GAT o veľkosti 128 a 8 attention hláv. Výstupná MLP sieť bola zložená z 2 vrstiev. Pochopiteľne sa pri prevádzaní experimentov kládol veľký dôraz na otestovanie mnohých parametrov sietí, ako boli napríklad veľkosti skrytých vrstiev pre MLP a GNN (od 16 po 512), počet samotných vrstiev v MLP a GNN sieťach, pridávanie batch normalizačných vrstiev do GNN alebo pridávanie *dropout* kroku do GNN. Samotné experimenty viedli k záverom, že pri väčšom datasete IMPACT sa osvedčilo pridanie viac GNN vrstiev a s väčšou veľkosťou skrytých dimenzií, kdežto pri datasete Lidové noviny, ktorý bol podstatne menší, boli lepšie výsledky dosahované s menšími sieťami. Ďalším poznatkom je to, že využitie vstupnej MLP vrstvy dokázalo na datasete IMPACT pozitívne ovplyvniť samotné výsledky modelu, ale ak sa počet vrstiev v MLP zvyšoval, tak sa kvalita modelu znižovala. Naopak, v prípade datasetu Lidové noviny, na ktorom bol vytvorený iba reading order model sa vstupná MLP sieť neosvedčila, a jej použitie znižovalo kvalitu modelu. Podobný trend bol pozorovaný aj pri výstupnej MLP sieti, ktorá slúžila svojou poslednou vrstvou ako klasifikačná sieť. Taktiež pri oboch modeloch platí to, že pridávanie batch normalizačných vrstiev do GNN ovplyvňovalo model pozitívne, ale používanie techniky dropout neprinieslo žiadne viditeľne lepšie výsledky.

Samotné výsledky metrík pre reading order sú interpretované nasledovne. Metrika SFD hovorí o tom, ako často model priradí riadku nesprávne poradie, pričom ak je poradie riadku uložené nesprávne, tak berie do úvahy aj to, ako veľmi vzdialené je jeho poradie od správneho. Metrika NPV hovorí o tom, ako často model nesprávne zoradil celkové poradie riadkov, pričom ak nastane jediná chyba v zoradení, tak sa berie aj celkové poradie ako nesprávne. Túto metriku je vo výsledku vhodné interpretovať ako doplnok k výslednej hodnote, ktorá nám hovorí o tom, ako často model nespraví jedinú chybu v zoradení. Posledná metrika pre reading order je NPP, ktorá nám popisuje, aké množstvo riadkov model vynechá zo svojej celkovej postupnosti čítania.

Samotné predikcie modelov bolo možné aj vizuálne validovať. Obrázky 6.2, 6.3, 6.4 a 6.5 sú porovnania výstupov modelu s ich ground truth na datasete Lidové noviny. V červených obdĺžnikoch sú zdôraznené chyby, ktoré model urobil v postupnosti čítania. Obecne

Tabuľka 6.1: Porovnanie multi-task modelov na datasete IMPACT.

Najlepšie multi-task modely				
Model	*SFD*	*NPV*	*NPP*	*wF1*
GAT+MLP	**0.27**	**0.73**	**0.15**	**0.96**
GCN+MLP	0.32	0.75	0.24	0.95
GraphSAGE+MLP	0.31	0.77	0.20	0.95
GAT	0.28	0.75	0.19	0.95
GCN	0.47	0.92	0.43	0.90
GraphSAGE	0.39	0.84	0.35	0.95

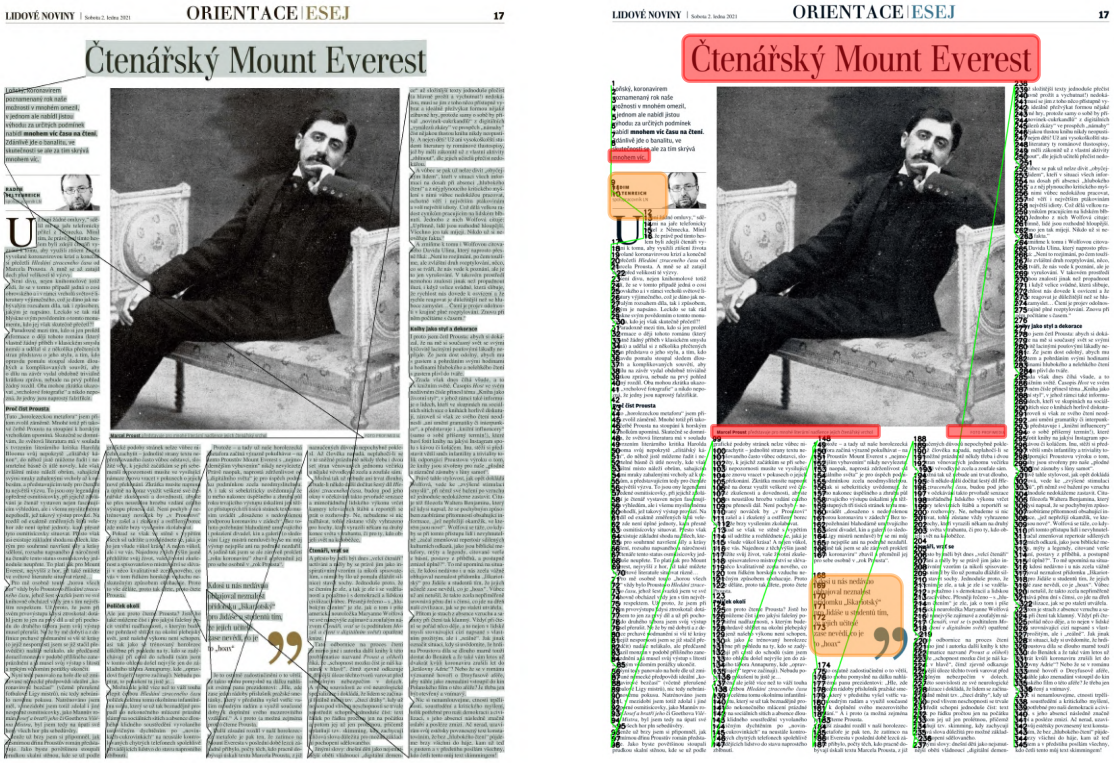
boli chyby modelu najčastejšie pozorované v prípade veľkých nadpisov. Pravdepodobnou príčinou je to, že stránky Lidových novín sa skladali z veľkého množstva riadkov paragrafov, ktoré boli v porovnaní s nadpismi oveľa menšie ako nadpisy a situované blízko pri sebe. Keďže tento model používa iba pozičné príznaky, tak sa naučil veľmi dobre prepojenia medzi geometricky podobnými riadkami, ale nie ak boli príliš odlišné, a teda model nebol schopný dobre generalizovať. Ďalším problémom boli riadky, ktoré boli v ground truth zakódované ako jedny medzi poslednými v postupnosti čítania. Typicky sa jednalo o meno autora článku, popis obrázku, zdroj fotografie alebo citačný úryvok. Tieto chyby sú na obrázkoch zdôraznené oranžovými obdĺžnikmi. Dôvod farebného rozlíšenia týchto chýb je ten, že práve takéto riadky model zaradil do svojej postupnosti čítania v teoreticky správnom poradí, ak by sme sa zamerali na logickú postupnosť, akou by človek čítal text, avšak poskytnutá ground truth brala takéto typy riadkov ako špeciálne prípady a zaraďovala ich medzi posledné v postupnosti čítania. Vo všeobecnosti sa ale vytvorený model vysporiadal veľmi dobre s celkovým prepojením riadkov, ktoré boli rozdelené do viacerých stĺpcov textu a vytvára pomerne rozumné poradie čítania. Obrázky 6.6 a 6.7 porovnávajú podobným spôsobom výstupy z multi-task modelu na datasete IMPACT, s tým že sú pridané aj farebné rozlíšenia predikcií sémantického významu textu. Podobne ako pri datasete Lidové noviny, aj v týchto prípadoch má model z pohľadu určenia poradia riadkov problémy pri nadpisoch. Oproti nadpisom z Lidových novín sú v tomto prípade nadpisy veľkosťou relatívne podobné bežným riadkom paragrafov. Tu je pravdepodobným koreňom problému najmä to, že nadpisy v pôvodných ground truth dátach na úrovni odstavcov nemali priradené poradie čítania, a automatické vytváranie ground truth riadkov, ktoré som implementoval, priraďuje automaticky poradie čítania týmto riadkom ako jedno z posledných. Z pohľadu určovania sémantiky textu sa model chová pomerne stabilne, pričom najčastejšie robí chyby v prípade, že je daný text marginálnej triedy (napríklad poznámka za okrajom, hlavička, začiatočné veľké písmeno) vložený medzi hustý text triedy paragraf. V takýchto prípadoch potom model priraďuje typicky nesprávnu triedu paragraf, ktorá má majoritné zastúpenie v datasete.

Limitácie a možné rozšírenia prístupu

Táto práca experimentovala s viacerými typmi GNN, nastavovaním hyperparametrov modelu, parametrom k pri tvorbe grafu a vytvorením vhodných dátových sád, čo prinieslo niekoľko zaujímavých poznatkov o efektívnosti GNN pri riešení zostavenia reading order a klasifikácie obsahu textu. Na druhej strane je v tejto práci stále veľký priestor pre vylepšenia a zároveň niekoľko limitácií.

Tabuľka 6.2: Porovnanie reading order modelov na datasete Lidové noviny.

Najlepšie reading order modely			
Model	*SFD*	*NPV*	*NPP*
GAT+MLP	0.17	0.74	0.14
GCN+MLP	0.33	0.91	0.31
GraphSAGE+MLP	0.30	0.87	0.27
GAT	**0.15**	**0.67**	**0.12**
GCN	0.31	0.90	0.27
GraphSAGE	0.27	0.84	0.25



Obr. 6.2: Výstup z modelu na určenie poradia riadkov v novinovom článku z datasetu Lidové noviny, kde ground truth je vľavo a predikcie vpravo. Červené obdĺžniky poukazujú na chyby, ktoré model urobil pri postupnosti čítania. Oranžové obdĺžniky poukazujú na chyby, ktoré model urobil voči ground truth zoradeniu, avšak ich chybovosť je diskutabilná.

- Prvým nedostatkom v tejto práci je **konečný výstup modelu**. V ideálnom prípade by bol implementovaný aj krok postprocessingu výstupu, ktorý by výstupy z modelu zakódoval v prípade nového obrázku pri inferencií naspäť do formátu Page XML, alebo by takýto výstup celý vygeneroval. Vytvorenie takéhoto súboru by mohlo byť následne ďalej využité ako nová ground truth, čím by bolo možné generovať kompletne nové datasety, prípadne obohacovať už existujúce, kde by sa vyžadoval už iba krátky čas

Obr. 6.3: Ďalší výstup z modelu na určenie poradia riadkov na novinovom článku z datasetu Lidové noviny, kde ground truth je vľavo a predikcie vpravo. Ako je možné vidieť, výstupné zoradenie je iba pre spodný článok na stránke, a vyplýva to z toho, že bol tento dataset vytvorený po článkoch. Červené obdĺžniky poukazujú na chyby, ktoré model urobil pri postupnosti čítania. V tomto prípade sú chyby pomerne veľké, keďže model vynechal z postupnosti čítania väčšie zoskupenia riadkov, ktoré nasledujú priamo za sebou.

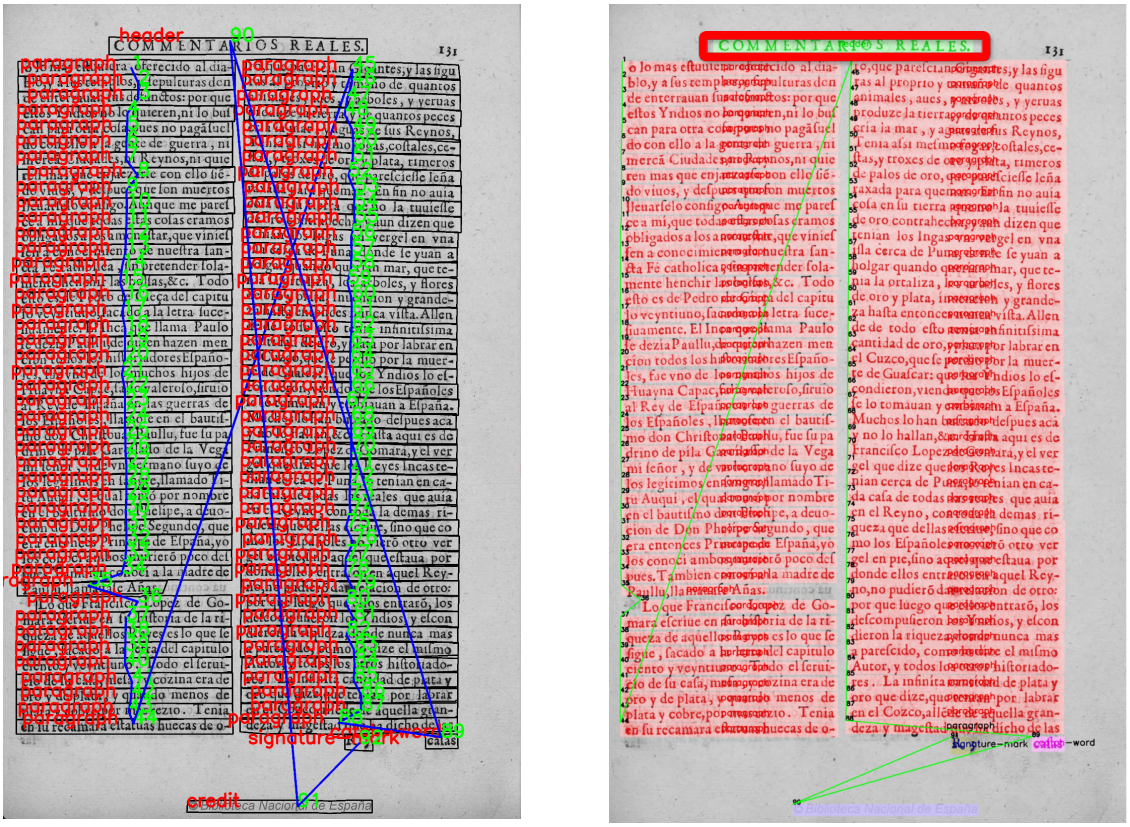
na manuálne korekcie, keďže model je schopný pomerne kvalitne predikovať výsledky. V súčasnom stave táto práca tento krok postprocessingu nezahŕňa a jej výstupy sú ďalej uložené iba vo vhodných dátových štruktúrach jazyku *Python*.

- Ďalším nedostatkom je **reprodukovateľnosť samotných výsledkov**. Keďže práca používa mnou vytvorené datasety Lidové noviny na vyhodnotenie reading order a modifikovaný dataset IMPACT (a to len jeho malú podmnožinu), ktorého granularita sa zmenila z odstavcov na riadky, v súčasnosti neviem o iných verejne dostupných prácach, ktoré by danú problematiku zostavenia poradia čítania pre riadky a určenie sémantiky riadkov riešili na rovnakých datasetoch a rovnakej úrovni. Samozrejme, jednou z možností by bolo pri datasete IMPACT doimplementovať logiku, ktorá by tieto výstupy spätne transformovala na úroveň odstavcov. Avšak riešenie týchto úloh na úrovni odstavcov nie je cieľom tejto práce. S reprodukovateľnosťou výstupov sa zároveň úzko viaže aj **použitie evaluačných metrík**. Keďže dané datasety nemali pevne stanovené porovnávacie metriky, bolo potrebné vyhľadať, a v prípade vyhodnotenia reading order aj vytvoriť nové metriky, ktoré by mi poskytli čo najucelenejší prehľad o tom, ako dobre sa model naučil riešiť dané problémy. Tieto skutočnosti teda značne

Obr. 6.4: Ďalší výstup z reading order modelu na článku z Lidových novín, kde ground truth je vľavo a predikcie vpravo. Červené a oranžové obdĺžniky zastupujú rovnaký význam, ako v predošlých obrázkoch. Typickou chybou modelu je vynechanie väčších nadpisov na stránke.

Obr. 6.5: Výstup z reading order modelu na ďalšom článku Lidových novín, kde ground truth je vľavo a predikcie vpravo. V tomto prípade model zahrnul do postupnosti čítania aj nadpisy, avšak v nesprávnom poradí.

obmedzili všeobecný benchmarking s inými prácami. Na druhej strane táto práca prináša aj nové prístupy na vyhodnotenie, ktoré by mohli slúžiť ako základný stavebný pilier pre nové práce, ktoré by sa rozhodli riešiť danú problematiku podobným spôsobom. Ďalším pozitívom je aj vytvorenie nových dátových sád, kde napríklad v prípade Lidových novín sa jedná o dataset, ktorý je svojou podstatou novinových článkov pomerne obtiažný, ak sa zameriame na zostavenie správnej postupnosti čítania, a preto by mohol slúžiť v budúcnosti ako benchmarking dataset.

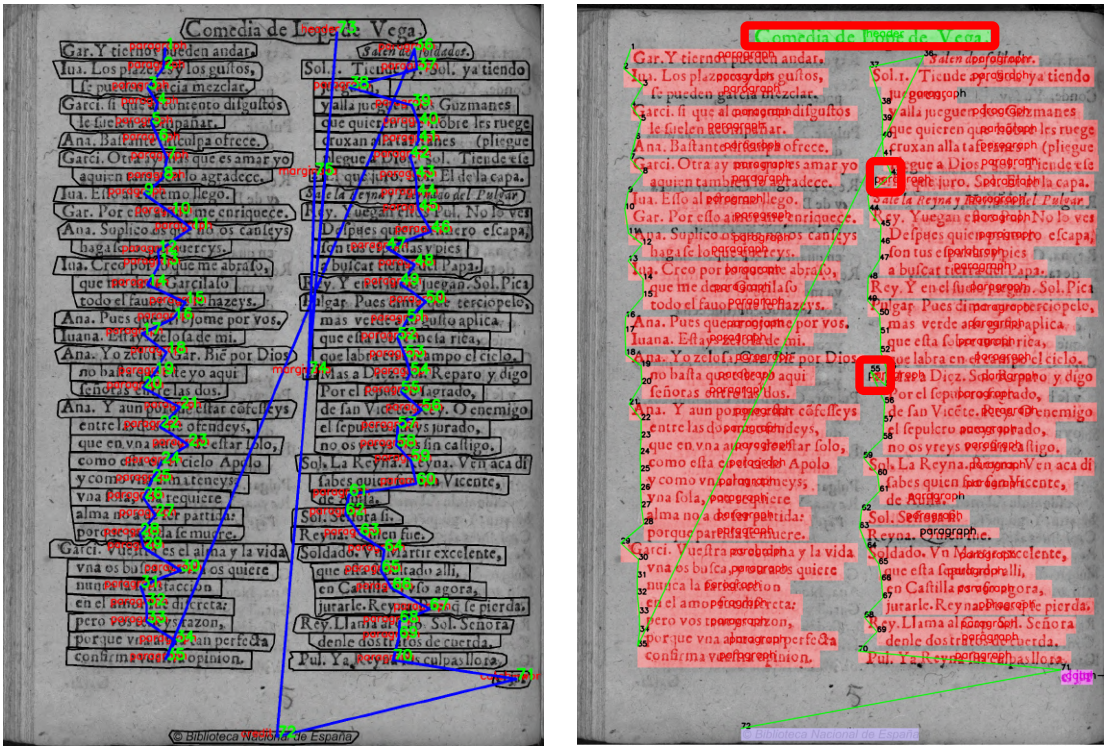


Obr. 6.6: Výstup z multi-task modelu, ktorý zrekonštruoval poradie čítania priradil jednotlivým riadkom ich triedy na stránke knihy z datasetu IMPACT, kde ground truth je vľavo a predikcie vpravo. Červený obdĺžnik poukazuje na jednu chybu, ktorú model z pohľadu postupnosti čítania riadkov urobil. Z pohľadu klasifikácie riadkov bol model schopný priradiť všetkým riadkom ich správnu triedu.

- Ako už bolo predstavené v úvodnej sekcií 2.5, v práci som analyzoval **niekoľko rôznych typov úloh**, ktoré by boli vhodne reprezentovateľné grafom a riešené pomocou GNN. Úloha spájania, resp. rozdeľovania riadkov, ktoré sú na výstupe z OCR sa javí ako veľmi užitočná na zvýšenie robustnosti riešenia. Keďže model je v súčasnom stave závislý od kvality výstupu z OCR, nie je schopný určiť či sa napríklad nejedná o ten istý riadok. Na obrázku 6.8 je možné vidieť, že pôvodný výstup z OCR rozdelil jeden riadok (v tomto prípade nadpis) na dva separátne riadky, a následne vytvorený model túto informáciu ďalej propagoval.

- Ďalším predmetom záujmu v nadväznosti na túto prácu by mohlo byť **experimentovanie s viacerými typmi GNN**, prípadne vytvorenie novej siete, ktorá by bola schopná lepšie propagovať vstupné informácie do modelu. V tejto práci som experimentoval iba s tromi typmi GNN sietí, pričom PyTorch Geometric [15] poskytuje oficiálne takmer 60 rôznych typov sietí. Tento krok by pochopiteľne musel zahŕňať obšírnejšie bádanie a hľadanie čo najlepšej varianty.

- **Limitácia tvorbou grafu.** V práci bol použitý jednoduchý algoritmus k-NN, kde body, medzi ktorými sa počítala vzdialenosť boli stredy bounding boxov riadkov.



Obr. 6.7: Výstup z multi-task modelu na stránke z datasetu IMPACT, kde ground truth je vľavo a predikcie vpravo. Červené obdĺžniky poukazujú na chyby, ktoré model urobil z pohľadu postupnosti čítania. Spodné dva obdĺžniky poukazujú okrem chýb v postupnosti čítania aj na nesprávnu klasifikáciu riadkov, kde v tomto prípade im model priradil triedu paragraf, pričom sa v skutočnosti jednalo o marginálie (poznámky za okrajom).

Obr. 6.8: Výstup z modelu, ktorý propagoval nesprávny výstup z OCR aj ďalej.

Aj keď sa vo výsledku vytvorený prístup vysporiadal s obmedzenosťou tohto algoritmu, zamedzilo to v istom smere využitiu plného potenciálu GNN. Ak uvážime prípady rozloženia stránok, ktoré sa skladajú z niekoľkých stĺpcov textu, ako napríklad noviny, alebo historické texty, kde štruktúra textu nebola vždy pevne daná, algoritmus k-NN zlyhával v zachytení dôležitých prepojení medzi riadkami, ktoré boli od seba geometricky príliš vzdialené, ale z logickej postupnosti čítania riadkov mali byť prepojené. Toto vo výsledku viedlo k tomu, že sa z výstupu siete získalo niekoľko spojených komponent (podgrafov), ktoré bolo nakoniec potrebné znovu spojiť do jedného celku, k čomu slúžil mnou vytvorený algoritmus Line precedence, ktorý bol ale zalo-

žený na pravidlách. Vhodná tvorba grafu je teda v tomto prípade kritickou časťou, a v súčasnosti nepoznám odpoveď na ideálne riešenie. Každopádne sa ale naskytuje niekoľko rôznych iných typov tvorby grafu, ktoré boli použité v prácach na analýzu rozloženia pomocou GNN, ako je napríklad graf viditeľnosti, popísaný v práci [43], alebo β-skeleton graf [51, 50, 28], a určite by bolo vhodné previesť experimenty aj s týmito algoritmami.

- **Extrakcia príznakov vstupujúcich do modelu**. Príznaky, ktoré sú v tejto práci použité, sú čisto pozičné súradnice bounding boxov, ktoré sú získané z výstupu OCR, a k nim sú ešte pridané príznaky *dĺžka riadku, výška riadku* a *súradnice stredu riadku*. Aj keď sa v prácach [51, 50] zámerne opierajú podobne iba o pozičné príznaky segmentov textu a argumentujú veľkosťou modelu a jeho nezávislosťou od jazyka, v prípade klasifikácie sémantiky textu sú tieto príznaky nepostačujúce. Ako už bolo uvedené vo vyhodnotení experimentov, GNN model dosahuje obstojné výsledky pri klasifikácii textu, ale na druhej strane je potrebné počítať s tým, že dataset IMPACT má veľkú prevahu triedy *paragraf*. Práve táto trieda častokrát ovplyvňuje minoritnejšie triedy, kde napríklad nadpisy sú zamenené za paragrafy. Z toho dôvodu by mohlo byť vhodné experimentovať aj s jazykovými features, ktoré by vnášali informáciu aj o sémantike textu. Príznaky tohto typu by mohli byť extrahované napríklad pomocou transformerového modelu BERT [12], kde výstupné príznaky by sa zobrali z jej poslednej skrytej vrstvy pred klasifikačnou. Keďže mnou vytvorené datasety obsahujú aj textový prepis riadkov, je to jedna z obohacujúcich informácií pre model. Typickým problémom v jazykových modeloch ale býva, že ich výstupné dimenzie sú veľmi veľké [22], a je potrebné ich pre zachovanie kompaktnosti modelu zredukovať. Ďalšou časťou príznakov, ktoré používam, sú príznaky pre hrany, kde každá hrana nesie informáciu o vzdialenosti medzi dvomi riadkami, a teda znovu sa jedná a výsledok z pozičných príznakov. Jedna z možností, ako obohatiť hranové príznaky bola predstavená v práci [50], kde používajú predtrénovaný *MobileNetV3-Small* [21] model, založený na konvolučných neurónových sieťach. Tento model používajú na výrezoch z obrázku, kde jeden výrez je bounding box dvoch riadkov, ktoré spája hrana, pričom tieto príznaky následne zakódujú do hrán. Samozrejme, existuje množstvo iných variant, akými by sa dal model takto obohatiť.

- **Vytvorenie ground truth datasetov**. Keďže pre oba datasety bola ground truth generovaná automaticky, tieto datasety nie sú svojou presnosťou na úrovni ručných anotácií, najmä ak sa zameriame na postupnosť čítania. V prípade datasetu IMPACT, ako už bolo spomenuté v sekcii 4.2, sú častokrát niektoré pôvodné odstavce na stránke zaradené do skupiny `Unordered`, a teda nevieme v akom poradí by sa mali dané odstavce čítať. Táto skutočnosť sa vo výsledku odráža aj v samotných anotáciách na úrovni riadkov, a teda častokrát nastane prípad, že nadpis na stránke je v poradí ako posledný, resp. jeden medzi poslednými. Tento fakt bohužiaľ ovplyvňuje do istej miery aj samotnú kvalitu modelu, keďže sa pri vytváraní ground truth grafov nevytvárajú pozitívne hrany medzi nadpisom a prvým riadkom v nasledujúcom odstavci. Príklad takéhoto výstupu je na obrázku 6.4, kde ground truth pre nadpis je ako jeden z posledných riadkov, a teda aj samotná sieť sa na takýchto typoch stránok naučila, že prepojenie medzi prvými dvomi riadkami (nadpis a prvý riadok z paragrafu) nemá existovať. Je to jedna z limitácií v tejto práci, ktorú by v budúcnosti bolo potrebné vyriešiť. Podobne pri datasete Lidové noviny sa vytváranie ground truth opieralo o dodané textové súbory, ktoré boli spísané sekvenčne v poradí, v akom sa mal čítať daný

článok. Aj v tomto prípade je však možné vidieť, že z logického poradia čítania neboli tieto ground truth texty vytvorené vždy správne. Obrázok 6.2 vizualizuje ground truth článku a predikciu modelu, pričom chybné zoradenia sú označené v červených, resp. oranžových blokoch. Práve oranžové bloky poukazujú na diskutabilné posudzovanie výsledkov, keďže model z logickej postupnosti čítania textu zhora dolu a zľava doprava zoradil riadky v správnom poradí, no naopak ground truth postupnosť týchto riadkov udáva až na svojom konci. Je to z toho dôvodu, že dodané textové súbory uvažovali o segmentoch, ako je meno autora článku, popis obrázku, zdroj fotografie alebo citačný úryvok, ako o posledných častiach, ktoré by si čitateľ prečítal. Z tohto hľadiska je teda otázne, čo je správne poradie, a je to hodné istej polemiky. Súčasne je nedostatkom datasetu to, že bola ground truth textu dodaná iba pre jednotlivé články na stránke, nie však celkové poradie článkov na stránke, čím bolo možné vytvoriť iba dataset, kde vstupom do modelu je článok na stránke, nie celá stránka.

Kapitola 7

Záver

Cieľom tejto práce bolo vytvoriť prístup na analýzu rozloženia stránok z pohľadu rekonštrukcie poradia čítania a určenia sémantického významu textu. Z pohľadu typov dokumentov sa jednalo o novinové články a ručne písané historické stránky. Obe z týchto úloh sa riešili na úrovni riadkov. Na vytvorenie tohto prístupu boli použité grafové neurónové siete. Predtým, ako sa mohlo začať s budovaním samotného riešenia, bolo potrebné získať vhodné dátové sady, ktoré by obsahovali správne typy anotácií a obsahovo by spĺňali stanovené podmienky. To viedlo k vytvoreniu dvoch nových dátových sád, menovite Lidové noviny a IMPACT, kde prvá spomenutá sada sa skladá z anotovanej postupnosti čítania riadkov v novinových článkoch a druhá dátová sada obsahuje anotácie postupnosti čítania riadkov a ich sémantický význam pre ručne písané historické spisy. Dataset Lidových novín bol vytvorený od úplných základov, čo zahŕňalo použitie OCR nástroja PeroOCR na získanie prepisov a pozícií riadkov, a automatizovanú generáciu ground truth poradia čítania pomocou dodaných textových súborov. Dataset IMPACT už mal dostupné OCR výstupy pre riadky aj odstavce, pričom anotácie pre reading order a triedy textu boli na úrovni odstavcov. Z toho dôvodu bol znovu vytvorený automatický prístup, ktorý tieto anotácie pretransformoval z odstavcov na riadky.

Súčasťou riešenia je aj experimentovanie s viacerými typmi GNN, pričom sa práca sústredila na otestovanie vhodnosti GCN, GAT a GraphSAGE grafových neurónových sietí. Z toho dôvodu bola vytvorená architektúra, ktorá by efektívne umožňovala experimentovanie s viacerými typmi sietí a datasetov. Implementácia riešenia sa delí na tri kroky: tvorba grafov, klasifikácia uzlov a hrán grafov, a v prípade určenia postupnosti čítania riadkov aj samotnej rekonštrukcie postupnosti čítania. Tvorba grafu prebiehala na štruktúrovane uložených dátach pomocou metódy k-NN. V kroku klasifikácie uzlov a hrán sú tieto výstupy získané pomocou GNN modelu, kde klasifikované uzly sa priamo používajú na určenie sémantického významu riadkov v prípade datasetu IMPACT a klasifikované hrany sú ďalej použité v treťom kroku rekonštrukcie globálneho poradia čítania. Tento krok zahŕňal vytvorenie algoritmu, ktorý pospája pozitívne predikcie hrán do spojených komponent (podgrafov), a následne zoradenia týchto podgrafov.

V práci bolo vykonaných niekoľko experimentov na nájdenie a vyhodnotenie najlepších modelov. Pre dataset Lidové noviny bol vytvorený model, ktorý je určený výhradne na rekonštrukciu poradia čítania, kde tento model bol založený na GAT grafovej neurónovej sieti. Pri datasete IMPACT bol vytvorený multi-task model, ktorý je schopný rekonštruovať poradie čítania, a zároveň určiť sémantický význam riadkov. Podobne, ako pre Lidové noviny, aj tu je model založený na GAT grafovej neurónovej sieti.

Na vyhodnotenie úspešnosti modelov sa práca zamerala na dva typy metrík, a to pre vyhodnotenie poradia čítania a vyhodnotenie klasifikácie textu. Na vyhodnotenie postupnosti čítania boli použité metriky Spearman footrule distance (SFD), normalizovaný počet výmen (NPV) a normalizovaný počet prerušení (NPP), kde posledné dve boli vymyslené a špecificky implementované v tejto práci za účelom získania objektívneho pohľadu o kvalite modelu. Na vyhodnotenie klasifikácie textu bola použitá metrika váženého F1 (wF1) skóre. Model určený iba na klasifikáciu poradia čítania dosahuje hodnoty 0.15 pre SFD, 0.67 pre NPV a 0.12 pre NPP. Multi-task model dosahuje hodnoty 0.27 pre SFD, 0.73 pre NPV, 0.15 pre NPP a 0.96 pre wF1.

V tejto práci existuje stále veľký priestor pre viaceré vylepšenia, ako sú napríklad vhodnejšia tvorba grafu, experimentovanie s inými typmi GNN sietí, alebo obohatenie vstupných príznakov. Avšak zásadným nedostatkom tejto práce je to, že vytvorené modely nie sú objektívne porovnané s už existujúcimi riešeniami, čo znižuje kredibilitu samotného riešenia, a vo veľkej miere to ovplyvnila aj voľba dátových sád. Aj z tohto dôvodu boli do práce implementované vlastné metriky, ktoré by mi poskytli viac dôveryhodnosti vo vytvorené modely. Napriek tomu vytvorené modely preukazujú sľubné signály pri riešení postupnosti čítania riadkov, ako aj pri priraďovaní sémantického významu textu, čím práca poskytuje dobrý základný pilier pre prípadné budúce rozšírenia.

Literatúra

[1] Aggarwal, C., He, G. a Zhao, P. Edge classification in networks. In: *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. 2016, s. 1038–1049. DOI: 10.1109/ICDE.2016.7498311.

[2] Aggarwal, C. C. *Data Mining*. Springer International Publishing, 2015. Dostupné z: http://dx.doi.org/10.1007/978-3-319-14142-8.

[3] Bengio, Y. Practical recommendations for gradient-based training of deep architectures. *CoRR*. 2012, abs/1206.5533. Dostupné z: http://arxiv.org/abs/1206.5533.

[4] Bieniecki, W., Grabowski, S. a Rozenberg, W. Image Preprocessing for Improving OCR Accuracy. In: *2007 International Conference on Perspective Technologies and Methods in MEMS Design*. 2007, s. 75–80. DOI: 10.1109/MEMSTECH.2007.4283429.

[5] Breuel, T. M. High Performance Document Layout Analysis. In:. 2003. Dostupné z: https://api.semanticscholar.org/CorpusID:2655325.

[6] Carbonell, M., Riba, P., Villegas, M., Fornés, A. a Lladós, J. Named Entity Recognition and Relation Extraction with Graph Neural Networks in Semi Structured Documents. In: *2020 25th International Conference on Pattern Recognition (ICPR)*. 2021, s. 9622–9627. DOI: 10.1109/ICPR48806.2021.9412669.

[7] Chaudhuri, B. B. Digital Document Processing. In:. Springer London, 2007. DOI: 1-84628-726-8. ISBN 978-1-84628-726-8.

[8] Clausner, C., Pletschacher, S. a Antonacopoulos, A. The Significance of Reading Order in Document Recognition and Its Evaluation. In: *2013 12th International Conference on Document Analysis and Recognition*. 2013, s. 688–692. DOI: 10.1109/ICDAR.2013.141.

[9] Clausner, C., Papadopoulos, C., Pletschacher, S. a Antonacopoulos, A. The ENP image and ground truth dataset of historical newspapers. In: *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. 2015, s. 931–935. DOI: 10.1109/ICDAR.2015.7333898.

[10] Clinchant, S., Déjean, H., Meunier, J., Lang, E. M. a Kleber, F. Comparing Machine Learning Approaches for Table Recognition in Historical Register Books. *CoRR*. 2019, abs/1906.11901. Dostupné z: http://arxiv.org/abs/1906.11901.

[11] David Doermann, K. T. Handbook of Document Image Processing and Recognition. In:. Springer London, 2014. DOI: 10.1007/978-0-85729-859-1. ISBN 978-0-85729-858-4.

[12] Devlin, J., Chang, M., Lee, K. a Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR*. 2018, abs/1810.04805. Dostupné z: http://arxiv.org/abs/1810.04805.

[13] Diem, M., Kleber, F., Sablatnig, R. a Gatos, B. CBAD: ICDAR2019 Competition on Baseline Detection. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. 2019, s. 1494–1498. DOI: 10.1109/ICDAR.2019.00240.

[14] Ferludin, O., Eigenwillig, A., Blais, M., Zelle, D., Pfeifer, J. et al. *TF-GNN: Graph Neural Networks in TensorFlow*. 2023.

[15] Fey, M. a Lenssen, J. E. Fast Graph Representation Learning with PyTorch Geometric. In: *ICLR Workshop on Representation Learning on Graphs and Manifolds*. 2019.

[16] Gemelli, A., Biswas, S., Civitelli, E., Lladós, J. a Marinai, S. Doc2Graph: A Task Agnostic Document Understanding Framework Based on Graph Neural Networks. In: *Computer Vision – ECCV 2022 Workshops*. Springer Nature Switzerland, 2023, s. 329–344. DOI: 10.1007/978-3-031-25069-9_22. ISBN 9783031250699. Dostupné z: http://dx.doi.org/10.1007/978-3-031-25069-9_22.

[17] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O. a Dahl, G. E. Neural Message Passing for Quantum Chemistry. *CoRR*. 2017, abs/1704.01212. Dostupné z: http://arxiv.org/abs/1704.01212.

[18] Grandini, M., Bagli, E. a Visani, G. *Metrics for Multi-Class Classification: an Overview*. 2020.

[19] Hamilton, W. L., Ying, R. a Leskovec, J. Inductive Representation Learning on Large Graphs. *CoRR*. 2017, abs/1706.02216. Dostupné z: http://arxiv.org/abs/1706.02216.

[20] Hochreiter, S. a Schmidhuber, J. Long Short-term Memory. *Neural computation*. December 1997, zv. 9, s. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[21] Howard, A., Sandler, M., Chen, B., Wang, W., Chen, L.-C. et al. Searching for MobileNetV3. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, s. 1314–1324. DOI: 10.1109/ICCV.2019.00140.

[22] Huertas García, Á., Martín, A., Huertas Tato, J. a Camacho, D. Exploring Dimensionality Reduction Techniques in Multilingual Transformers. *Cognitive Computation*. Mar 2023, zv. 15, č. 2, s. 590–612. DOI: 10.1007/s12559-022-10066-8. ISSN 1866-9964. Dostupné z: https://doi.org/10.1007/s12559-022-10066-8.

[23] Islam, N., Islam, Z. a Noor, N. A Survey on Optical Character Recognition System. *CoRR*. 2017, abs/1710.05703. Dostupné z: http://arxiv.org/abs/1710.05703.

[24] Kanakaris, N., Karacapilidis, N. a Kournetas, G. On the Exploitation of Textual Descriptions for a Better-informed Task Assignment Process. In:. Január 2020, s. 304–310. DOI: 10.5220/0009151603040310.

[25] Kendall, A., Gal, Y. a Cipolla, R. Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. *CoRR*. 2017, abs/1705.07115. Dostupné z: http://arxiv.org/abs/1705.07115.

[26] Kingma, D. P. a Ba, J. *Adam: A Method for Stochastic Optimization*. 2017.

[27] Kipf, T. N. a Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. *CoRR*. 2016, abs/1609.02907. Dostupné z: http://arxiv.org/abs/1609.02907.

[28] Kirkpatrick, D. G. a Radke, J. D. A Framework for Computational Morphology. In: TOUSSAINT, G. T., ed. *Computational Geometry*. North-Holland, 1985, sv. 2, s. 217–248. Machine Intelligence and Pattern Recognition. DOI: https://doi.org/10.1016/B978-0-444-87806-9.50013-X. ISSN 0923-0459. Dostupné z: https://www.sciencedirect.com/science/article/pii/B978044487806950013X.

[29] Kišš, M., Beneš, K. a Hradiš, M. AT-ST: Self-training Adaptation Strategy for OCR in Domains with Limited Transcriptions. In: Lladós, J., Lopresti, D. a Uchida, S., ed. *Document Analysis and Recognition – ICDAR 2021*. Cham: Springer International Publishing, 2021, s. 463–477. ISBN 978-3-030-86337-1.

[30] Kitaev, N. a Klein, D. *Constituency Parsing with a Self-Attentive Encoder*. arXiv, 2018. DOI: 10.48550/ARXIV.1805.01052. Dostupné z: https://arxiv.org/abs/1805.01052.

[31] Kodym, O. a Hradis, M. Page Layout Analysis System for Unconstrained Historic Documents. *CoRR*. 2021, abs/2102.11838. Dostupné z: https://arxiv.org/abs/2102.11838.

[32] Kohút, J. a Hradis, M. TS-Net: OCR Trained to Switch Between Text Transcription Styles. *CoRR*. 2021, abs/2103.05489. Dostupné z: https://arxiv.org/abs/2103.05489.

[33] Kumar, R. a Vassilvitskii, S. Generalized distances between rankings. In: *Proceedings of the 19th International Conference on World Wide Web*. New York, NY, USA: Association for Computing Machinery, 2010, s. 571–580. WWW '10. DOI: 10.1145/1772690.1772749. ISBN 9781605587998. Dostupné z: https://doi.org/10.1145/1772690.1772749.

[34] Li, L., Gao, F., Bu, J., Wang, Y., Yu, Z. et al. An End-to-End OCR Text Re-organization Sequence Learning for Rich-Text Detail Image Comprehension. In: Vedaldi, A., Bischof, H., Brox, T. a Frahm, J.-M., ed. *Computer Vision – ECCV 2020*. Cham: Springer International Publishing, 2020, s. 85–100. ISBN 978-3-030-58595-2.

[35] Li, M., Xu, Y., Cui, L., Huang, S., Wei, F. et al. DocBank: A Benchmark Dataset for Document Layout Analysis. *CoRR*. 2020, abs/2006.01038. Dostupné z: https://arxiv.org/abs/2006.01038.

[36] Luo, S., Ding, Y., Long, S., Poon, J. a Han, S. C. *Doc-GCN: Heterogeneous Graph Convolutional Networks for Document Layout Analysis*. arXiv, 2022. DOI: 10.48550/ARXIV.2208.10970. Dostupné z: https://arxiv.org/abs/2208.10970.

[37] Namboodiri, A. M. a Jain, A. K. Document Structure and Layout Analysis. In: Chaudhuri, B. B., ed. *Digital Document Processing: Major Directions and Recent Advances*. London: Springer London, 2007, s. 29–48. DOI: 10.1007/978-1-84628-726-8_2. ISBN 978-1-84628-726-8. Dostupné z: https://doi.org/10.1007/978-1-84628-726-8_2.

[38] Pacal, I., Karaboga, D., Basturk, A., Akay, B. a Nalbantoglu, U. A comprehensive review of deep learning in colon cancer. *Computers in Biology and Medicine*. 2020, zv. 126, s. 104003. DOI: https://doi.org/10.1016/j.compbiomed.2020.104003. ISSN 0010-4825. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0010482520303346.

[39] Papadopoulos, C., Pletschacher, S., Clausner, C. a Antonacopoulos, A. The IMPACT Dataset of Historical Document Images. In: *Proceedings of the 2nd International Workshop on Historical Document Imaging and Processing*. New York, NY, USA: Association for Computing Machinery, 2013, s. 123–130. HIP '13. DOI: 10.1145/2501115.2501130. ISBN 9781450321150. Dostupné z: https://doi.org/10.1145/2501115.2501130.

[40] Pfitzmann, B., Auer, C., Dolfi, M., Nassar, A. S. a Staar, P. DocLayNet: A Large Human-Annotated Dataset for Document-Layout Segmentation. In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, Aug 2022. DOI: 10.1145/3534678.3539043. Dostupné z: https://doi.org/10.1145%2F3534678.3539043.

[41] Pletschacher, S. a Antonacopoulos, A. The PAGE (Page Analysis and Ground-Truth Elements) Format Framework. In: *2010 20th International Conference on Pattern Recognition*. 2010, s. 257–260. DOI: 10.1109/ICPR.2010.72.

[42] Prasad, A., Déjean, H. a Meunier, J.-L. Versatile Layout Understanding via Conjugate Graph. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. 2019, s. 287–294. DOI: 10.1109/ICDAR.2019.00054.

[43] Prieto, J. R. a Vidal, E. Improved Graph Methods for Table Layout Understanding. In: Lladós, J., Lopresti, D. a Uchida, S., ed. *Document Analysis and Recognition – ICDAR 2021*. Cham: Springer International Publishing, 2021, s. 507–522. ISBN 978-3-030-86331-9.

[44] Quirós, L. a Vidal, E. Reading order detection on handwritten documents. *Neural Computing and Applications*. 2022, zv. 34, č. 12, s. 9593–9611. DOI: 10.1007/s00521-022-06948-5.

[45] 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.

[46] Singh, A. a Pandey, B. An euclidean distance based KNN computational method for assessing degree of liver damage. In: *2016 International Conference on Inventive*

Computation Technologies (ICICT). 2016, sv. 1, s. 1–4. DOI: 10.1109/INVENTIVE.2016.7823222.

[47] Srihari, S. N., Shekhawat, A. a Lam, S. W. Optical character recognition (OCR). In: *Encyclopedia of Computer Science*. GBR: John Wiley and Sons Ltd., 2003, s. 1326–1333. ISBN 0470864125.

[48] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P. et al. *Graph Attention Networks*. 2018.

[49] Wang, J., Krumdick, M., Tong, B., Halim, H., Sokolov, M. et al. A Graphical Approach to Document Layout Analysis. In: Fink, G. A., Jain, R., Kise, K. a Zanibbi, R., ed. *Document Analysis and Recognition - ICDAR 2023*. Cham: Springer Nature Switzerland, 2023, s. 53–69. ISBN 978-3-031-41734-4.

[50] Wang, R., Fujii, Y. a Bissacco, A. *Text Reading Order in Uncontrolled Conditions by Sparse Graph Segmentation*. 2023.

[51] Wang, R., Fujii, Y. a Popat, A. C. Post-OCR Paragraph Recognition by Graph Convolutional Networks. In: *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2022, s. 2533–2542. DOI: 10.1109/WACV51458.2022.00259.

[52] Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M. et al. Dynamic Graph CNN for Learning on Point Clouds. *ACM Trans. Graph.* New York, NY, USA: Association for Computing Machinery. oct 2019, zv. 38, č. 5. DOI: 10.1145/3326362. ISSN 0730-0301. Dostupné z: https://doi.org/10.1145/3326362.

[53] Wang, Z., Xu, Y., Cui, L., Shang, J. a Wei, F. LayoutReader: Pre-training of Text and Layout for Reading Order Detection. *CoRR*. 2021, abs/2108.11591. Dostupné z: https://arxiv.org/abs/2108.11591.

[54] Wei, S. a Xu, N. *PARAGRAPH2GRAPH: A GNN-based framework for layout paragraph analysis*. 2023.

[55] Wikipedia contributors. *BLEU — Wikipedia, The Free Encyclopedia*. 2024. [Online; accessed 5-May-2024]. Dostupné z: https://en.wikipedia.org/w/index.php?title=BLEU&oldid=1221925142.

[56] Wikipedia contributors. *Nearest neighbor graph — Wikipedia, The Free Encyclopedia*. 2024. [Online; accessed 3-May-2024]. Dostupné z: https://en.wikipedia.org/w/index.php?title=Nearest_neighbor_graph&oldid=1217132532.

[57] Wu, L., Cui, P., Pei, J. a Zhao, L. Graph Neural Networks: Foundations, Frontiers, and Applications. In:. Springer Singapore, January 2022. DOI: 981-16-6054-2. ISBN 978-981-16-6054-2.

[58] Yehudai, G., Fetaya, E., Meirom, E. A., Chechik, G. a Maron, H. On Size Generalization in Graph Neural Networks. *CoRR*. 2020, abs/2010.08853. Dostupné z: https://arxiv.org/abs/2010.08853.

[59] Zbyněk Křivka, T. M. *Grafové algoritmy*. 2018. Dostupné z: https://www.fit.vutbr.cz/study/courses/GAL/public/gal-slides.pdf.

[60] Zhao, W.-L., Wang, H. a Ngo, C.-W. Approximate k-NN Graph Construction: A Generic Online Approach. *IEEE Transactions on Multimedia*. 2022, zv. 24, s. 1909–1921. DOI: 10.1109/TMM.2021.3073811.

[61] Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C. et al. Graph neural networks: A review of methods and applications. *AI Open*. 2020, zv. 1, s. 57–81. DOI: https://doi.org/10.1016/j.aiopen.2021.01.001. ISSN 2666-6510. Dostupné z: https://www.sciencedirect.com/science/article/pii/S2666651021000012.

Príloha A

Celkový prehľad experimentov

Táto príloha obsahuje zoznam, ktorý prezentuje jednotlivé výstupy metriky *Accuracy* pre každú triedu (sémantického významu textu), ktorá bola v datasete IMPACT. Zoznam obsahuje iba hodnoty z najlepšie vytvoreného multi-task modelu založeného na GAT grafovej neurónovej sieti so vstupnou MLP sieťou.

- Paragraf: 0.962
- Nadpis: 0.983
- Hlavička: 0.997
- Pätička: 1.0
- Číslo strany: 0.998
- Catch-word: 0.997
- Marginália: 0.998
- Kredit: 0.999
- Iniciály: 0.995
- Obsah: 0.994
- Poznámka pod čiarou: 0.995
- Podpis: 0.998
- Pokračovanie poznámky pod čiarou: 0.999
- Titulok: 1.0
- Nepriradený text: 1.0

Príloha B

Obsah priloženej SD karty

- **program**
  - obsahuje všetky zdrojové kódy a vstupné dáta, avšak samotné originálne dokumenty boli pridané iba pre dataset Lidové noviny, keďže dataset IMPACT veľkosťou presahuje 60GB.
- **text**
  - LaTeX zdrojové súbory.
  - LaTeX vygenerovaná práca v PDF formáte.
- **video**
  - video prezentujúce túto prácu