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

AUTOMATIZOVANÁ EXTRAKCE STRUKTUROVANÝCH DAT DOKUMENTŮ

AUTOMATED METADATA EXTRACTION FROM DOCUMENT IMAGES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **JAKUB KŘIVÁNEK**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. JAN KOHÚT**
SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání bakalářské práce

155520

Ústav:	Ústav počítačové grafiky a multimédií (UPGM)
Student:	**Křivánek Jakub**
Program:	Informační technologie
Název:	**Automatizovaná extrakce strukturovaných dat dokumentů**
Kategorie:	Zpracování obrazu
Akademický rok:	2023/24

Zadání:

1. Nastudujte základy neuronových sítí pro detekci řádků textu a rozpoznávání textu.
2. Vytvořte si přehled o současných přístupech pro extrakci strukturovaných dat dokumentů.
3. Navrhněte vhodnou metodu pro extrakci strukturovaných dat dokumentů.
4. Vytvořte vhodnou datovou sadu nebo rozšiřte některou z existujících datových sad.
5. Vyhodnoťte navrženou metodu na datové sadě.
6. Zhodnoťte dosažené výsledky.
7. Vytvořte krátké video prezentující výsledky vaši práce.

Literatura:

- Shi, B., Bai, X. and Yao, C., 2016. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, *39* (11), pp.2298-2304.
- Wick, C., Zöllner, J. and Grüning, T., 2021, September. Transformer for handwritten text recognition using bidirectional post-decoding. In *International Conference on Document Analysis and Recognition* (pp. 112-126). Cham: Springer International Publishing.
- Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).

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

Vedoucí práce:	**Kohút Jan, Ing.**
Vedoucí ústavu:	Černocký Jan, prof. Dr. Ing.
Datum zadání:	1.11.2023
Termín pro odevzdání:	9.5.2024
Datum schválení:	9.11.2023

Abstrakt

Tato bakalářská práce řeší problém získávání strukturovaných dat ze skenů dokumentů českých knihoven. Cílem práce je usnadnit časově náročný manuální proces knihovníkům. Zaměřil jsem se vytvoření datových sad z dokumentů českých knihoven a na detekci metadat na těchto datasetech. Datové sady jsem vytvořil pro knihy a druhou pro periodika. Detekce byla realizována způsoben klasifikace řádků přečtených z dokumentů. Pro to jsou použita plně propojená neuronoví síť a síť využívající Transformer Encoder. Druhý způsob detekce metadat je založen na detekci objektů na skenech dokumentů pomocí modelu YOLOv8. Detekce pomocí plně propojené neuronové sítě dosahuje F1 skóre 0,83 na datasetu knih a 0,78 na datasetu periodik. F1 skóre sítě s Transformer Encoder dosahuje hodnot 0,84 na datasetu knih a 0,59 na datasetu periodik. Model YOLO dosahuje F1 skóre 0,86 (confidece na 0,549) na datasetu knih a 0,7 (confidence na 0,336) na datasetu periodik.

Abstract

This Bachelor thesis addresses the problem of extracting structured data from scans of documents from Czech libraries. The aim of the thesis is to simplify the time-consuming manual process for librarians. I focused on creating datasets from documents of Czech libraries and on detecting metadata on these datasets. I created one dataset for books and another for periodicals. Detection was performed by classifying lines read from the documents. This utilized a fully connected neural network and a network employing a Transformer Encoder. The second method of metadata detection is based on object detection in document scans using the YOLOv8 model. Detection using the fully connected neural network achieves an F1 score of 0.83 on the book dataset and 0.78 on the periodicals dataset. The Transformer Encoder network achieves F1 scores of 0.84 on the book dataset and 0.59 on the periodicals dataset. The YOLO model achieves an F1 score of 0.86 (confidence at 0.549) on the book dataset and 0.7 (confidence at 0.336) on the periodicals dataset.

Klíčová slova

automatická extrakce metadat, metadata, klasifikace dat dokumentů, zpracování dokumentů, neuronové sítě, YOLO, Transformer Encoder

Keywords

automatic metadata extraction, metadata, document data classification, document processing, neural networks, YOLO, Transformer Encoder

Citace

KŘIVÁNEK, Jakub. *Automatizovaná extrakce strukturovaných dat dokumentů*. Brno, 2024. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Jan Kohút

Automatizovaná extrakce strukturovaných dat dokumentů

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jana Kohúta. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Jakub Křivánek
9. května 2024

Poděkování

Na tomto místě bych chtěl vyjádřit své díky všem, kteří mi pomáhali a podporovali mě během práce na mé bakalářské práci. Speciální poděkování patří mému vedoucímu práce, Ing. Janu Kohútovi, za jeho cenné rady, odborné vedení a trpělivost. Dále děkuji členům mé rodiny a přátelům, kteří mi poskytli morální podporu a pochopení v průběhu celého studia. Výpočetní zdroje byly poskytnuty projektem e-INFRA CZ (ID: 90254), který podporuje Ministerstvo školství, mládeže a tělovýchovy České republiky.

Obsah

Kapitola 1

Úvod

Extrakce strukturovaných dat, tzv. metadat, z dokumentů je problém převodu cenných informací o názvu dokumentu, jménu autora, datu vydání atd. Tyto metadata jsou v knihovnách, archivech a dalších informačních institucích, použita k organizaci, správu a vyhledávání. Tématem této práce je automatické získání těchto metadat ze skenů dokumentů českých knihoven s cílem usnadnit a zefektivnit proces práce knihovníků, který je v současnosti velmi časově náročný a převážně manuální. Práce je realizovaná v rámci projektu Smart digilinka[1], jehož cílem je vývoj nástrojů využívajících strojového učení v procesu digitalizace a vytvoření poloprovozu digitalizační linky tak, aby se tento proces výrazně zefektivnil a bylo z něj odstraněno co nejvíce činností.

Hlavním zaměřením je navrhnout a vyvinout metody pro automatické rozpoznávání a extrakci metadata z dokumentů, které jsou dostupné ve formě skenů. To zahrnuje vytvoření dvou datových sad. Jedna sada se skládá z 670 stan z 10 českých knihy. Druhá sada je složena ze 700 titulních stránek z přibližné 350 periodik. Sada periodik, jejichž struktura a formát mohou být pro proces automatizace značnou výzvou, je mnohem rozmanitější a komplexnější než sada knih.

V průběhu práce byly vyvinuty a testovány různé přístupy k analýze dokumentů, od plně propojených neuronových sítí k „state of the art“ modelu YOLOv8 [10]. Pro tyto přístupy jsem se věnoval druhům vstupů, které by měli pozitivní vliv na výsledky přístupu, a způsobům, tyto vstupy popřípadě modifikovat. Příklad tohoto je obarvovaní sémanticky významných informací na obrázcích pro model YOLO, kde tato přidaná informace významně zlepšila výsledky modelu na datasetu periodik.

[1] https://www.mzk.cz/o-knihovne/vyzkum-projekty/naki-iii/smart-digilinka

Kapitola 2

Přístupy k automatické extrakci metadat z obrázků dokumentů.

Metadata mají zásadní roli v knihovnách, archivech a dalších informačních institucích. Jedná se o strukturované údaje, které poskytují informace o jiných datech, typicky o dokumentech, knihách, časopisech atd. Tyto data jsou zásadní pro organizaci, správu a vyhledávání. Problém nastává u jejich výroby. Proces výroby metadat je momentálně časově náročný a schopnost vyrábět metadata institucemi spravující tyto dokumenty je omezená. To způsobuje, že každoročně počet digitalizovaných stran menší, než počet fontů co do knihoven přibývá, jak je na stránkách projektu Smart digilinka[1], kterého je tato práce součástí, řečeno. Hlavním úkolem při automatické extrakci metadat je ulehčit práci knihovníkům tak, aby se současný proces výrazně zefektivnil a předejít podléhání starších dokumentů zkáze.

Na řešení tohoto problému jsou k dispozici naskenované dokumenty různých typů (knihy, noviny, časopisy ...), jak zobrazují ukázky 4.4. Z těchto skenů lze při extrakci metadat reprezentovat daný dokument různými přístupy. Reprezentace, použité ke klasifikaci dat z dokumentů, se dají rozdělit na tři hlavní kategorie [3, 9], které se dají vzájemně kombinovat. První z nich je samotný text dokumentu. Textová reprezentace se zaměřuje na samotný obsah dokumentu – slova, věty a odstavce. Tato forma reprezentace je klíčová pro analýzu syntaktických a sémantických vztahů mezi slovními jednotkami. Syntaktická analýza zkoumá gramatickou strukturu textu, zatímco sémantická analýza se snaží odhalit významy a kontext slov a frází. Další z přístupů je rozložení (layout). Rozložení zahrnuje pozice a rozměry textových oblastí jako jsou nadpisy, odstavce, popisky atd. Reprezentace obrazu se zaměřuje na vizuální aspekty dokumentu, které nelze zachytit pouze pomocí textu nebo rozložení. Tento přístup zahrnuje analýzu barev, typů písma, grafických prvků jako jsou obrázky, loga, grafy a další vizuální symboly. Tato kategorie je klíčová pro rozpoznání vizuálních vzorů a stylů, které mohou být důležité pro identifikaci a klasifikaci dokumentů podle jejich vizuální identity.

Textová reprezentace. Text dokumentu je nejdůležitější z informací získaných z obrázku dokumentu. Může být použit při klasifikaci jednotlivých tříd metadat, ale hlavně po klasifikaci jsou z textu (v klasifikovaných oblastech) získána samotná metadata, nebo alespoň jejich podoba, než se dále zpracuje do vhodného formátu. Zisk textového přepisu z obrazu je založen na systémech optické rozpoznávání znaků (OCR). V této práci je použit

[1] https://www.mzk.cz/o-knihovne/vyzkum-projekty/naki-iii/smart-digilinka

systém PERO-OCR [12]. Nezpracovaný text není přímo kompatibilní s neuronovými sítěmi, proto je nezbytné reprezentovat jej pomocí jiných metod.

Jedna z takových metod je **Bag of Words (BoW)** [11]. V tomto modelu je text (například věta nebo dokument) převeden na vektor, který reprezentuje početnost jednotlivých slov ve slovníku, který obsahuje všechna slova, jež se v daném textu mohou objevit. Tento model ignoruje syntaxi a pořadí slov, ale zachovává informace o slovech a jejich frekvenci. Problém tohoto přístupu je, že slovníky mohou být poměrně velké, což vede k vysoké rozměrnosti vektorů a potenciálně k problémům s výpočetní efektivností a paměťovou náročností.

Další přístup je **Term Frequency-Inverse Document Frequency (TF-IDF)** [11]. Metoda používaná k posouzení významnosti slova v dokumentu v rámci sbírky dokumentů. Pomocí dvou klíčových ukazatelů, TF[2] a IDF[3], tato metoda hodnotí slova na základě jejich frekvence v dokumentu a jejich vzácnosti napříč všemi dokumenty ve sbírce.

Textu lze také reprezentovat slovním embedding. Tento slovní embedding je schopen zaznamenat sémantické a syntaktické vztahy mezi slovy. Jeden z přístupů vytváření slovního embedding je BERT, který byl použit na extrakci metadat v práci [5]. Model **BERT (Bidirectional Encoder Representations from Transformers)** [7] je jedním z nejpokročilejších přístupů v oblasti zpracování přirozeného jazyka. Využívá techniku architekturu Transformer, představenou v práci [24], což jsou modely založené na pozornostních mechanismech (attention mechanisms), které umožňují modelu současně zpracovávat celé věty, což je významný rozdíl oproti předchozím modelům, které zpracovávaly text sekvenčně. Díky tomu je BERT schopen lépe pochopit kontext a význam slov v textu, protože dokáže zohlednit všechna slova věty naráz. Další přístupy slovního embedding jsou například **GloVe** [18] a **Word2Vec** [16].

Rozložení. Údaje využívané při práci s rozložením můžou být pozičního typu. Ty potom vyjadřují **pozici**, na které se daná oblast textu nachází. Dále to také mohou být **údaje o rozměrech** dané oblasti textu. Pozice a rozměry se dají vyjádřit, jak je použito v práci StructuralLM [13], čtveřicí (x_0, y_0, x_1, y_1), kde (x_0, y_0) udává levý horní roh oblasti a (x_1, y_1) pravý spodní roh. Tento přístup se dvěma body je ale možné použít pouze v případě, že oblasti textu (řádky) jsou vodorovné. Tyto informace nejčastěji dostáváme z obrazu pomocí systému OCR. OCR nástroje nejen převádějí obrazový obsah textu na strojově čitelný formát, ale také mohou identifikovat a zachovat původní rozložení textu. Tato schopnost je klíčová pro rozpoznání strukturovaných informací v dokumentech. Zejména u metadat, které jsou často umístěny na konkrétních místech dokumentu a mají specifický vizuální styl, který je odlišuje od hlavního textu, jako například *titulek* v periodikách (4.4) je na vrchní části strany a je výrazně větší oproti ostatnímu textu.

Obraz. Využití obrazové analýzy v procesu automatické extrakce metadat z dokumentů představuje důležitou technologickou výhodu, která rozšiřuje možnosti zpracování a organizace informací. Tato metoda může být klíčová zejména ve fázích, kde je nutné zachytit a interpretovat vizuální charakteristiky dokumentů, jako jsou formáty, loga, speciální typografie a další grafické prvky. Opět je problém jak obraz reprezentovat. Obraz se sice dá přímo vkládat do sítí například pomocí reprezentace každého pixelu nebo podvzorkovaného souboru původního obrazu třemi barvami (RGB), ale takové přímé vkládání obrazů je ne-

[2]Term Frequency (TF) ukazuje, jak často se termín objevuje v daném dokumentu v poměru k celkovému počtu termínů dokumentu, což pomáhá vyvážit vliv délky dokumentu.

[3]Inverse Document Frequency (IDF) vyjadřuje, jak unikátní nebo vzácné je slovo ve sbírce dokumentů, čímž snižuje váhu běžných slov.

Obrázek 2.1: Plně propojená neuronová síť (FCNN) používaná v této práci pro klasifikaci.

efektivní z hlediska výpočetních zdrojů a paměti, zejména u velkých datových sad. Navíc, při použití takové reprezentace se mohou ztratit některé důležité informace o struktuře nebo obsahu obrazu, které by mohly být důležité pro analýzu a zpracování obrazů. Pro zlepšení extrakce užitečných rysů a snížení dimenzionality dat se proto používají metody jako **lineárního embedding** (Linear Embedding) [9, 2], který transformuje obrazová data do kompaktnější a informativnější formy, vhodné pro další zpracování. Tento proces zahrnuje extrakci vlastností, kde neuronové sítě, zejména konvoluční neuronové sítě (CNN), jsou využívány k extrakci vlastností z obrazů. Tyto vlastnosti jsou reprezentací obrazu v komprimované formě. Následná dimenzionální redukce – techniky jako PCA (Principal Component Analysis) nebo t-SNE (t-distributed Stochastic Neighbor Embedding) jsou použity k redukci dimenzí extrahovaných vlastností na zpracovatelnější úroveň, zatímco zachovávají klíčové informace. Výsledné nízkorozměrné reprezentace jsou známé jako embedding, které mohou být použity pro různé účely, včetně klasifikace, klasterizace nebo vyhledávání podobnosti.

Jak už bylo řečeno obraz lze dávat do sítě reprezentovaný hodnotami pixelů. V práci [4] autoři použili **Mask R-CNN** s Feature Pyramid Network (FPN) [14] deteckci a klasifikaci metadat. Další modely, jako **YOLO** (You Only Look Once), **SSD** (Single Shot Multibox Detector) nebo **Faster R-CNN** (Region-based Convolutional Neural Networks), jsou široce používány pro detekci objektů. Tato metoda využívání pixelových hodnot obrazu se při detekci objektů osvědčuje, protože umožňuje modelům přesně lokalizovat a klasifikovat různé objekty v obrazových datech. Pro účely extrakce metadat z dokumentů mohou být tyto modely obzvláště užitečné, protože umožňují identifikovat a extrahovat specifické prvky z dokumentů, jako jsou textové bloky, obrázky, grafy, tabulky a další vizuální informace.

2.1 Plně propojená neuronová síť (FCNN)

Plně propojená neuronová síť je jedním z nejzákladnějších typů neuronových sítí používaných v oblasti strojového učení a umělé inteligence. Tyto sítě jsou také známé jako „dense“ nebo „fully connected“ sítě, protože každý neuron ve vrstvě je propojen se všemi neurony v předchozí a následující vrstvě. Je možno tuto neuronovou síť reprezentovat rovnicí:

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right), \tag{2.1}$$

kde x jsou vstupy vrstev, w jsou váhy vrstev, b je bias a f je aktivační funkce na výstupu sítě. Struktura plně propojené neuronové sítě je jednoduchá a přímá, což ji činí vhodnou pro řadu základních úloh strojového učení, včetně klasifikace a regrese.

V této práci je implementována FCNN zobrazená na obrázku 2.1. Hlavním účelem této sítě je mít nějaký nejjednodušší model pro srovnání s ostatními implementacemi, protože neexistují žádné výsledky jiných prací na datasetu použitém v této práci.

2.2 Neuronová síť využívající Transformer Encoder (TENN)

V této kapitole se zaměříme na klíčové aspekty architektury encoderu, který je součástí modelu Transformer [24]. Transformer je model založený na mechanismu pozornosti (attention), který se vyznačuje vysokou paralelizací a schopností efektivně zpracovávat sekvence dat bez potřeby rekurzivních nebo konvolučních vrstev. V klasifikačních úlohách je Transformer Encoder schopen zpracovat celý vstup jako celek a generovat reprezentace, které jsou následně využity klasifikační vrstvou. Díky schopnosti modelu efektivně zachytit vztahy v datech a důležité kontextové informace je Transformer Encoder ideální pro složité klasifikační úlohy.

Architektura Transformer Encoderu. Transformer Encoder architektura se skládá z několika vrstev, kde každá vrstva obsahuje multi-head attention mechanismus a position-wise feed-forward network. Klíčovou inovací je mechanismus pozornosti, který umožňuje modelu získat informace z různých částí vstupních dat bez ohledu na jejich sekvenční pozici.

Multi-head self-attention umožňuje modelu současně se zaměřit na různé segmenty vstupu, což zlepšuje schopnost modelu zachytit různé kontextové závislosti. Tento mechanismus je základem pro flexibilitu a sílu Transformer modelů v rámci různých aplikací. Výpočet Multi-head attention můžeme vyjádřit následovně [24]:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.2}$$

$$\text{head}_i = \text{Attention}(QW_i^Q, K\ W_i^K, V\ W_i^V) \tag{2.3}$$

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O, \tag{2.4}$$

kde Q, K, V jsou matice dotazů (queries), klíčů (keys) a hodnot (values), QW_i^Q, KW_i^K, VW_i^V jsou parametrické matice pro každou hlavu i, W^O je výstupní parametrická matice, která kombinuje výstupy všech hlav a d_k je dimenze klíčů, použitá pro škálování v dotykovém součinu, aby se zabránilo příliš velkým hodnotám v exponentu softmax.

Každá vrstva dále obsahuje **Position-Wise Feed-Forward Network**, které aplikují stejné lineární transformace na každé pozici vstupu nezávisle, což přispívá k paralelizaci a efektivnímu zpracování dat.

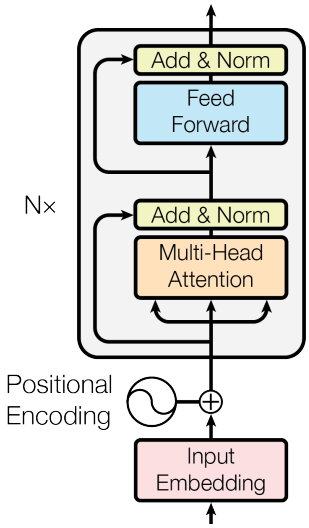
Poziční kódování. Jelikož model Transformer neobsahuje žádné rekurzivní nebo konvoluční komponenty, je nezbytné, aby se do modelu explicitně přidaly informace o pozici jednotlivých vstupů, v kontextu této práce řádků, ve vstupní sekvenci. Toto se realizuje pomocí pozičního kódování, které je přičteno k vstupním vektorem před jejich zpracováním v první vrstvě encoderu.

Poziční kódování může být implementováno pomocí sinusových a kosinusových funkcí různých frekvencí [24], kde každá dimenze vstupního vektoru odpovídá sinusové nebo kosinusové funkci s určitou periodicitou. Tento způsob kódování umožňuje modelu zachytit informace o relativních pozicích řádků ve vstupní sekvenci.

V práci byli použity tři druhy pozičního kódování.

- **1D sekvenční** – Pozice je v tomto případě umístění prvku (řádku) v sekvenci. Implementace tohoto kódování byla realizovaná podle PyTorch návodu[4] a zle ho vyjádřit

[4] https://pytorch.org/tutorials/beginner/transformer_tutorial.html



Obrázek 2.2: Schéma Transformer Encoder architektury. Ke vstupu je přičteno poziční kódování, po čemž následuje N Transformer Encoder vrstev (převzato z [24]).

rovnicemi:

$$\begin{aligned} \mathrm{PE}(pos, 2i) &= \sin\left(pos \cdot 10000^{-\frac{2i}{d}}\right) \\ \mathrm{PE}(pos, 2i+1) &= \cos\left(pos \cdot 10000^{-\frac{2i}{d}}\right), \end{aligned} \tag{2.5}$$

kde $\mathrm{PE}_{\mathrm{seq}}$ označuje hodnotu pozičního kódování na pozici pos a dimenzi j, pos je index prvku (řádku) v sekvenci, d je dimenze vstupního vektoru a i je index poloviny d, který běží od 0 do $\frac{d}{2} - 1$.

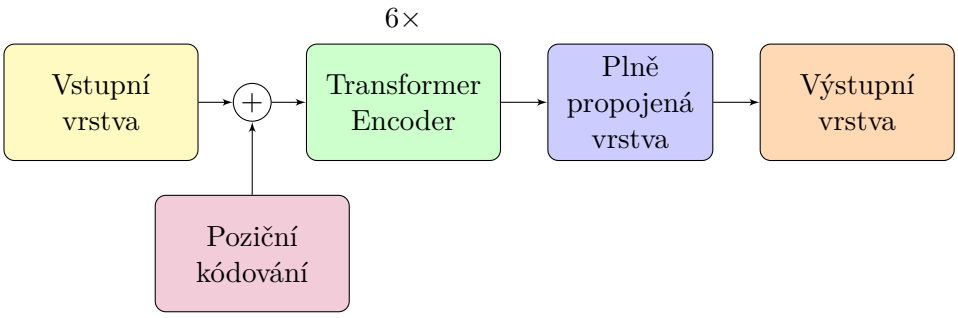
- **1D pořadí na stránce** – OCR dává řádky v určité pořadí, které většinou odpovídá pořadí, jak by byly po sobě řádky čteny. Toto poziční kódování využívá toto pořadí místo pořadí v dané sekvenci. Implementace byla udělána podle třídy `PositionalEncoding1D` z github repozitáře[5] a je vyjádřena stejnými rovnicemi jako 1D sekvenční kódování 2.5.

- **2D** – Kódování využívá x a y pozici prvku (řádku) na straně. Implementace byla inspirována `PositionalEncoding1D` ze stejného repozitářem[6] jako kódování 1D pořadí na stránce. Lze ho reprezentovat rovnicemi:

$$\begin{aligned} \mathrm{PE}(x, y, 2i) &= \sin\left(x \cdot 10000^{-\left(\frac{4i}{d}\right)}\right) \\ \mathrm{PE}(x, y, 2i+1) &= \cos\left(x \cdot 10000^{-\left(\frac{4i}{d}\right)}\right) \\ \mathrm{PE}(x, y, 2j+\frac{d}{2}) &= \sin\left(y \cdot 10000^{-\left(\frac{4j}{d}\right)}\right) \\ \mathrm{PE}(x, y, 2j+1+\frac{d}{2}) &= \cos\left(y \cdot 10000^{-\left(\frac{4j}{d}\right)}\right), \end{aligned} \tag{2.6}$$

kde (x, y) reprezentuje bod středu *baseline*[7], i a j jsou celočíselné indexy v rozsahu $[0, \frac{D}{4})$, kde D je dimenze vstupního vektoru.

[5] https://github.com/tatp22/multidim-positional-encoding/tree/master
[6] Ibid.
[7] soubor bodů, na kterých řádek „leží“



Obrázek 2.3: Síť využívající Transformer Encoder (TENN) používaná v této práci pro klasifikaci.

2.3 Přístup využívající YOLOv8 model

YOLOv8 [10], navazující na úspěchy svých předchůdců ve slavné rodině YOLO (You Only Look Once), představuje nejnovější inovace v oblasti detekce objektů. Tento model je vyvinutý s důrazem na rychlost a přesnost, a je zvláště vhodný pro aplikace vyžadující zpracování v realném čase. V porovnání s modely jako je Faster R-CNN a SSD, YOLOv8 nabízí výhody v rychlosti a jednoduchosti implementace. Zatímco R-CNN a SSD modely využívají metody, které zahrnují vícefázové zpracování a komplexnější postupy pro detekci objektů, YOLOv8 dosahuje srovnatelné nebo lepší úrovně přesnosti s méně výpočetní zátěží. Faster R-CNN, i přes svou vysokou přesnost, je pomalejší kvůli své potřebě generovat návrhy oblastí a následně je klasifikovat, zatímco SSD je rychlejší, ale může trpět na nižší přesností v detekci malých objektů.

Architektura YOLOv8 modelu. Model se skládá ze dvou hlavních částí, Backbone a Head. Graf struktury YOLOv8 si lze prohlédnout na github[8]. **Backbone** zahrnuje vrstvy, které postupně zmenšují obraz, a vícenásobné konvoluční vrstvy, které postupně z několika rozlišení extrahují příznaky. **Head** zpracovává rysy získané z Backbone části a provádí samotnou detekci objektů. Tato část obsahuje několik vrstev, které provádí predikce umístění objektů (bounding boxes), jejich tříd a pravděpodobnosti přítomnosti objektů. Detekce probíhá na více úrovních, což umožňuje modelu efektivně detekovat objekty různých velikostí.

2.4 Srovnání přístupů FCNN, TENN a YOLO

Při porovnávání přístupů, popsaných výše v této kapitole, je důležité posoudit jejich výhody a nevýhody při použití v procesu extrakce metadat z dokumentů.

FCNN a TENN. Obě tyto sítě spadají pod stejný přístup, klasifikují řádky dokumentu na základě modalit textu a rozložení, pouze s jinou vnitřní architekturou sítě. Řádky dokumentu jsou přečteny OCR a v tomto spočívá první z nevýhod tohoto přístupu. Pokud není řádek z dokumentu přečten OCR, nemůže ani být klasifikován, tedy ani z něho nemohou být získána metadata. Tento problém je přítomný hlavně u třídy *titulek* v periodikách, kde je často napsán velkými, tučnými a okrasnými písmeny, nebo je zakomponován do nějakého

[8] https://github.com/ultralytics/ultralytics/issues/189, odkaz na github issue s grafem architektury YOLOv8 od uživatele RangeKing

obrázku či nákresu. Příklady těchto případů jsou vidět v obrázcích 2.4. S těmito případy má OCR problém a často buď přečte text špatně, nebo ho nepřečte vůbec.

Další problém nastává po klasifikaci, jelikož sítě klasifikují řádky, nikoliv přesné oblasti metadat. Proto je nutné udělat post-processing, abychom získali z řádku část, která je ke klasifikované třídě relevantní. Toto je možné u některých tříd, jako například *datum vydání* nebo *místo vydání* u periodik, realizovat podobným využitím Czert NER, který je popsán v 4.5. Text klasifikovaného řádku by byl vložen do NER a slova určena jako entita s relevancí k dané klasifikované třídě by byla z řádku vybrána.

Výhodou tohoto systému je jeho snadná možnost rozšířitelnosti. V současné podobě do sítě nejde modalita obrazu. K přidání by stačilo implementovat například lineární embedding a ten by se pouze připojoval za dosud používaný příznakový vektor.

YOLO. Přístup s využitím modelu YOLO vytváří detekce na jakémkoliv místě na vstupním obrázku. Není tedy omezen jak předchozí dvě neuronové sítě, které se pracovali pouze s řádky. To přináší výhodu tohoto přístupu. Na výstupu z tohoto modelu jsou oblasti původního obrázku, které je už jen potřeba přečíst systémem OCR. PERO-OCR, OCR systém, použitý v této práci, se z mého pozorování výsledků zdá, že funguje lépe, když zpracovává pouze vyříznutou oblast, kterou označilo YOLO. Na základě mých pozorování se domnívám, že to je způsobeno tím, že se často odstraní dekorativní prvky, které komplikují práci OCR, pokud jsou přítomné.

Nevýhoda tohoto systému je, že do něj nejde textová modalita. To může vést k tomu, že tento přístup bude málo, nebo dokonce vůbec, reagovat na syntaktické a sémantické vlastnosti dokumentu. Snaha tento nedostatek vyvážit je v této práci práci pomocí barvení syntakticky významných slov různými barvami, podrobněji popsané v 4.5.

Další nevýhoda, kterou nelze přehlédnout, spočívá v obtížnosti rozšiřování a úprav. YOLOv8 je vyspělý „state of the art“ model, což znamená, že patří mezi nejmodernější a technologicky nejpropracovanější dostupné systémy v oblasti strojového vidění. Tato špičková technologie však přináší i určité komplikace, jelikož jakákoliv modifikace či rozšíření existujícího modelu by vyžadovala rozsáhlé odborné znalosti.

Obrázek 2.4: Obrázky zobrazující případy, kdy je *titulek* součástí obrázku či nákresu.

Kapitola 3

Klasifikace

Tato kapitola je věnována klasifikaci, která patří mezi nejrozšířenější aplikace strojového učení současnosti. Klasifikace je proces přiřazení třídy objektu v daném kontextu na základě patřičných vstupů.

V této kapitole vysvětluji různé vyhodnocovací techniky, které jsou nezbytné pro objektivní posouzení účinnosti a přesnosti použitých klasifikačních modelů. Dále taky objasňuji obecnou terminologie pro neuronové sítě.

3.1 Terminologie související s klasifikací a neuronovými sítěmi

V této sekci se seznámíme s klíčovými termíny a pojmy, které jsou často používány v oblasti klasifikace v rámci strojového učení. Porozumění těmto termínům je zásadní pro hlubší pochopení metod a technik popsaných v následujících částech.

- **Příznaky** – Vlastnosti nebo charakteristiky dat, které jsou použity k popisu instance. Tyto informace musí být přeměněny do číselné podoby než mohou být použity jako vstupy do sítě.
- **Instance** – Příznakový vektor, který reprezentuje objekt v datové sadě. Lze si představit jako řádek v tabulce dat, kde každý sloupec reprezentuje příznak.
- **Datová sada** – Soubor instancí použitý pro trénování a validaci modelu.
- **Trénovací sada** – Podmnožina datové sady, na kterém je klasifikátor trénován. Tato sada obsahuje příklady s předem definovanými třídami, které model používá pro naučení rozpoznávání vzorů.
- **Validační/Testovací sada** – Podmnožina datové sady, používaný pro ověření účinnosti modelu. Zásadní vlastnost je, že nesdílí instance s trénovací sadou. Tato sada umožňuje zjistit, jak dobře model generalizuje na nová, dříve neviděná data.
- **Ground truth** – V kontextu datové analýzy, strojového učení nebo jiných vědeckých oblastí označuje tato fráze přesné, faktické, objektivní informace, které jsou používány jako standard pro ověření a porovnání výsledků experimentálních testů nebo modelů.
- **True Positives (TP)** – Počet správně klasifikovaných pozitivních případů. Tato hodnota označuje, kolik skutečně pozitivních případů bylo správně identifikováno modelem jako pozitivní.

- **True Negatives (TN)** – Počet správně klasifikovaných negativních případů. Tato hodnota označuje, kolik skutečně negativních případů bylo správně identifikováno modelem jako negativní.
- **False Positives (FP)** – Počet negativních případů, které byly nesprávně klasifikovány jako pozitivní. Tato hodnota označuje, kolik negativních případů bylo chybně označeno modelem jako pozitivní.
- **False Negatives (FN)** – Počet pozitivních případů, které byly nesprávně klasifikovány jako negativní. Tato hodnota označuje, kolik pozitivních případů bylo chybně označeno modelem jako negativní.
- **Character Error Rate (CER)** – Character Error Rate vyjadřuje procento znaků, které byly přečteny systénem OCR špatně. Jeho vztah lze vyjádřit následovně:

$$\text{CER} = \frac{S + I + D}{N}, \tag{3.1}$$

kde S je počet substitucí (znaky rozpoznané chybně), I je počet vložení (nadbytečné znaky), D je počet smazání (chybějící znaky) a N je celkový počet znaků ve správném přepise.

3.2 Klasifikační metriky

V oblasti strojového učení a datové analýze jsou metriky klasifikace klíčovými nástroji pro hodnocení výkonnosti klasifikačních modelů. Tyto metriky pomáhají pochopit, jak dobře model predikuje různé třídy. Mezi nejzákladnější a nejpoužívanější metriky patří precision, recall, F1 Skóre a accuracy. V této kapitole se podíváme na každou z těchto metrik, jak se počítají a jaký mají význam v rámci hodnocení modelů.

Precision. Precision je metrika, která měří, kolik z pozitivních predikcí modelu je skutečně pozitivních. Jinými slovy, je to podíl správně identifikovaných pozitivních případů vzhledem ke všem případům, které model označil jako pozitivní. Matematicky je precision definována jako:

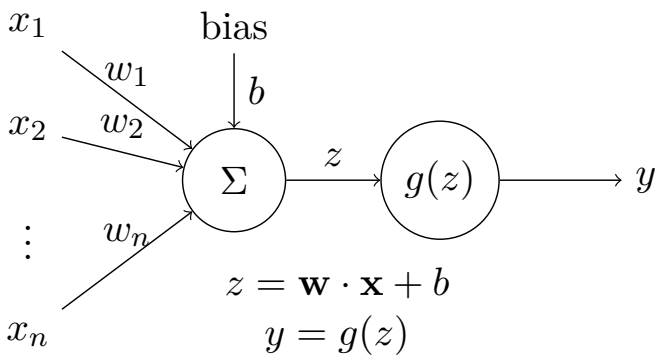
$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

Recall. Recall neboli úplnost, známá také jako senzitivita, měří, kolik z aktuálních pozitivních případů bylo správně identifikováno modelem. Vyjadřuje schopnost modelu detekovat všechny relevantní instance. Recall je definován jako:

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

F1 Skóre. F1 Skóre je harmonický průměr přesnosti a úplnosti. Tato metrika se používá, když potřebujeme vyvážit přesnost a úplnost, zejména když je distribuce tříd nerovnoměrná. F1 Skóre je užitečné v situacích, kdy není vhodné favorizovat ani přesnost ani úplnost. F1 Skóre je definováno jako:

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{3.4}$$



Obrázek 3.1: Ukázka umělého neuronu s n vstupy x, váhami w a aktivační funkcí g.

Accuracy. Accuracy měří celkovou správnost modelu tím, že počítá podíl správných predikcí (jak pozitivních, tak negativních) ze všech predikcí. Je to jedna z nejjednodušších a nejintuitivnějších metrik, ale může být zavádějící, zejména v případě nerovnoměrné distribuce tříd. Je definována jako:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.5}$$

3.3 Neuronové sítě

Neuronové sítě jsou modely inspirované fungováním lidského mozku, které se používají v oblastech strojového učení a umělé inteligence. Jsou tvořeny jednotkami zvanými umělými neurony, které jsou organizovány do vrstev a propojeny váhami představujícími sílu spojení mezi neurony. Informace ve formě dat procházejí sítí od vstupních neuronů přes skryté vrstvy, kde jsou zpracovávány, až k výstupním neuronům, které produkují výsledek. V dnešní době hrají neuronové sítě klíčovou roli v klasifikaci, jelikož jsou tyto sítě schopné učit se ze složitých a rozmanitých datových sad, což umožňuje automatizovat a zefektivnit mnoho úloh, které byly tradičně závislé na manuální klasifikaci.

Umělý neuron a aktivační funkce

Umělý neuron, často označovaný jako perceptron, je inspirován biologickými neurony nalezenými v mozku. Tyto jednoduché jednotky jsou základem, na kterém jsou postaveny všechny typy neuronových sítí. Jeho funkcí je přijímat vstupy, vážit je pomocí přiřazených vah, a generovat výstup skrze aktivací funkci, jak je ukázáno na obrázku 3.1.

Každý neuron přijímá jeden nebo více vstupů, které mohou pocházet buď přímo z dat, nebo od výstupů jiných neuronů v síti. Každý vstup je násoben váhou, která určuje jeho význam. Tyto váhy jsou nastavitelné parametry neuronu a díky jejích změnám pří trénování jsou klíčem k učení neuronových sítí. Bias je další parametr, který posouvá aktivaci neuronu tak, aby lépe odpovídala požadovaným výstupům. Výsledný součet vážených vstupů a bias tvoří agregovaný vstup neuronu. Agregovaný vstup je následně transformován aktivací funkcí, která určuje, zda a jak neuron „aktivuje“ (tj. reaguje).

Aktivační funkce je klíčová složka pro schopnost neuronových sítí učit se a modelovat komplexní funkce. Bez nelinearity, kterou aktivační funkce poskytuje, by neuronová síť byla schopna modelovat pouze lineární vztahy, což je pro většinu praktických aplikací nedostatečné.

Rectified Linear Unit (ReLU). ReLU je jednou z nejčastěji používaných aktivací funkcí v současných neuronových sítích. Aktivuje neurony pouze, když jejich vstup je kladný, a je definována jako:

$$\text{ReLU}(x) = \max(0, x), \tag{3.6}$$

Leaky ReLU. Tato varianta ReLU dovoluje malý gradient i pro negativní vstupy, což pomáhá udržet aktivní gradienty během tréninku a potenciálně zabraňuje problémům s „mrtvými neurony“ [25]. Definuje se jako:

$$\text{LeakyReLU}(x) = \max(ax, x), \tag{3.7}$$

kde a je zvolená hodnota, určující jak velký bude záporný sklon.

Sigmoid. Funkce sigmoid, vyjádřena vztahem:

$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{3.8}$$

mapuje libovolný reálný vstupní hodnotu na interval (0, 1), díky tomuto je užitečná pro binární klasifikaci jako funkce výstupní vrstvy.

Softmax. Softmax je další důležitá aktivací funkce, často používaná ve výstupních vrstvách neuronových sítí pro klasifikaci do více tříd. Tato funkce

$$\text{Softmax}(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}, \tag{3.9}$$

převádí ne-normalizované výstupy na distribuci pravděpodobností, kde každá hodnota výstupu reprezentuje pravděpodobnost, že daný vstup patří do určité třídy. Na rozdíl od Sigmoid funkce součet softmax výstupních pravděpodobností je 1, proto není ideální, když chceme klasifikovat více tříd pro jednu instanci.

Principy učení neuronových sítí

Neuronové sítě se učí optimalizací svých vah tak, aby minimalizovaly rozdíl mezi predikovanými a skutečnými výstupy. Tento proces je realizován pomocí algoritmu zpětné propagace společně s metodou optimalizace, nejčastěji gradientním sestupem.

Gradientní sestup (Gradient Descent). Gradientní sestup je iterativní optimalizační algoritmus používaný k nalezení minima funkce. Jeho základní myšlenka spočívá v tom, že postupně krokuje v opačném směru gradientu funkce s cílem najít lokální minimum diferencovatelné funkce. Existují tři základní typy. **Batch Gradient Descent (Batch GD)**, který vypočítá gradient na celém datasetu, což může být velmi náročné na výpočetní zdroje u velkých datových sad. **Stochastic Gradient Descent (SGD)** vypočítá gradient na základě jednoho vzorku dat, což způsobuje, že aktualizace parametrů jsou méně přesné, ale mnohem rychlejší. Kompromis mezi Batch GD a SGD je **Mini-batch gradientní sestup**, kde se gradient vypočítává pouze na malé podmnožině dat.

Zpětná propagace chyby (Backpropagation). Zpětná propagace chyby je klíčová technika používaná pro trénování neuronových sítí. Spočívá v propagaci chyby (rozporu mezi skutečným a predikovaným výstupem) od výstupní vrstvy zpět k vstupní vrstvě. Během tohoto procesu jsou upravovány váhy tak, aby se snížila celková chyba sítě a aby se zajistilo, že se neuronová síť "učí"z každého příkladu v datové sadě. Proces zpětné propagace se skládá z několika kroků:

1. **Dopředný Průchod (Forward Pass)** – Nejprve se provede dopředný průchod, kde jsou vstupní data poslána do neuronové sítě a postupně procházejí různými vrstvami až do výstupní vrstvy, kde jsou vygenerovány predikce. Výstupy jsou vypočítány postupně pro každou vrstvu na základě aktuálních vah a bias.

2. **Výpočet chyby** – Po dopředném průchodu se vyhodnotí výstupy sítě porovnáním s očekávanými výstupy (cílové hodnoty). Chyba je obvykle vypočítána pomocí ztrátové funkce, jako je například Mean squared error (MSE) pro regresní úkoly, pro klasifikaci Cross-Entropy Loss (CELoss) nebo Binary Cross Entropy (BCELoss), která byla použita pro FCNN a TENN v této práci. BCELoss může být popsána následovně [6]:

$$\begin{aligned} \ell(x, y) &= \text{průměr}(L) \\ L &= \{l_1, \ldots, l_N\}^\top \\ l_n &= -w_n \left[y_n \cdot \log x_n + (1 - y_n) \cdot \log(1 - x_n) \right], \end{aligned} \tag{3.10}$$

   kde x představuje předpovídané hodnoty nebo pravděpodobnosti generované modelem, y jsou skutečná označení nebo skutečné hodnoty, w představuje váhu nebo důležitost přiřazenou každému vzorku v batch[1] a N je velikost batch.

3. **Zpětný Průchod (Backward Pass)** – Zpětný průchod (Backpropagation) je klíčovým procesem v učení neuronových sítí, kde je hlavním cílem výpočet gradientů ztrátové funkce vzhledem ke všem parametrům sítě – tedy vahám a bias. Tento proces se opírá o pravidlo o derivaci složené funkce [1], které umožňuje efektivní propagaci chyb od výstupní vrstvy směrem k vstupním vrstvám. Během zpětného průchodu je pro každý neuron vypočítán gradient, který ukazuje, jakým směrem a jak moc by měly být jeho váhy a bias upraveny, aby došlo ke snížení celkové chyby sítě. Tyto informace jsou poté využity pro aktualizaci parametrů.

4. **Aktualizace parametrů** – Po výpočtu gradientů následuje aktualizace vah a bias. Tento krok se často provádí s využitím algoritmu jako je stochastický gradientní sestup (SGD) nebo jeho varianty, jako jsou Adam nebo RMSprop. Aktualizační rovnice může vypadat například takto:

$$w_{\text{nová}} = w_{\text{stará}} - \eta \cdot \nabla_w \mathcal{L}, \tag{3.11}$$

   kde w představuje váhy, η je learning rate (rychlost učení) a $\nabla_w \mathcal{L}$ gradient ztrátové funkce vzhledem k vahám.

5. **Iterace** – Zpětná propagace obvykle není prováděna jednou, ale opakuje se mnohokrát v rámci tréninkových epoch[2]. V každé epoše se data znovu prochází dopředným a zpětným průchodem, chyby se vyhodnocují a parametry se aktualizují, dokud síť nezačne dosahovat přijatelné úrovně přesnosti nebo jiných metrik výkonnosti.

[1] batch – podmnožina datové sady určité velikosti, která slouží jako vstup do sítě místo celého datasetu
[2] epocha v kontextu strojového učení – jeden kompletní průchod trénovací datovou sadou

Kapitola 4

Datové sady českých knih a periodik

Kvůli motivaci práce v rámci projektu Smart digilinka[1], podrobněji popsané v úvodu 1, nebyla použita již existující datová sada, jako například IIT-CDIP dataset použitý v LayoutLMv3 [9]. Místo toho jsem s pomocí vedoucího práce a Ondřeje Lehrla z Národní knihovny České republiky[2].vytvořili dvě datové sady soustředěné na převážně české knihy a periodika.

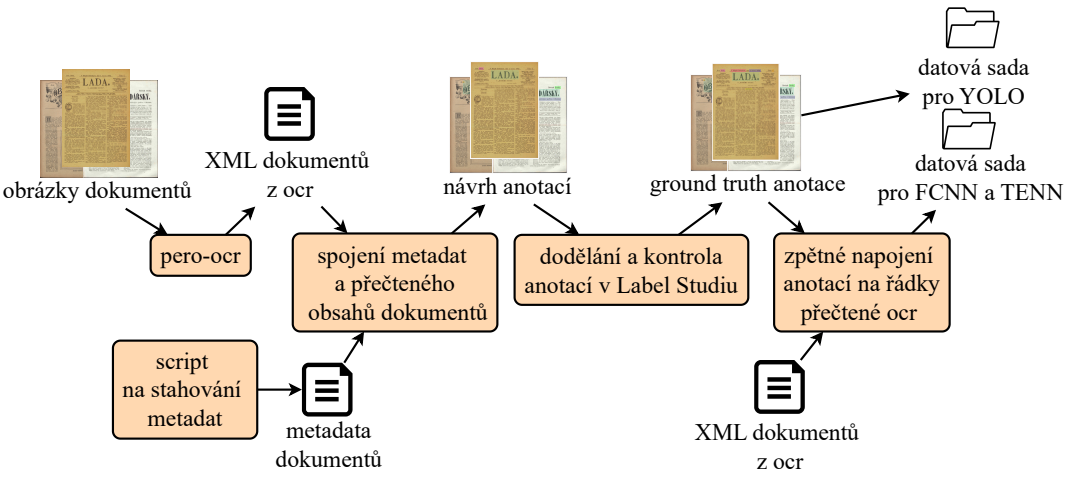
Obě sady mají variantu pro YOLO a pro FCNN a TENN. Proces tvoření (4.1) datové sady začal získáním metadat a obrázků dokumentů. Tyto obrázky byly přečteny pomocí PERO-OCR [12], poté byly výsledky výstupu OCR porovnány s metadaty. Při úspěšném nalezení jedné z hledané třídy metadat v přepisu OCR byl vytvořen návrh anotace, tedy bounding box řádku nebo řádků, na kterém se tento napojený text nacházel, s příslušnou třídou. Takto byly vytvořeny návrhy anotací, které byly importován do Label Studia [23]. Navržené anotace byly zkontrolovány a upraveny, aby přesněji seděli na oblast s výskytem metadat. Zároveň byly přidány anotace, které se nepovedlo automaticky spojit, ať už to bylo špatným přečtením OCR nebo chybějícím údajem v metadatech. V této fázi je datová sada pro YOLO hotová a je potřeba dodělat sadu pro FCNN a TENN. Toho docílíme procesem zpětného napojení anotací na řádky, popsaného v 4.4.

4.1 Datová sada knih

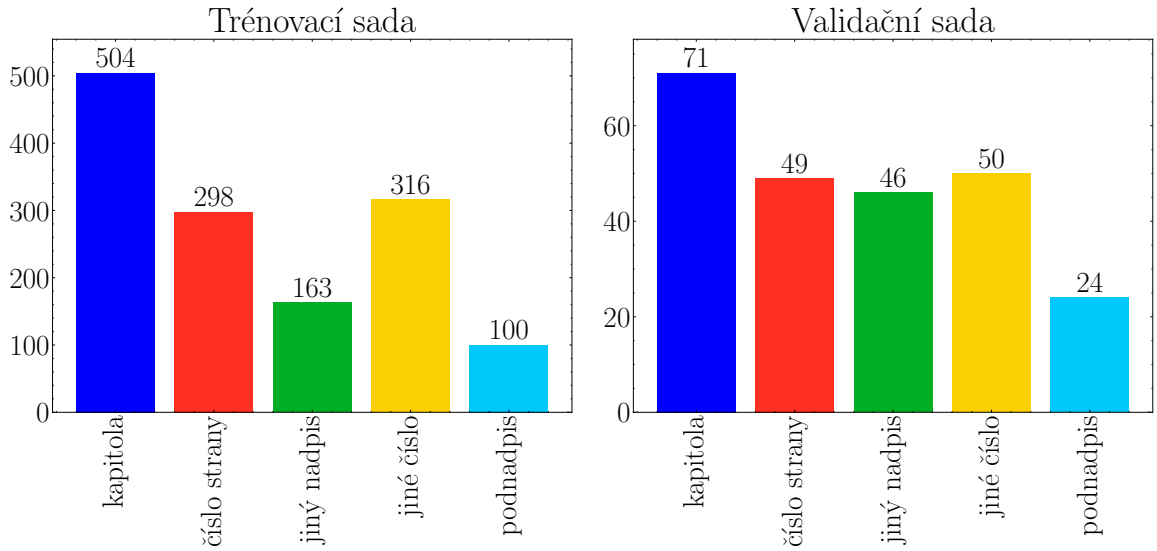
Datová sada se skládá z 671 stran z 10 knih a třídy datasetu se dělí následovně:

- kapitola
- číslo strany
- jiný nadpis – významnější část textu, která není kapitola ani podnadpis
- jiné číslo – číslo, které není očíslování strany, často označení sekce či podsekce
- podnadpis – část textu pod (výjimečně nad) kapitolou nebo jiným nadpisem, která jasně ještě souvisí s danou kapitolou či jiným nadpisem

[1] https://www.mzk.cz/o-knihovne/vyzkum-projekty/naki-iii/smart-digilinka
[2] https://text.nkp.cz/o-knihovne/zakladni-informace/zakladni-dokumenty/vizitky/odbor-osdf8



Obrázek 4.1: Obrázek popisující proces vytváření datových sad.



Obrázek 4.2: Rozložení tříd datové sady knih. Jedná se počet o ground truth anotací s metadaty dané třídy.

4.2 Datová sada periodik

Dataset je složen z titulní stran periodik. Strany byly získány z veřejně dostupných děl z více jak 10 knihoven Digitální knihovny[3]. Vybírány byly periodika s frekvencí vydávaní alespoň jednou za 30 dní. Tento výběr periodik byl hlavně kvůli zaměření se na noviny, týdeníky, měsíčníky a podobné periodika. Důvod byl, aby byl výskyt alespoň nějaké skupiny metadat co nejkonzistentnější, což například u ročenek a podobných často chybělo. Na rozdíl od datové sady knih je tato mnohem rozmanitější, z jednoho periodika jsou v datová sadu maximálně dvě strany. Je také komplexnější, má dvakrát více tříd jak dataset knih. Třídy datasetu jsou následující:

- titulek – název periodika
- podtitulek – dodatek k titulku, např.: „týdenník věnovaný veřejným otázkám.“
- číslo vydání
- datum čísla – datum, kdy bylo číslo vydáno
- ročník – číslo ročníku daného vydání
- datum ročníku – rok, ve kterém byl ročník vydán
- místo vydání
- redaktor
- nakladatel
- vydavatel

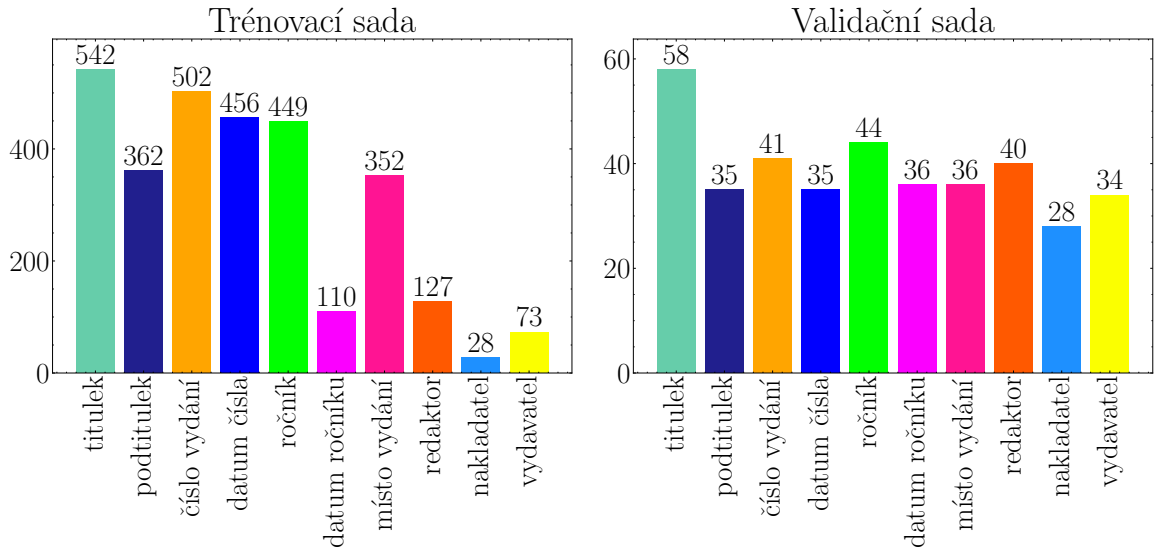
Rozdíly mezi nakladatelem a vydavatelem se, jak si můžeme přečíst například v bakalářské práci Kateřiny Sýkorové [22], v minulosti měnily a vzhledem k definici nakladatele[4] a vydavatele[5] byly tyto třídy anotovány pouze, když bylo z obrázku jasné, o kterou třídu se jedná, tedy, když bylo u jména například „Nakladatelem:“, „Vydává:“ atd. Jméno tedy anotováno nebylo, když to, že daná osoba, organizace či firma má příslušnou funkci vyplývalo z okolního textu. Stejný přístup se aplikoval u redaktora. Pokud je u jednoho jména více označení funkce například „Nakladatel, vydavatel a redaktor:“ je toto jméno anotováno všemi přítomnými třídami.

Dataset je sice rozmanitější a má více tříd, ale pořád je poměrně malý a to dělá problém zejména u tříd, které se nevyskytují tak často. Toho si můžeme všimnout velkých rozdílů výsledků u experimentů YOLO 5.2 například na třídách redaktor a nakladatel. Tyto třídy si jsou velice podobné – obě osoby nebo organizace na obrázku vyznačené svým příslušných slovem („redaktor“, „nakladatel“). Nicméně v trénovací sadě je anotací redaktorů 127, zatím co nakladatelů je 28.

[3] https://www.digitalniknihovna.cz/

[4] Fyzická nebo právnická osoba odpovědná za uvedení neperiodické publikace (knihy, hudebniny, reprodukce, mapy, fotografie apod.) na knižní trh. Zajišťuje hmotné prostředky k vydávání publikací, pečuje o odbornou a výtvarnou stránku vydávaných publikací, zajišťuje jejich vytištění, někdy též distribuci. [17]

[5] Fyzická či právnická osoba odpovědná za přípravu neperiodické či periodické publikace k vydání a oprávněná k vydání této publikace (tj. její zveřejnění tiskem). [17]



Obrázek 4.3: Rozložení tříd datové sady periodik. Jedná se počet o ground truth anotací s metadaty dané třídy.

4.3 Vytvoření návrhu anotací na základě existujících metadat

Cílem tohoto procesu je pouze usnadnit následnou práci v Label Studiu, a tak nebylo nutné, aby tyto anotace byly přesné. Průběh spojování metadat a řádků se liší podle třídy, ale ve většině případů jde o porovnání textu získaného z pole dané třídy v metadatech s řádkem nebo řádky pomocí Levenshteinovy vzdálenosti [8]. V procesu spojování není tato vzdálenost jediná metrika, další jsou podle tříd:

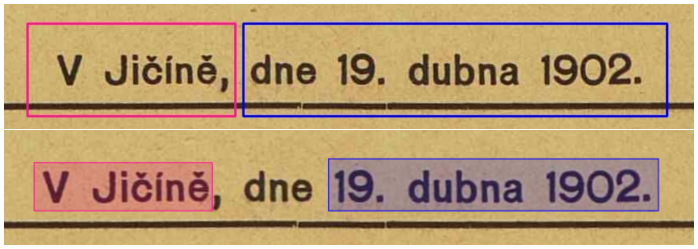
- *kapitola, titulek* – zde se upřednostňují řádky s větší výškou řádku
- *ročník* – přednost se dává řádkům, které jsou pozičně výše na stránce a mají v textu jakýkoliv tvar slova „ročník“
- *datum vydání* – porovnávání probíhá po převedení z textové do číselné podoby data

Návrh anotace je vždy celý jeden nebo více řádků, ale oblast, kde se nachází metadata, nemusí zabírat celou oblast řádku. Proto bylo u tříd *místo vydání* a *datum vydání* (třídy, které nejčastěji nezabírají celý řádek a ve většině případů jsou spolu na stejném řádku) uděláno rozdělení oblasti řádku v poměru 1 : 2 (místo vydání : datum vydání), což zhruba odpovídá poměru, ve kterém se ty třídy na řádku nachází, jak můžeme vidět na ukázce 4.5.

Metadata a jejich struktury. Existuje široká škála typů pro popis metadat dokumentů. Tato práce se zaměřuje na metadatový standard známý jako Metadata Object Description Schema, zkráceně **MODS**[6], který je založený na strukturovaném formátu **XML**[7]. Druhy metadat, na které se tato práce zaměřuje, byly vybrány po domluvě s knihovníky a vedoucím práce. Stejně jak datové sady se i použitá metadata dělí na dva odlišné typy metadat.

[6] https://www.loc.gov/standards/mods/
[7] https://www.w3.org/TR/xml/

Obrázek 4.4: Obrázky s ukázkami částí stran s anotacemi tříd metadat z obou datových sad. Vrchní obrázek je z datové sady knih a spodní z datové sady periodik.



Obrázek 4.5: Ukázka rozdělení řádku s více třídami. Ve vrchním obrázku je vidět návrh anotací pro řádek a ve spodním jsou již upravené ground truth anotace.

Metadata knih. Metadata jedné knihy jsou v všechna v jednom souboru. Tento soubor obsuje kromě schéma MODS i METS[8]. U tohoto typu metadat se zajímáme o *kapitolu*, *číslo strany*, *jiný nadpis*, *jiné číslo* a *podnadpis*. V zpracovávaných metadatech se nachází pouze *kapitola* a *číslo strany*. Jelikož jsou všechna metadata v jednom souboru je potřeba namapovat jednotlivá metadata na soubory se skeny stran, na kterých se nachází. K tomuto se využívá již zmíněné schéma METS.

Metadata periodik. U periodik se zajímáme o *titulek*, *podtitulek*, *číslo vydání*,*datum čísla*, *ročník*, *datum ročníku*, *místo vydání*, *redaktor*, *nakladatel* a *vydavatel*. Souborová struktura metadat periodik je poměrně komplikovanější, jelikož má každé periodikum hierarchické rozdělení na následující 4 úrovně metadat:

```
periodikum ..... např. Národní noviny
├── ročník periodika ..... např. 1849 (druhý ročník)
│   ├── číslo (vydání) periodika ..... např. páté číslo
│   │   ├── stránka periodika ..... např. titulní strana
│   │   └── ...
│   └── ...
└── ...
```

V těchto úrovních můžeme nalézt následující údaje:

- metadata periodika – název periodika, alternativní název, podtitulek, redaktor, vydavatel, nakladatel, místo vydání, téma, frekvence vydávání, jazyk atd.
- metadata ročníku – číslo ročníku, rok ročníku
- metadata čísla – číslo a datum vydání
- metadata stany – číslo strany, typ strany (titulní nebo normální)

Údaje se občas opakují v různých vrstvách metadat, obzvlášť, když se měnil například redaktor. Poté může být více redaktorů na úrovni metadat periodika, obvykle s časovým rozmezím působnosti, a poté na úrovni ročníku nebo čísla.

4.4 Zpětné napojení anotací na řádky a výroba datové sady

Po vytvoření anotací v Label Studiu je, abychom získali datovou sado pro FCNN a TENN, potřeba tyto anotace napojit zpět na řádky, které byly získány z OCR. Toto je nutné,

[8] https://www.loc.gov/standards/mets/

protože jsou neuronové sítě FCNN a TENN závislé na výstupu OCR, tedy klasifikují řádky, ne jakékoliv oblasti na dokumentu, což jsou výstupní anotace z Label Studia.

Při tomto procesu narážíme na jeden z problémů tohoto přístupu. Anotace totiž můžou zahrnovat kousek řádku, celý řádek nebo i více řádků naráz a tak nelze jenom zkontrolovat jestli okraje řádku jsou v boxu anotace. Řešením bylo napojování anotace na řádek, když prochází *baseline*[9] řádku boxem anotace. K řádku tedy bude přidána třída v případě, že platí:

- y pozice *baseline* je v rozmezí y pozic boxu anotace
- jeden z:
  - alespoň jeden z krajních bodů *baseline* je v boxu anotace
  - levý okraj *baseline* je nalevo od boxu anotace a pravý okraj *baseline* je napravo od boxu anotace

Řádek lze takto označit jednou nebo více třídami. Tento přístup může způsobit nechtěné napojení řádků, které prochází boxem anotace pouze okrajově. Na druhou stranu velký počet tříd (*číslo vydání*, *ročník vydání* atd.) se typicky nachází pouze na kraji řádku.

Zároveň nemůžeme napojovat anotace na řádky, které OCR nepřečetlo, tyto anotace se do datové sady vůbec nedostanou. Řádky, ke kterým nebyla napojena žádná anotace, se jsou označeny jako třída *text*.

V následujících tabulkách můžeme vidět kolik anotací se nepodařilo napojit ani na jeden řádek. V naprosté většině je to kvůli nepřečtení OCR, jak už bylo výše zmíněno.

celkem	**kapitola**	**číslo strany**	**jiný nadpis**	**jiné číslo**	**podnadpis**
1556/1621 96 %	562/575 97 %	302/347 87 %	209/209 100 %	363/366 99 %	124/124 100 %

Tabulka 4.1: Tabulka znázorňující úspěšnost zpětné napojení anotací na řádky z OCR pro datovou sadu knih.

celkem	**titulek**	**podtitulek**	**číslo vydání**	**datum čísla**	**ročník**
3341/3392 98 %	576/600 96 %	396/397 99 %	529/543 97 %	487/491 99 %	487/493 99 %
datum ročníku		**místo vydání**	**redaktor**	**nakladatel**	**vydavatel**
144/146 99 %		388/388 100 %	167/167 100 %	55/56 99 %	107/107 100 %

Tabulka 4.2: Tabulka znázorňující úspěšnost zpětné napojení anotací na řádky z OCR pro datovou sadu periodik.

Následně jsou k řádkům připojeny znakové a sémantické příznaky 4.5.

4.5 Reprezentace znakových a sémantických informací

Jelikož není do ani jednoho z použitých systému vkládán text bylo zapotřebí vymyslet způsob, jak text rozumně reprezentovat. V současné době používá velká část architektur

[9] soubor bodů, na kterých řádek „leží“

pro extrakci metadat reprezentační jazykový model BERT [7]. Například v Architektuře LayoutLMv3 [9] je použit na reprezentaci textu embedding slov, který je tvořen modelem RoBERTa [15]. Tento přístup v této práci nebyl využit, kvůli nedostatku anotovaného textu s metadaty, na kterém by se model BERT dal natrénovat. Místo toho byl použit podobný, nicméně zjednodušený, systém. Text je reprezentován z pohledu znaků v textu a z pohledu sémantiky.

Přidání znakových informací. Mnoho z metadatových tříd je v textu reprezentováno svým specifickým způsobem, ten by bylo možné poznat i jenom z textu (bez dalších informací). Z tohoto důvodu byly do vstupních příznaků zakomponovány příznaky znakové. Jeden z nejjasnějších případů je *číslo stran*, kde se téměř nikde nevyskytuje žádný jiný znak než číselný, ale toto platí i pro další třídy jako *titulek*, který velice často bývá napsán všemi velkými písmeny, nebo *jiné číslo*, kde jsou pouze buď číselné znaky nebo velká písmena symbolizující římské číslice.

Jednotlivé příznaky reprezentující znakové informace:

- celkový počet znaků
- počet písmen
- počet velkých písmen
- počet číselných znaků
- počet mezer
- počet zbylých znaků – speciální znaky, čárky, tečky atd.

Tyto informace jsou využívané FCNN a TENN a jsou reprezentovány počtem jednotlivých typů znaků v textu řádku.

Využití Named Entity Recognition pro přidání sémantických informací. Podobně jako můžeme získat velice hodnotnou informaci ze znaků textu lze získat informaci sémantickou. Možná nejlepší ukázky jsou na třídách *redaktor*, *nakladatel* a *vydavatel*. Tyto třídy byli anotovány, jak už bylo vysvětleno v 4.2, pouze v případě výskytu slova pro příslušnou třídu a tedy informace, že se dané slovo v textu nachází, je podstatná, jak můžeme z experimentů v 5.3 vidět.

Pro rozpoznání sémantických informací v textu byly použity dva přístupy:

- První používá Named Entity Recognition (NER) z projektu Czert [21], který byl trénován na českém datasetu Czech Named Entity Corpus (CNEC) [20] a slovanském BSNLP 2019 shared task datasetu [19]. Cílem bylo zlepšit hlavně výsledky pro datovou sadu periodik a tak byly z tříd, které NER detekuje, vybrány časové údaje:
  - časové údaje
  - geografické údaje
  - jména
- Druhý pomocí slovníku se slovy jako „redaktor“, „vydavatelství“ a jejich různými pády a čísly a Levenshteinovy vzdálenosti nebo funkcí detekující římskou číslici určuje následující sémantické třídy:

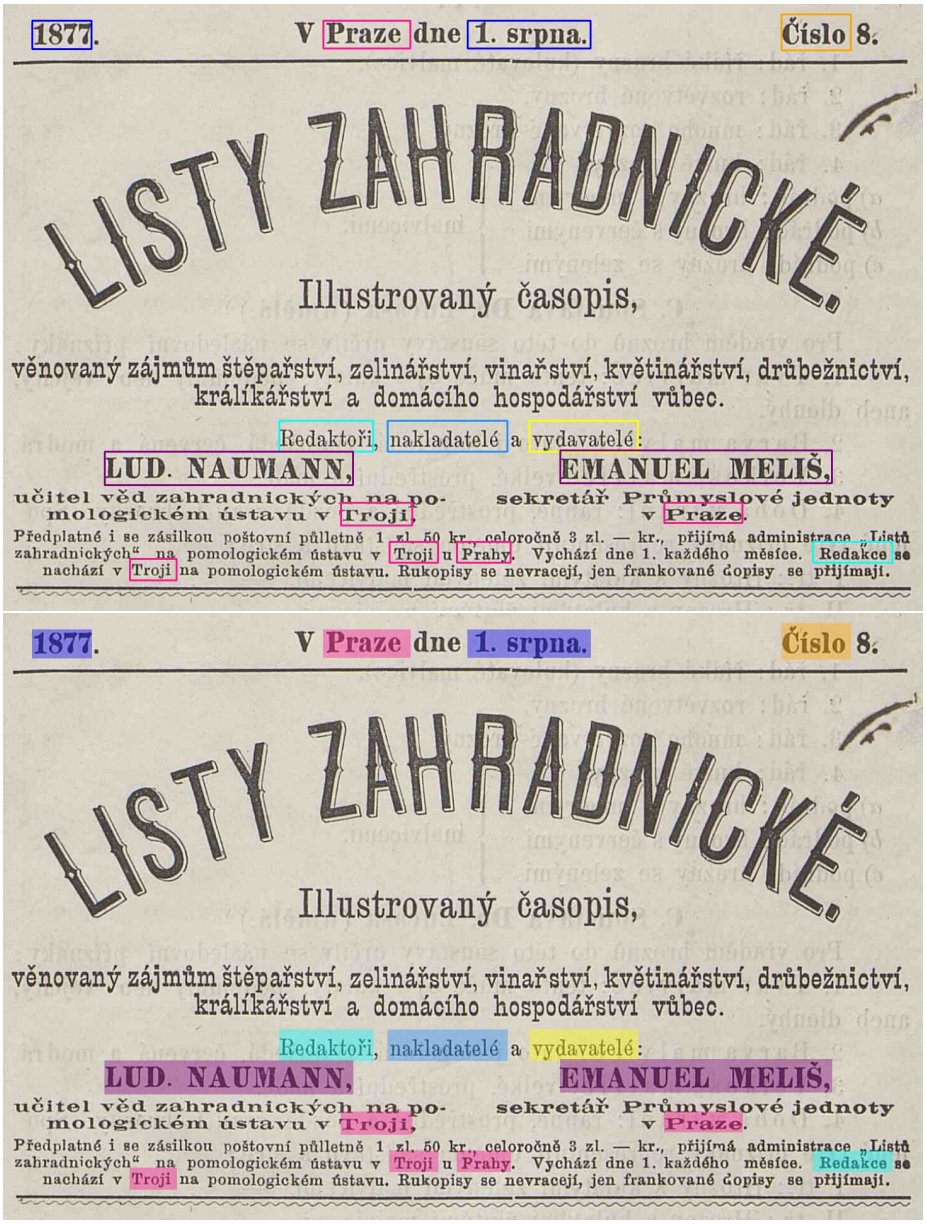
- redaktor – redaktor, redakce, rediguje
- nakladatel – nakladatel, nakladatelství, nakládá
- vydavatel – vydavatel, vydavatelství, vydává
- ročník – ročník
- číslo ročníku – číslo, č., sešit
- římské číslo

Při vytváření tohoto procesu bylo Sémantické informace jsou využívané v FCNN, TENN i YOLO. V případě FCNN a TENN jsou tyto informace reprezentovány počtem těchto sémantických slov. Pro YOLO byla vytvořena technika obarvovaní obrázku, místech, kde se dané slovo nachází.

Obarvování obrázků za účelem reprezentace sémantických informací pro přístup YOLO. Jelikož jako vstup do YOLO jde pouze obrázek, tak bylo potřeba vymyslet způsob, jak tyto sémantické informace reprezentovat. Řešením v této práci je nabarvení těchto sémantických slov. Po zpracování textu modelem NER máme slova označená tokeny[10]. Ty je potřeba zpět namapovat na oblast obrázku. To se děje pomocí `logits`[11]. Vezme se pozice slov (začátek a konec podřetězce označené sémantickým tokenem NER) v `logits` řádku, ta se vynásobí velikostí podvzorkování (vzhledem k pixelové velikosti řádku) `logits`. Výsledeky označuje jak daleko od levé strany řádku slovo začíná a končí. Pro získání x souřadnicím slova se vzdálenosti přičtou k souřadnici x levé strany (začátku) řádku a y souřadnice se nemění. V experimentech 5.9 byly vyzkoušeny dva druhy obarvení. Jeden co pouze obarvil strany oblasti výskytu sémantické informace a druhý, který podbarvil (vyplnil) celou oblast.

[10] označení sémantických tříd (v kontextu využití v této práci)

[11] pole s pravděpodobnostmi výskytu symbolů v podvzorkované oblasti řádku nasekané na vertikální proužky



Obrázek 4.6: Obrázky ukazující barvení sémantických informací na dva způsoby (okrajové a výplňové). Tmavě modře jsou obarveny časové údaje, světle modře nakladatelé, tyrkysově redaktoři, žlutě vydavatelé, oranžově čísla vydání, růžově geografické údaje a fialově jména.

Kapitola 5

Výsledky experimentů

V této kapitole jsou zhodnoceny experimenty na třech použitých přístupech. Experimenty jsou prováděny s různými vstupy a konfiguracemi neuronových sítí.

Příznaky pro experimenty s FCNN a TENN se dělí do následujicích skupin:

- základní – délka a šířka řádku, relativní (k velikosti strany) délka a šířka řádku, čtyři (vrchní, spodní, pravé,levé) odsazení krajů (padding) řádku od stran stránky.
- znakové údaje – délka přepisu řádku, počet písmen, počet velkých písmen, počet číslic, počet mezer a počet ostatních znaků.
- sémantické údaje se lišily podle datové sady:
  - knihy – počet geografických míst, počet časových údajů, počet jmen a počet římských čísel.
  - periodika – počet geografických míst, počet časových údajů, počet jmen, počet římských čísel, dále počet výskytů slov a jejich variant pro *ročník*, počet výskytů slova *číslo*, počty výskytů slov *vydavatel*, *nakladatel* a *redaktor*.

V experimentech se zmiňuje počet sousedních řádků, ten udává kolik se vezme nejbližších řádků z každé strany. Celkový počet řádků lze vyjádřit rovnicí:

$$\text{Celkový počet řádků} = 1 + 2 \cdot n, \tag{5.1}$$

kde n je počet sousedních řádků.

Třída	**Tréninková sada**	**Validační sada**
kapitola	553	84
číslo strany	262	42
jiný nadpis	170	49
jiné číslo	311	49
podnadpis	153	30
obyčejný text	553	84

Tabulka 5.1: Rozdělení počtů řádků v tréninkové a validační sadě knih pro FCNN a TENN.

Třída	Tréninková sada	Validační sada
titulek	1400	126
podtitulek	533	62
číslo vydání	490	41
datum čísla	456	36
číslo ročníku	445	44
datum ročníku	108	38
místo vydání	363	37
redaktor	127	40
nakladatel	31	32
vydavatel	79	39
obyčejný text	1400	126

Tabulka 5.2: Rozdělení počtů řádků v tréninkové a validační sadě periodik pro FCNN a TENN.

Anotace datasetu je nutné před použitím na FCNN a TENN navázat na řádky přečtené OCR, jak je popsáno v 4.4. Liší se tedy od datasetů používaným YOLO modelem. Výsledné rozložení počtu řádků na třídy je zobrazeno v tabulce 5.1 pro knihy a 5.2 pro periodika. Obyčejné řádky jsou ty, na které nebyla navázana žádna anotace.

Rozložení YOLO datasetů jsou přímo shodné s grafy ground truth anotací v obrázku 4.2 pro knihy a 4.3 pro periodika.

5.1 Experimenty s plně propojenou neuronovou sítí (FCNN)

V této části kapitoly prezentujeme výsledky experimentů realizovaných s využitím plně propojené neuronové sítě (FCNN). Trénování bylo provedeno s následujícími parametry:

- **Počet epoch:** 500
- **Počáteční learning rate:** 0.5
- **Decay rate:** 0.5
- **Decay step:** 100
- **Začátek learning rate decay:** 100
- **Velikost batch:** 256

Tyto parametry byly zvoleny na základě předběžných testů, které naznačily optimální konvergence modelu v kontextu dané úlohy.

Datová sada knihy

Vstupy	Precision	Recall	F1 Skóre
základní	**0,91**	0,43	0,51
se znakovými údaji	0,89	0,65	**0,74**
se znakovými a sémantickými údaji	0,86	**0,66**	0,73

Datová sada periodik

Vstupy	Precision	Recall	F1 Skóre
základní	0,38	0,14	0,18
se znakovými údaji	0,77	0,28	0,37
se znakovými a sémantickými údaji	**0,95**	**0,52**	**0,63**

Tabulka 5.3: Výsledky experimentů pro FCNN s různými vstupy na obou datových sadách.

V tabulce 5.3 je možno sledovat jaké zlepšení přinesly dodatečné příznaky. U experimentů na datové sadě knih je vidět, že zatím, co znakové údaje výrazně zvýšily recall a F1 skóre, po přidání sémantických údajů se výsledky nijak výrazně nezměnily. To bylo očekávaným výsledkem, jelikož počet sémantický informací pro datovou sadu knih je výrazně menší, než počet pro periodika. Zároveň jsou tyto informace méně sémanticky užitečné jak u periodik, protože žádná třídy konzistentně neobsahuje jména, místy nebo časovými údaji.

F1 skóre u periodik s pouze základními vstupy bylo pouze 0,18. Při přidání znakových údajů se F1 skóre více jak zdvojnásobilo a to se po přidání údajů sémantických opět téměř zdvojnásobilo. Zajímavé je, že i přes větší komplexitu datové sady, se precision modelu na periodikách díky sémantickým informace dostalo až na 0,95 a tím překonalo i nejvyšší hodnotu precision dosaženou na datové sadě knih.

Datová sada knihy

Počet sousedních řádků	Precision	Recall	F1 Skóre
0	0,86	0,66	0,73
2	**0,93**	0,77	**0,83**
4	0,87	0,75	0,80
8	0,87	**0,78**	0,81

Datová sada periodik

Počet sousedních řádků	Precision	Recall	F1 Skóre
0	**0,95**	0,52	0,63
2	0,89	**0,72**	**0,78**
4	0,89	0,61	0,69
8	0,85	0,59	0,67

Tabulka 5.4: Výsledky experimentů pro FCNN se všemi vstupy s různými počty sousedních řádků.

Další z experimentů spočíval v testování jestli připojení příznakových vektorů okolních řádků k příznakovému vektoru řádku, který je právě klasifikován, bude mít vliv na výkonost sítí. Je nutné zdůraznit, že klasifikován je pořád pouze jeden řádek, nezávisle na počtu připojených řádků. Celkový počet řádků vzhledem k počtu sousedních řádků je vysvětlen na začátku této kapitoly 5. Výsledky jsou zobrazeny v tabulce 5.4. Pro datovou sadu knih se recall zvyšoval s postupným přidáváním řádků, precision se ale po počátečním zvýšení

u dvou sousedních řádků snížil na hodnotu podobnou hodnotu jak bez přidaných řádků. U periodik bylo zaznamenáno nejlepší zlepšení při přidání dvou sousedních řádků, kde se výrazně zvýšil recall za cenu menšího zhoršení precision. Z experimentů na obou datových sadách vyplývá, že z hlediska F1 skóre je nejlepší konfigurace se všemi příznaky a dvěma sousedními řádky.

Datová sada knihy – (všechny vstupy, 2 sousední řádky)

Třída	**Precision**	**Recall**	**F1 Skóre**
celkem	0,93	0,77	0,83
kapitola	0,91	0,92	0,91
číslo strany	**1**	**0,98**	**0,99**
jiný nadpis	0,86	0,49	0,62
jiné číslo	**1**	0,84	0,91
podnadpis	0,93	0,47	0,62
obyčejný text	0,9	0,96	0,93

Datová sada periodik – (všechny vstupy, 2 sousední řádky)

Třída	**Precision**	**Recall**	**F1 Skóre**
celkem	0,89	0,72	0,78
titulek	0,91	0,64	0,75
podtitulek	0,91	0,79	0,85
číslo vydání	0,92	0,81	0,86
datum čísla	0,81	0,81	0,81
číslo ročníku	0,93	**0,84**	**0,88**
datum ročníku	**1**	0,47	0,64
místo vydání	0,86	0,65	0,74
redaktor	0,8	0,5	0,62
nakladatel	**1**	0,28	0,44
vydavatel	0,92	0,59	0,72
obyčejný text	0,92	0,89	0,9

Tabulka 5.5: Výsledky nejlepších přístupů na obou datových sadách pro FCNN na základě experimentů.

Celkově byly tyto FCNN sítě relativně úspěšné (při porovnání s ostatními třídami) v klasifikaci tříd, které obecně obsahují hodně číselných znaků. Tedy třídy jako *číslo strany*, *jiné číslo*, *číslo vydání* a *číslo ročníku*. Zdaleka nejlepší F1 skóre (0,97-0,99) má pro dataset knih při jakékoliv kombinaci vstupů třída *číslo strany*. To je dle mého názoru díky jeho konzistentní podobě a umístění. Číslo strany je ve všech případech svého výskytu na obrázku umístěno na vrchním či spodním okraji strany, je v poměru s ostatním textem malé a v téměř všech případech obsahuje pouze číselné znaky. U datové sady periodik je to třída *číslo ročníku* s F1 skórem v rozmezí od 0,84 do 0.91.

5.2 Experimenty se sítí Transformer Encoder (TENN)

Oproti FCNN klasifikuje TENN všechny řádky, které dostane. Pokud tedy dostane 8 řádků na vstup, bude na výstupu 8 klasifikačních vektorů, každý s hodnotou pravděpodobnosti výskytu pro každou třídu. V této části kapitoly prezentuji výsledky experimentů reali-

zovaných s využitím plně Transformer Encoder neuronové sítě (TENN). Trénování bylo provedeno s následujícími parametry:

- **Počet epoch:** 1000
- **Počáteční learning rate:** 0.05
- **Decay rate:** 0.8
- **Decay step:** 100
- **Začátek learning rate decay:** 500
- **Velikost batch:** 256

Tyto parametry zvoleny na základě předběžných testů.

Datová sada knihy

Poziční kódování	Vstupy	Precision	Recall	F1 Skóre
1D sekvenční	bez paddingu	0,82	0,41	0,52
1D sekvenční	všechny	0,81	0,4	0,51
1D pořadí na stránce	bez paddingu	0,82	0,44	0,55
1D pořadí na stránce	všechny	0,82	0,71	0,75
2D	bez paddingu	0,81	**0,76**	0,78
2D	všechny	**0,84**	**0,76**	**0,8**
1D sekvenční + 2D	bez paddingu	0,81	0,73	0,76
1D sekvenční + 2D	všechny	0,81	0,71	0,75

Datová sada periodik

Poziční kódování	Vstupy	Precision	Recall	F1 Skóre
1D sekvenční	bez paddingu	0.6	0,35	0,44
1D sekvenční	všechny	0,61	0,39	0,48
1D pořadí na stránce	bez paddingu	0,72	0,43	0,52
1D pořadí na stránce	všechny	0,67	**0,48**	0,55
2D	bez paddingu	0,57	0,44	0,48
2D	všechny	**0,76**	0,45	**0,56**
1D sekvenční + 2D	bez paddingu	0,45	0,25	0,31
1D sekvenční + 2D	všechny	0,5	0,31	0,36

Tabulka 5.6: Výsledky experimentů pro TENN s různými vstupy a pozičními kódováními, na vstup dostává síť 1 (vybraný načítačem dat) + 8 sousedních řádků.

V tabulce 5.6 je zobrazeno jaké zlepšení přinesly různé poziční kódování. 1D sekvenční pracuje na bázi pozice ve vstupní sekvenci. 1D pořadí na stránce kóduje pozici na stránce v pořadí jako ho přečetlo OCR, většinou pořadí čtení. 2D zakódovává (x, y) pozici, kde (x, y) reprezentuje středový bod *baseline*[1]. Podrobněji jsou tyto kódování vysvětlena na 2.2. Byl ještě vyzkoušen přístup kombinace 1D sekvenčního a 2D kódování, který byl realizován součtem těchto dvou kódování. Pro každé poziční kódování byl vyzkoušen vstup se všemi příznakovými vektory a vstup bez paddingu (čtyři příznaky, každý reprezentující vzdálenost

[1]soubor bodů, na kterých řádek „leží“

okraje oblasti řádku od své strany). Záměrem bylo sledovat, jak efektivní jsou různé typy kódování na těchto dvou použitých datových sadách. Také šlo o sledování jestli padding, který může být považován za poziční informaci, ovlivní tyto experimenty. Tedy jestli není zbytečný, když je v příznakovém vektoru zakódovaná poziční informace. U výsledků většiny modelů, které neměly padding, lze pozorovat malý pokles F1 skóre oproti modelům, které ho měly. Padding tedy zůstává, i u modelů, kde je přítomné poziční kódovaní. Druhou částí těchto experimentů je porovnání mezi jednotlivými druhy pozičního kódování. U datasetu knih změna pozičního kódování neměla téměř žádný vliv na precision, ale u recall jsou vidět výrazné rozdíly. Nejlépe si vedly experimenty s 2D pozičním kódováním, a to jak u precision, tak i u recall. Výsledky u periodik se poměrně značně lišili v precision, menší rozdíly byly zaznamenány i u recall. U obou datových sad se osvědčilo 2D poziční kódování, nicméně 1D kódovaní vzhledem k pořadí na stránce nebylo výsledky u obou datových sad o moc horší.

Datová sada knihy

Počet sousedních řádků	Precision	Recall	F1 Skóre
8	0,84	0,76	0,8
16	**0,92**	**0,8**	**0,84**
24	0,87	0,75	0,80

Datová sada periodik

Počet sousedních řádků	Precision	Recall	F1 Skóre
8	**0,76**	0,45	**0,56**
16	0,64	**0,5**	0,55
24	0,75	0,49	**0,59**

Tabulka 5.7: Výsledky experimentů pro TENN se všemi vstupy s různými počty sousedních řádků.

Cílem tohoto dalšího experimentu je sledovat vliv počtu vkládaných řádků na vstup TENN. Z předešlého experimentu se vybralo poziční kódování s nejlepšími výsledky. S tímto kódováním byly udělány experimenty s 8, 16 a 24 sousedními řádky. Jak bylo zmíněno na začátku kapitoly 5, celkový počet vstupních řádků je $1 + 2 \cdot n$, kde n je počet sousedních řádků.

Datová sada knihy – (2D poziční kódování, 16 sousedních řádků)

Třída	Precision	Recall	F1 Skóre
celkem	0,92	0,8	0,84
kapitola	0,93	0,86	0,9
číslo strany	**1**	**0,95**	**0,98**
jiný nadpis	0,98	0,85	0,91
jiné číslo	0,98	0,92	0,95
podnadpis	0,6	0,23	0,33
obyčejný text	1	1	1

Datová sada periodik – (2D poziční kódování, 24 sousedních řádků)

Třída	Precision	Recall	F1 Skóre
celkem	0,75	0,49	0,59
titulek	0,95	**0,97**	**0,96**
podtitulek	0,42	0,4	0,41
číslo vydání	0,92	0,86	0,89
datum čísla	**1**	0,71	0,83
číslo ročník	0,68	0,68	0,68
datum ročníku	**1**	0,07	0,13
místo vydání	0,5	0,47	0,48
redaktor	0,5	0,06	0,11
nakladatel	0,67	0,14	0,24
vydavatel	0,67	0,11	0,18
obyčejný text	1	1	1

Tabulka 5.8: Výsledky nejlepších přístupů na obou datových sadách pro TENN na základě experimentů.

V tabulce 5.8 jsou výsledky jednotlivých tříd pro modely, které pro datovou sadu přinášely nejlepší výsledky. Při porovnání s FCNN si TENN model na datové sadě knih vedl podobně. Má lepší F1 skóre u třídy *jiný nadpis*, ale zase horší u *podnadpis*. U datové sady periodik došlo ke zlepšení u klasifikace třídy *titulek* a při rozpoznání obyčejného textu. Tyto dvě třídy jsou v datasetu nejvíce zastoupená a tedy i přes podobné nebo horší výsledky u ostatních tříd nejsou metriky celkových modelů tak rozdílné. Hlavní důvod dle mého názoru je malá schopnost generalizace[2]. F1 skóre na trénovací sadě dosahoval TENN a FCNN podobných (až na třídy *redaktor*, *nakladatel* a *vydavatel*), zatím co na validační sadě TENN značně zaostával. Myslím si, že k tomuto přispívají dva faktory – nedostatečná velikost datasetu a zbytečná vnitřní velikost Feed-Forward sítě v Transformer Encoder (kde je výchozí dimenzionalita 2048).

5.3 Výsledky experimentů s přístupem využívající YOLOv8 model

Experimenty v této sekci se zabývají různými přístupy při práci s modelem YOLO. Jsou vyzkoušeny obarvení datasetů, nastavení augmentací dat při trénování modelu a testovaní čtení OCR z oblastí detekovaných modelem.

[2]Generalizace je schopnost modelu aplikovat to, co se naučit v čase tréninku, na nová data.

Datová sada knihy

Obarvení datové sady	Precision	Recall	mAP50	mAP50-90
žádné	0,914	**0,878**	**0,922**	**0,67**
okrajové	**0,933**	0,818	0,917	**0,67**
výplňové	0,922	0,814	0,901	0,646

Datová sada periodik

Obarvení datové sady	Precision	Recall	mAP50	mAP50-90
žádné	0,724	0,543	0,609	0,391
okrajové	0,615	0,608	0,623	0,407
výplňové	**0,786**	**0,663**	**0,721**	**0,471**

Tabulka 5.9: Výsledky experimentů na datových sadách knih a periodik s různými obarveními datové sady pro přístup YOLO.

První experiment je prováděn se dvěma různě obarvenými datasety a jedním nenabarveným. Nabarvené jsou na obrázcích sémanticky významná slova, podrobněji v 4.5. V tabulce 5.9 je vidět podobný trend jako u experimentů FCNN 5.3. U datové sady knih přidání okrajového barvení přispívá ke zvýšení precision, ale zase snížení recall, u výplňového dochází ke zhoršení ve všech metrikách až na precision. Nedá se tedy říct, že by obarvení pro datovou sadu knih vedlo k lepším výsledkům. Přidání sémantických informací, v tomto přístupu je to obarvení, vede k významnému zlepšení u datové sady periodik. Toto zlepšení je zejména u tříd jako je *redaktor*, *nakladatel* a *vydavatel*. Výsledky jednotlivých tříd pro datasety s obarvením i bez obarvení na obrázcích 5.1.

Augmentace dat	Obarvení datové sady	Precision	Recall	mAP50	mAP50-90
výchozí	žádné	0,724	0,543	0,609	0,391
výchozí	výplňové	**0,786**	**0,663**	**0,721**	**0,471**
bez augmentací	žádné	0,633	0,293	0,362	0,206
bez augmentací	výplňové	0,581	0,476	0,531	0,295
bez barevných augmentací	žádné	0,556	0,543	0,543	0,345
bez barevných augmentací	výplňové	0,671	0,645	0,683	0,443

Tabulka 5.10: Výsledky experimentů na datové sadě periodik s různými kombinacemi obarvení datové sady a augmentace dat při trénování pro přístup YOLO.

YOLOv8 při trénování používá řadu augmentací dat[3]. Skupina dalších experimentů na datové sadě periodik slouží ke kontrole, jestli některé augmentace nezhoršují výsledky modelů. Hlavním podmětem byly obavy, jestli nemůžou barevné augmentace negativně poznamenat předávání syntaktických informací pomocí obarvování. Součástí experimentů byly tři nastavení augmentací dat. První je nastavení výchozí, druhé je nastavení s vypnutím všech augmentací a třetí je vypnutí pouze augmentace upravující barvy. Jak jde vidět

[3] https://rumn.medium.com/yolo-data-augmentation-explained-turbocharge-your-object-detection-model-94c33278303a, článek vysvětlující augmentace dat využívané při tréninku YOLOv8

z tabulky 5.10 vypnutí augmentací nezlepšilo výsledky. Výchozí nastavení augmentace dat při tréninku je tedy podle těchto experimentů nejlepší.

Obarvení validační sady	Precision	Recall	mAP50	mAP50-90
žádné	0,69	0,602	0,612	0,41
výplňové	0,718	0,654	0,674	0,447
žádné + výplňové	0,708	0,626	0,64	0,423

Tabulka 5.11: Tabulka validačních výsledků experimentu trénování YOLO na kombinovaně nabarvené a neobarvené datové sadě periodik. Žádné + výplňové značí spojení, tak že se od každého obrázku dal obrázek nabarvený i nenabarvený.

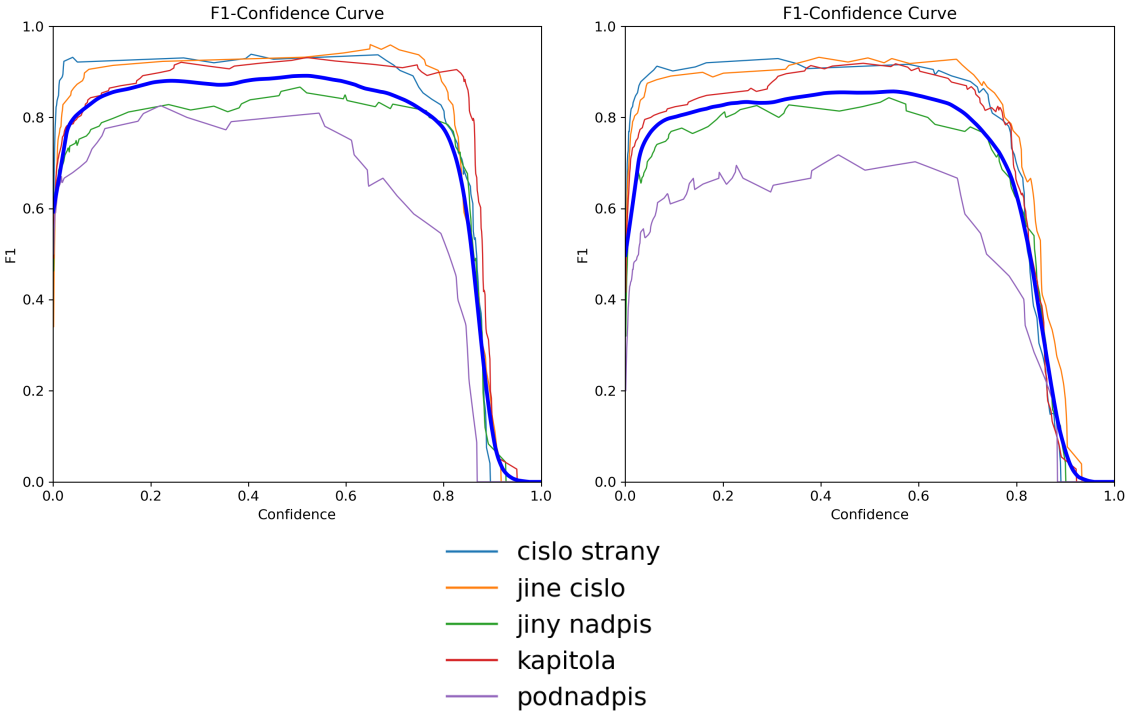
Obarvování sémanticky významných slov se v experimentech 5.9 ukázalo jako efektivní pro zvýšení přesnosti detekcí. Systém obarvování ovšem není perfektní a muže se stát, že některá sémanticky významná slova nenabarví. Další experiment se zabýval otázkou, jestli by šlo natrénovat model pro dataset periodik tak, aby nebyl na obarvení závislý, ale zároveň aby byl schopen obarvení využít. Experiment probíhal tak, že se každý obrázek dal do své příslušné (trénovací nebo validační) sady se dvakrát, jednou nabarvený a jednou nenabarvený. Takto by se teoreticky mohl model na obarvení zvyknout používat, ale stále být schopný detekce v případě, kdy obarvení chybí. Barvení bylo na základě lepší výsledků z experimentu 5.9 vybráno výplňové. V tabulce 5.11 jsou výsledky takto natrénovaného modulu na již rozdělených validačních sadách. Na validační sadě bez nabarvení došlo ke zlepšení u recall a mAP a ke zhoršení v precision oproti trénováním pouze na nenabarveném datasetu z pokusu 5.9. Na obarvené validační sadě došlo k výrazném výraznému poklesu hlavně u metrik precision a mAP50. Při validaci obou sad zároveň reprezentují metriky přibližný průměr výsledků na obarvené a neobarvené sadě. Tento přístup není úplně vhodný pro použitý dataset periodik, ale u potencionální budoucí datové sady, kde se nabarvení sémanticky významných slov nedaří konzistentně by měl být tento přístup otestován.

Datová sada	FP	FN	TP	CER	Přečteno správně	Přečteno špatně
knihy	45	14	212	11,9 %	173 (81,6 %)	39 (18,4 %)
periodika	89	88	265	16,9 %	197 (74,3 %)	68 (25,7 %)

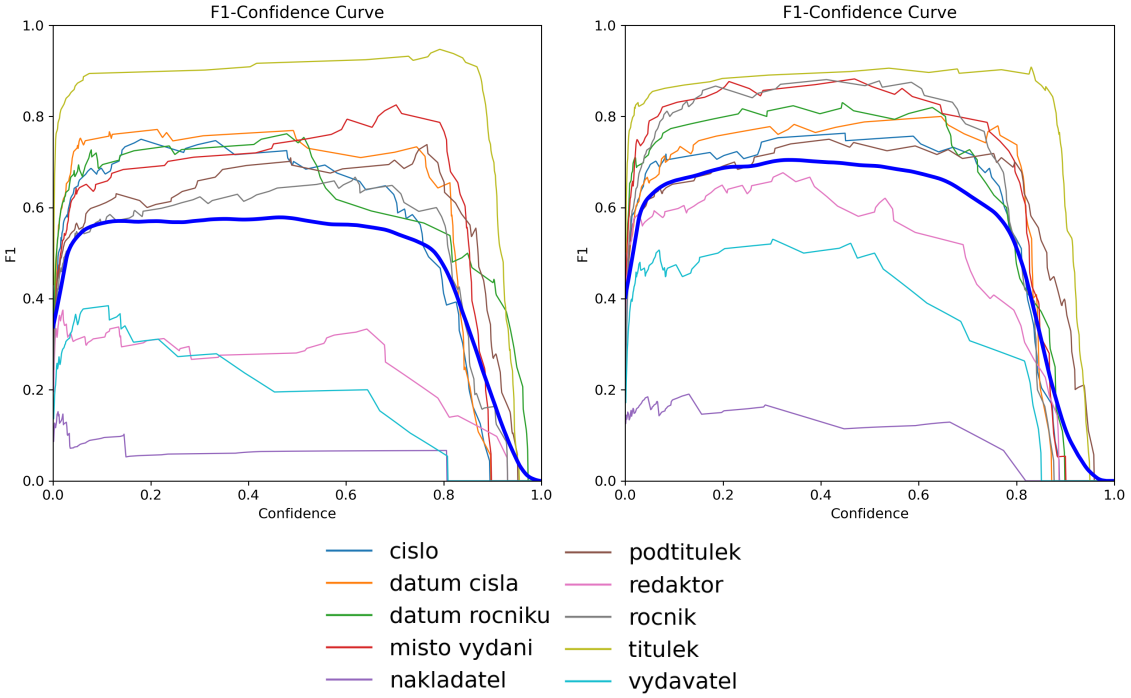
Tabulka 5.12: Výsledky YOLO pipeline s modelem YOLO trénovaným se základními datovými augmentacemi na obarvených (výplň) datový sadách. Za správně přečtou se považuje TP detekce, která po přečtení má CER < 0.2.

Poslední experiment se zaměřuje na schopnost OCR přečíst oblasti označené detekcemi YOLO. Byla vytvořena „YOLO pipeline“, která se skládá z natrénovaného modelu YOLO a OCR systému. Pro účely tohoto experimentu jsem pro všechny validační anotace vytvořil ground truth přepisy, tedy přepsaný text, který se nachází v oblasti anotace. Model poté zpracoval každý obrázek. Oblasti TP detekce, tedy anotace co měli alespoň IoU[4] > 0.6 s ground truth anotací, byly přečteny systémem OCR. Text přečtený OCR byl poté porovnáván s ground truth přepisem. Výsledky jsou v tabulce 5.12. Aby bylo možné říct kolik TP detekcí bylo přečteno (s nějakou tolerancí) správně zvolil jsem si hodnotu CER 0,2. TP detekce byla označena za správně přečtenou, pokud měla hodnotu CER menší než 0,2.

[4]IoU – Intersection over Union neboli poměr plochy průniku ku ploše sjednocení



Obrázek 5.1: Grafy zobrazující F1-Confidence křivky pro datovou sadu knih. Vlevo je základní datová sada a vpravo je obohacená obarvením sémanticky významných slov.



Obrázek 5.2: Grafy zobrazující F1-Confidence křivky pro dataset periodik. Vlevo je základní datová sada a vpravo je obohacená obarvením sémanticky významných slov.

Kapitola 6

Závěr

Cílem práce bylo navrhnout navrhnout a vyvinout metody pro automatické rozpoznávání a extrakci metadata z dokumentů. Cílem bylo získat cenné informace, jako jsou názvy dokumentů, jména autorů, data vydání atd., ze skenů dokumentů. Pro tyto účely jsem vytvořil dva datové sady. Jedna datová sada skládajicí ze skenů knih. Tato sada je ta jednodušší, je poskládaná z 10 knih a metadata se v ní dělí na 5 tříd. Druhá datová sada je složena z periodik a je významně komplexnější a rozmanitější než sada s knihami. Obsahuje přes 700 titulních stránek periodik a metadata jsou rozdělena do 10 tříd.

Detekci metadat provádím dvěma metodami. První metodou klasifikuji řádky přečtené systémem OCR na základě rozměrů, počtu znaků z jednotlivých typů (písmena, čísla atd.) a sémantické informace získané z textu. Pro tuto metodu jsem vytvořil dvě neuronové sítě – plně propojená síť (FCNN) a síť využívající Transformer Encoder (TENN). FCNN dosahuje F1 skóre 0,83 na datasetu knih a 0,78 na datasetu periodik. F1 skóre TENN dosahuje hodnot 0,84 na datasetu knih a 0,59 na datasetu periodik. Důvodem horších výsledků je špatná generalizace TENN.

Další metoda spočívá v detekci objektů přímo na skenech dokumentů pomocí modelu YOLO. YOLO dosahuje F1 skóre 0,86 (confidence na 0,549) na datasetu knih a 0,7 (confidence na 0,336) na datasetu periodik.

Přímé porovnávání metrikami přístupů FCNN a TENN s přístupem využívající YOLO ale není možné, protože jde o přístupy s jinými typy datových sad a s jinými výstupy. V potaz musí být také vzat fakt, že při zpětného napojování (4.4) mohou některé anotace nekorektně napojeny na špatné řádky a tedy FCNN a TENN nepracují se stoprocentní ground truth.

Literatura

[1] Apostol, T. M. *Calculus volume 1 One-variable calculus, with an itroduction to linear algebra.* 2d ed. New York: Wiley, 1967. ISBN 978-0-471-00005-1.

[2] Appalaraju, S.; Jasani, B.; Kota, B. U.; Xie, Y. a Manmatha, R. *DocFormer: End-to-End Transformer for Document Understanding.* 2021.

[3] Baviskar, D.; Ahirrao, S.; Potdar, V. a Kotecha, K. Efficient Automated Processing of the Unstructured Documents Using Artificial Intelligence: A Systematic Literature Review and Future Directions. *IEEE Access*, 2021, sv. 9, s. 72894–72936.

[4] Boukhers, Z.; Beili, N.; Hartmann, T.; Goswami, P. a Zafar, M. A. MexPub: Deep Transfer Learning for Metadata Extraction from German Publications. In: *2021 ACM/IEEE Joint Conference on Digital Libraries (JCDL).* 2021, s. 250–253.

[5] Choi, J.; Kong, H.; Yoon, H.; Oh, H.-S. a Jung, Y. *LAME: Layout Aware Metadata Extraction Approach for Research Articles.* 2021.

[6] Contributors, P. *Binary Cross Entropy torch dokumentace* online. Dostupné z: https://pytorch.org/docs/stable/generated/torch.nn.BCELoss.html. [cit. 2024-30-04].

[7] Devlin, J.; Chang, M.-W.; Lee, K. a Toutanova, K. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* 2019.

[8] Grashchenko, S. Levenshtein Distance Computation. *Baeldung* online. 27. července 2023. Dostupné z: https://www.baeldung.com/cs/levenshtein-distance-computation. [cit. 2024-18-04].

[9] Huang, Y.; Lv, T.; Cui, L.; Lu, Y. a Wei, F. *LayoutLMv3: Pre-training for Document AI with Unified Text and Image Masking.* 2022.

[10] Jocher, G.; Chaurasia, A. a Qiu, J. *Ultralytics YOLO* online. 8.0.0. Leden 2023. Dostupné z: https://github.com/ultralytics/ultralytics. [cit. 2024-18-04].

[11] Jurafsky, D. a Martin, J. H. *Speech and Language Processing.* 3. vyd. 2023. Dostupné z: https://web.stanford.edu/~jurafsky/slp3/.

[12] Kohút, J. a Hradiš, M. TS-Net: OCR Trained to Switch Between Text Transcription Styles. In: *Lecture Notes in Computer Science.* Springer International Publishing, 2021, s. 478–493. ISBN 9783030863371. Dostupné z: http://dx.doi.org/10.1007/978-3-030-86337-1_32.

[13] Li, C.; Bi, B.; Yan, M.; Wang, W.; Huang, S. et al. *StructuralLM: Structural Pre-training for Form Understanding.* 2021.

[14] Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B. et al. *Feature Pyramid Networks for Object Detection*. 2017.

[15] Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M. et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019.

[16] Mikolov, T.; Chen, K.; Corrado, G. a Dean, J. *Efficient Estimation of Word Representations in Vector Space*. 2013.

[17] Národní knihovna České republiky. *Online katalog Národní knihovny ČR* online. 2014. Dostupné z: https://aleph.nkp.cz/. [cit. 2024-20-04].

[18] Pennington, J.; Socher, R. a Manning, C. GloVe: Global Vectors for Word Representation. In: Moschitti, A.; Pang, B. a Daelemans, W., ed. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, říjen 2014, s. 1532–1543. Dostupné z: https://aclanthology.org/D14-1162.

[19] Piskorski, J.; Laskova, L.; Marcińczuk, M.; Pivovarova, L.; Přibáň, P. et al. The Second Cross-Lingual Challenge on Recognition, Normalization, Classification, and Linking of Named Entities across Slavic Languages. In: *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*. Florence, Italy: Association for Computational Linguistics, Srpen 2019, s. 63–74. Dostupné z: https://www.aclweb.org/anthology/W19-3709.

[20] Ševčíková, M.; Žabokrtský, Z. a Krůza, O. Named Entities in Czech: Annotating Data and Developing NE Tagger. In: Matoušek, V. a Mautner, P., ed. *Text, Speech and Dialogue*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, s. 188–195. ISBN 978-3-540-74628-7.

[21] Sido, J.; Pražák, O.; Přibáň, P.; Pašek, J.; Seják, M. et al. *Czert – Czech BERT-like Model for Language Representation*. 2021.

[22] Sýkorová, K. *Vydavatelství a nakladatelství – rozdíly a shody v předlistopadovém a polistopadovém období*. Praha, CZ, 2015. 54 s. Bakalářská práce. Univerzita Karlova, Fakulta sociálních věd, Institut komunikačních studií a žurnalistiky. Katedra žurnalistiky. Dostupné z: https://dspace.cuni.cz/bitstream/handle/20.500.11956/64684/BPTX_2012_2_11230_0_357016_0_138809.pdf. [cit. 2024-19-04].

[23] Tkachenko, M.; Malyuk, M.; Holmanyuk, A. a Liubimov, N. *Label Studio: Data labeling software* online. 2020-2022. Dostupné z: https://github.com/heartexlabs/label-studio. [cit. 2024-19-04].

[24] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L. et al. *Attention Is All You Need*. 2023.

[25] Xu, J.; Li, Z.; Du, B.; Zhang, M. a Liu, J. Reluplex made more practical: Leaky ReLU. In: *2020 IEEE Symposium on Computers and Communications (ISCC)*. 2020, s. 1–7.

Příloha A

Obsah přiloženého paměťového média

```
./
├── latex/ ..... Zdrojový tvar bakalářské práce
├── digilinka/ ..... Zdrojové soubory kódu
│   ├── dataset/ ..... Zdrojové soubory na přípravu datových sad
│   ├── pero-orc/ ..... Použitý OCR systém
│   └── nets/ ..... Definice modelů a pomocných funkcí
├── examples/ ..... Ukázky výstupů
├── BP.pdf ..... Bakalářská práce ve formátu PDF
└── video.mp4 ..... Video prezentující bakalářskou práci
```