Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky

Detekce osobních údajů pomocí metod hlubokého učení v nestrukturovaném textu

DIPLOMOVÁ PRÁCE

Studijní program: Informační systémy a technologie

Specializace: Vývoj informačních systémů

Autor: Bc. David Ondrášek

Vedoucí diplomové práce: Ing. Josef Doležal

Praha, červen 2022

Poděkování

Rád bych poděkoval panu Ing. Josefu Doležalovi za vedení diplomové práce a za jeho trefné připomínky, které mi během zpracování diplomové práce poskytl. Dále bych rád poděkoval panu Mgr. Tomáši Lechnerovi, Ph.D. za zpětnou vazbu a za poskytnutí zdrojů informací týkajících se právní stránky této diplomové práce.

Abstrakt

Tato práce se zabývá problematikou anonymizace osobních údajů v nestrukturovaných textech, konkrétněji potom doménově specifickou detekcí a klasifikací osobních údajů v různých typech vstupních dokumentů.

Cílem této diplomové práce je navrhnout a implementovat prototyp modulárního anonymizačního nástroje, který je možné jednoduše programaticky upravit pro klasifikaci jmenných entit v různých doménově specifických typech vstupních dokumentů ve formátu nestrukturovaného textu.

Při implementaci anonymizačního nástroje je využito technik hlubokého učení a dojde pomocí frameworku spaCy k vytrénování vlastního multilingvního Named Entity Recognition modelu, který je následně integrován do samotného anonymizačního nástroje, vyvinutého pomocí SDK Presidio. Nástroj je upraven pro konkrétní doménově specifickou aplikaci klasifikace osobních údajů v dokumentech nahrávaných do Veřejného registru smluv.

Klíčová slova

Hluboké učení, Natural Language Processing, Ochrana osobních údajů, Named Entity Recognition. spaCy, Presidio, Multilingvní modely

JEL klasifikace

C45 Neural Networks and Related Topics

Abstract

This diploma thesis deals with a PII (Personally identifiable information) protection in unstructured texts, specifically domain-specific detection and classification of PIIs in various types of input documents.

The main goal of this diploma thesis is to design and implement the functional prototype of modular PII anonymizer tool, which can be easily programmatically adjusted to classify named entities in various types of input documents.

The PII anonymizer tool utilizes the strength of Deep Learning techniques. Using the spaCy framework, new custom Entity Named Recognition model is trained and then integrated into the PII anonymizer tool, which is built with help of the Presidio SDK. PII anonymizer tool is then adjusted for a unique, domain-specific task, which is a PII classification within the documents subjecting to obligatory upload to „Veřejný registr smluv“ (Public registry of contracts).

Keywords

Deep Learning, Natural Language Processing, PII protection, Named Entity Recognition. spaCy, Presidio, Multilingual models

JEL Classification

C45 Neural Networks and Related Topics

Obsah

Úvod 10
1.1 Vymezení problému 11
1.1.1 Absence jednoznačné definice osobního údaje 11
1.1.2 Limitující závislost na jazyce trénovacích datasetů 11
1.1.3 Nemodulární implementace existujících nástrojů 12
1.2 Účel a cíle práce 12
1.3 Omezení projektu 13
1.4 Význam a přínos práce 13
2 Rešerše 15
2.1 Rešeršní strategie 15
2.2 Absence jednoznačné definice osobního údaje 16
2.3 Limitující závislost na jazyce trénovacích datasetů 17
2.4 Nemodulární implementace existujících nástrojů 18
2.5 Shrnutí 19
3 Metodika práce 20
3.1 CRISP-DM 20
3.1.1 Fáze Business Understanding 21
3.1.2 Fáze Data Understanding 21
3.1.3 Fáze Data Preparation 22
3.1.4 Fáze Modeling 22
3.1.5 Fáze Evaluation 22
3.1.6 Fáze Deployment 23
4 Kategorizace osobních údajů 24
4.1 Osobní údaje v kontextu Veřejného registru smluv 24
4.1.1 Zákon o registru smluv 24
4.1.2 Registr smluv 24
4.1.3 Nástroj pro anonymizaci dokumentů 25
4.2 Typy vstupních dokumentů 26
4.3 Kategorizace osobních údajů 27
4.3.1 Kompletní seznam 29
4.4 Mapa kategorií 30
4.4.1 Jmenné identifikátory 32
4.4.2 Číselné identifikátory 32

4.4.3 Znakové identifikátory ... 32
4.4.4 Lokalizační identifikátory ... 32
4.4.5 Ostatní identifikátory ... 32
4.4.6 Redukovaný seznam identifikátorů ... 33
5 Použité techniky hlubokého učení ... 34
5.1 Hluboké učení ... 34
5.2 Natural Language Processing ... 35
5.3 Named Entity Recognition ... 36
5.4 NER v českém jazyce ... 39
5.4.1 Stematizace a lematizace ... 40
5.4.2 POS ... 40
5.5 Multilingvní modely ... 40
5.5.1 mBERT ... 41
5.5.2 XLM ... 42
5.5.3 XML-RoBERTa ... 43
6 Výběr technologií ... 44
6.1 NLP frameworky ... 44
6.1.1 Kritéria pro výběr NLP frameworku ... 45
6.1.2 Stanford CoreNLP (Stanford NER) ... 45
6.1.3 NLTK ... 45
6.1.4 Flair ... 46
6.1.5 SpaCy ... 46
6.1.6 Porovnání ... 47
6.2 Vývojové prostředí ... 47
7 Dataset ... 49
7.1 Anotace NER datasetů ... 49
7.1.1 Granularita NER datasetů ... 49
7.1.2 Způsoby anotace jmenných entit ... 50
7.2 Dostupné datasety ... 50
7.2.1 Czech Named Entity Corpus (CNEC) ... 50
7.2.2 BSNLP ... 50
7.2.3 SumeCzech-NER ... 51
7.2.4 Shrnutí ... 51
7.3 Zpracování datasetu ... 51
7.3.1 Načtení CNEC_extended ... 52

7.3.2 Načtení CNEC 2.0 52
7.4 Analýza dat 54
7.4.1 Kategorie klasifikovaných jmenných entit 54
7.4.2 Úprava datasetu 55
8 Vývoj prototypu 59
8.1 Trénování NER modelu 59
8.1.1 Konfigurace spaCy pipeline 60
8.1.2 Evaluace trénovaných modelů 64
8.2 Návrh a implementace anonymizačního nástroje 67
8.2.1 SDK Presidio 67
8.2.2 Vývoj a přiřazení recognizerů 68
8.2.3 Namapování kategorií osobních údajů na Presidio recognizery 69
8.3 Grafické rozhraní 71
9 Evaluace anonymizačního nástroje 74
9.1 Vytvoření doménově specifického evaluačního datasetu 74
9.2 Automatizovaná evaluace 75
9.3 Zhodnocení výsledků 77
Závěr 79
Použitá literatura 82
Přílohy I
Příloha A: Kompletní mapa kategorií osobních údajů I
Příloha B: Zdrojový kód Anonymizačního nástroje I
Příloha C: Evaluační dataset contract_eval I

Seznam obrázků

Obrázek 1: Fáze metodiky CRISP-DM 21
Obrázek 2: Nástroj pro anonymizaci dokumentů – intrografika 26
Obrázek 3: Rozdíl mezi distribuovaným a nedistribuovaným rozdělením vstupních dat (Ganesh, 2019) 38
Obrázek 4: Zpracování vstupu pomocí modelu mBERT (Devlin et al., 2019) 42
Obrázek 5: Zpracování vstupu modelem XML (Lample a Conneau, 2019) 43
Obrázek 6:Hierarchie jmenných entit CNEC 2.0 54
Obrázek 7:Zastoupení jmenných entit v datasetu CNEC 2.0 56
Obrázek 8: Graf rozložení výskytu jmenných entit v upraveném datasetu CNEC 2.0 58
Obrázek 9: CPU spaCy pipeline 60
Obrázek 10: GPU spaCy pipeline 60
Obrázek 11: Ukázka grafického rozhraní anonymizačního nástroje 73
Obrázek 12: Grafická ukázka nástroje Label Studio 75
Obrázek 13: Výsledná evaluace anonymizačního nástroje používajícího model CPU_fine_nomorph 77
Obrázek 14: Výsledná evaluace anonymizačního nástroje používajícího model GPU_bert_cased 77

Seznam tabulek

Tabulka 1: Mapa kategorií osobních údajů 31
Tabulka 2: Seznam kategorií jmenných entit upraveného datasetu CNEC 2.0 58
Tabulka 3: Evaluace NER modelů – porovnání 66
Tabulka 4: Mapa kategorií osobních údajů - výsledná kategorizace.....71

Úvod

S nástupem hromadného zpracování velkých dat a následným příchodem bezpečnostních opatření, reagujících na zvýšenou potřebu ochrany soukromí jednotlivců v online prostoru, vznikla celá nová řada činností, které musí nyní firmy vykonávat, aby se vyhnuly velkým pokutám.

Jednou z těchto činností je i detekce a klasifikace osobních údajů v různých textech. Tato práce je ve velké části případů vykonávána kvůli povinnosti anonymizace těchto údajů na základě různých zákonných opatření, týkajících se ochrany osobních údajů, v čele se známým evropským nařízením GDPR.

Problematika detekce a klasifikace identifikátorů v textu je součástí vědecké oblasti s názvem Named Entity Recognition (NER), která se soustavně vyvíjí již mnoho let. Samotný termín Named Entity byl definován R. Grishmanem již v roce 1996. Od té doby došlo k objevu řady významných metod klasifikace jmenných entit, jako jsou některé slovníkové metody, klasifikace pomocí lookup tabulky nebo pomocí regulárních výrazů (Nadeau, 2007). Poslední dobou se však jako nejvhodnější způsob pro klasifikaci jmenných entit v textu jeví využití hlubokého učení, zejména pak technik Natural Language Processing (NLP). To totiž umožňuje nastavit detekční pravidla, která se vyznačují vyšší abstrakcí a dokážou tak zachytit i případy, které se v textu vymykají běžným slovním vyjádřením. To v důsledku znamená, že techniky hlubokého učení mají v NER výsadní postavení a v současné době lze s jejich pomocí dosáhnout nejlepších výsledků (Xu et al., 2021).

Detekce a klasifikace osobních údajů je tedy typově specifickou aplikací NER a potažmo NLP na konkrétní sadu dokumentů, obsahujících osobní údaje. Problém v tomto případě vzniká nejasnou definicí toho, co to osobní údaj vlastně je, respektive tím, že se osobním údajem mohou stát i informace, které nejsou explicitně unikátní (Mihulková, 2018). V některých případech je tedy nutné stanovit mez, co vše se dá považovat za osobní údaj na základě doménově specifického charakteru analyzovaného textu. To existující zahraniční vědecké práce, které se detekcí osobních údajů již zabývají spíše ignorují a soustředí se více na kvalitativní zpřesnění detekce jmenných entit na základě pokročilejších statistických modelů (Silva et al., 2020).

Ač se řada současných prací problematikou doménově specifické NER zabývá, je tato funkčnost modelů hlubokého učení stále problematická a složitá (Lim et al., 2019). Každý nově vyvinutý model je totiž vytrénován jen pro klasifikaci sady několika doménově specifických jmenných entit, což však zabraňuje jeho nasazení na rozdílné sadě dokumentů, které jsou součástí jiné domény a potažmo obsahují i rozdílné druhy jmenných entit.

Současná řešení, která se klasifikací osobních údajů zabývají, se tak nedají považovat jako dostatečně modulární, aby se dala aplikovat v různých doménách na konkrétní specifické typy jmenných entit a rozdílné druhy vstupních dokumentů. Klíčovým úkolem této práce je tedy zkoumat možnosti, jak vytvořit nástroj, který se dá jednoduše upravit pro konkrétní využití na různé typy vstupních dokumentů patřících do rozdílných domén.

V této diplomové práci je navržen a implementován pomocí state-of-the-art metod hlubokého učení prototyp tohoto modulárního anonymizačního nástroje. Ten je následně pro jeho pohodlnou evaluaci upraven tak, aby dokázal klasifikovat osobní údaje v dokumentech běžně nahrávaných do Veřejného registru smluv. V této podobě je tento nástroj možné využít jako volné rozšíření existujícího nástroje „Nástroj pro anonymizaci dokumentů“, dostupného na Portálu veřejné správy[1].

1.1 Vymezení problému

Subjekty, které mají povinnost nahrávat smlouvy do Veřejného registru smluv, čelí při tomto procesu řadě těžkostem, které dokonce mohou vzhledem k šíři problematiky klasifikace různých druhů osobních údajů v některých případech zamezit vkládání smluv pracovníkům bez dostatečného právnického a technického vzdělání.

Jedním z těchto problémů je povinnost anonymizovat osobní údaje v nahraných dokumentech a přílohách. Tato povinnost je stanovena dle 160/1999 Sb. Zákona o svobodném přístupu k informacím a následně i 101/2000 Sb. Zákona o ochraně osobních údajů, který upravuje předpisy GDPR. Samotná činnost detekce osobních údajů pro jejich následnou anonymizaci je nejen jednotvárná a nezáživná, ale existuje zde velké riziko lidské chyby, ať už přehlédnutím důležitého údaje nebo i jeho špatnou klasifikací a interpretací.

Následující problémy jsou společné pro jakoukoliv detekci osobních údajů v nestrukturovaném textu. Tímto pojmem (nestrukturovaný text) se označuje jakýkoliv text, který nemá dopředu pevně stanovenou strukturu v podobě předem definovaných kapitol či povinných textových částí. Neobsahuje také žádná povinná textová pole, do kterých by byly zapisovány konkrétní informace (např. pole pro jméno a příjmení).

1.1.1 Absence jednoznačné definice osobního údaje

Dle Zákona o zpracování osobních údajů lze termín osobní údaj definovat jako jakoukoli informaci, která se týká identifikované nebo identifikovatelné žijící osoby (Mihulková, 2018). Tato definice je z hlediska implementace anonymizačního nástroje velmi široká. Neexistuje žádný souhrnný seznam, který by definoval, co všechno se dá považovat za osobní údaj. Je proto třeba definovat jednotlivé osobní, případně citlivé osobní údaje pro konkrétní typ dokumentů, které bude nástroj zpracovávat. Tento seznam osobních údajů je rozdílný pro různé typy dokumentů. Pro funkční implementaci anonymizačního nástroje je ale tento seznam klíčový.

1.1.2 Limitující závislost na jazyce trénovacích datasetů

Pro úspěšné využití jakékoliv techniky hlubokého učení je potřeba mít k dispozici kvalitní dataset, na jehož datech může být vznikající model natrénován. V oblasti NLP (Natural

[1] https://anonymizace.gov.cz/crossroad

Language Processing) je tak klíčové mít buď k dispozici dataset v požadovaném jazyce nebo předtrénovaný multilingvní model, který by se dal následně metodou transfer learning přetrénovat pro požadované využití (Moberg, 2020). V případě detekce osobních údajů se navíc v existujících datových sadách obsahujících ukázky různých smluv logicky žádné osobní údaje nevyskytují, protože tyto smlouvy již anonymizací musely projít.

1.1.3 Nemodulární implementace existujících nástrojů

Jednotlivé již existující nástroje, určené pro detekci identifikátorů v nestrukturovaném textu nejsou dostatečně modulární, aby se daly upravit pro konkrétní doménově specifickou aplikaci. Zároveň se většina existujících akademických řešení soustředí na detekci pouze několika druhů jmenných entit, což vzhledem k větší šíři detekce konkrétních údajů v doménově specifické aplikaci nestačí. Pro větší flexibilitu a modularitu vyvíjeného konceptu je tedy třeba rozšířit tradiční seznam jmenných entit tak, aby jejich šíře lépe odpovídala možnému rozšíření nástroje na konkrétní sadu dokumentů a identifikátorů.

Zároveň jsou vyvíjené NER modely často vytvářeny pouze jako akademické koncepty a necílí na žádnou konkrétnější doménu do které by měly spadat zpracovávané dokumenty. Také vzhledem k jejich abstraktnímu využití nejsou některé tyto modely veřejně dostupné.

1.2 Účel a cíle práce

Hlavním cílem této práce je navrhnout a implementovat prototyp nástroje, který pomocí algoritmů hlubokého učení a případně i dalších relevantních metod dokáže v nestrukturovaném textu klasifikovat osobní údaje a umožní tak rychlejší a přesnější zpracování dokumentů v případě potřeby anonymizace těchto údajů.

Klíčovým požadavkem pro tento prototyp je jeho modularita. To znamená, že vyvinutý nástroj musí být dostatečně modulární do té míry, aby se dal jednoduše programaticky přizpůsobit klasifikaci osobních údajů v různých doménách. Různými doménami se pak v tomto případě myslí konkrétní aplikace klasifikačního nástroje na konkrétních typech textových dokumentů s doménově specifickými typy osobních údajů – jmenných entit.

V tomto případě je konkrétní implementace nástroje přizpůsobena úloze povinné anonymizace osobních údajů při nahrávání dokumentů do veřejného registru smluv. Může tedy sloužit jako rozšíření existujícího nástroje “Nástroj pro anonymizaci dokumentů”, dostupném na Portálu veřejné správy[2] a pomáhat může všem subjektům povinným v tomto registru smlouvy zveřejňovat. Nástroj je následně evaluován na vhodně vybraném evaluačním datasetu, obsahujícím vhodný typ doménově specifického dokumentu.

Dílčím cílem práce je analyzovat zákony a nařízení, které je potřeba respektovat při anonymizaci osobních údajů při nahrávání dokumentů do Veřejného registru smluv. Na

[2] https://anonymizace.gov.cz/crossroad

základě této analýzy je vypracován model, který vyjadřuje míru potřeby anonymizace konkrétního identifikátoru v nestrukturovaném textu. Tento model slouží jako podklad pro vývoj konkrétních klasifikátorů v modulární struktuře vyvíjeného prototypu.

Dalším dílčím cílem je provést rešerši existujících metod hlubokého učení, využitelných pro klasifikaci osobních údajů v nestrukturovaném textu, zejména pak metod, které umožňují tuto klasifikaci v textech v českém jazyce. Dochází také k analýze možností kombinace těchto metod za účelem dosažení co nejlepšího výsledku detekce a klasifikace údajů.

1.3 Omezení projektu

Vzhledem ke komplexnosti celé problematiky ochrany osobních údajů a kvůli specifickému charakteru osobních údajů v různých druzích textů je vyvíjený koncept navržen modulárně tak, aby se dal přizpůsobit konkrétní aplikaci klasifikace osobních údajů. V další fázi je však omezen jen na jednu konkrétní sadu dokumentů, a to na sadu textových dokumentů, které jsou běžně nahrávány do Veřejného registru smluv.

Dále je práce omezena limitujícím počtem českých datasetů vhodných pro trénování vyvíjeného modelu. Vytváření nutně komplexnějších datových sad není součástí této práce. To může znamenat menší přesnost detekce než u modelů vyvíjených zahraničními výzkumníky a trénovaných na rozsáhlejších a lépe připravených textových datech v jiném jazyce.

Práce se také nesnaží zkoumat nové algoritmy hlubokého učení. Místo toho v ní jde o výběr state-of-the-art metod hlubokého učení a jejich aplikaci na danou problematiku. Znamená to, že kvalita klasifikace jmenných entit vyvíjeného nástroje bude s největší pravděpodobností maximálně stejně velká, jako přesnost klasifikace u existujících metod hlubokého učení, z kterých bude vycházet.

1.4 Význam a přínos práce

Přínosem této práce je samotný prototyp modulárního anonymizačního nástroje pro detekci a klasifikace osobních údajů v nestrukturovaných textech. Modularita nástroje by měla umožňovat ostatním výzkumníkům a uživatelům jednoduše převzít existující řešení a upravit nástroj tak, aby dokázal detekovat a klasifikovat jiný, doménově specifický seznam osobních údajů, a to v jiných, doménově specifických typech dokumentů. Nástroj pak bude vhodný k detekci a klasifikaci osobních údajů i v jiném případě užití, jako je například anonymizace osobních údajů při digitalizaci veřejně přístupného archivu nebo při pseudonymizaci API requestů, obsahujících některé citlivější informace.

Dalším přínosem této práce je upravení modulárního anonymizačního nástroje pro konkrétní aplikaci, kterou je klasifikace osobních údajů v nestrukturovaných dokumentech běžně nahrávaných do Veřejného registru smluv. Tato část slouží jako konkrétní proof of concept využitelnosti modulárního přístupu při anonymizaci osobních údajů v doménově specifické oblasti.

Vyvinutý nástroj pomůže všem subjektům, povinným uveřejňovat smlouvy ve Veřejném registru smluv, v rychlejší a přesnější detekci a klasifikaci osobních údajů v nahrávaných dokumentech za účelem jejich následné anonymizace. V ideálním případě bude moci být tento nástroj použit jako rozšíření existujícího nástroje "Nástroj pro anonymizaci dokumentů", dostupném na Portálu veřejné správy[3].

V neposlední řadě se dá za přínos považovat i fakt, že vytrénovaný model pro klasifikaci jmenných entit je určen pro klasifikaci osobních údajů v českém jazyce, což vzhledem ke složitosti českého jazyka a stavu vědeckého poznání popisovaného běžně na anglických příkladech není triviální záležitost. Zároveň zde dochází k využití moderního state-of-the-art NLP frameworku spaCy[4] a k upravení open source SDK Presidio[5]. Práce s oběma těmito technologiemi je prováděna s ohledem na jejich efektivní využití pro klasifikaci českých textů.

[3] https://anonymizace.gov.cz/crossroad
[4] https://github.com/explosion/spaCy
[5] https://github.com/microsoft/presidio

2 Rešerše

2.1 Rešeršní strategie

Systematická část rešerše vhodné literatury proběhla s využitím databáze ACM, Google Scholar a vyhledávačem Univerzity Karlovy UKAŽ. Na ACM a Google Scholar byly vyhledávány práce technického zaměření, týkající se nejnovějších poznatků NLP ve spojení s klasifikací a detekcí jmenných entit v textu nebo využití multilingvních modelů v praktických aplikacích.

V databázi UKAŽ pak byly vyhledávány práce zabývající se ochranou osobních údajů a jejich logickou identifikací, kterých nebylo na předchozích databázích nalezeno dostatečné množství. Nebyl vybrán žádný časový filtr, ale bylo nastaveno řazení článků od nejnovějších ke starším a následně i řazení dle relevance. Při každém hledání bylo analyzováno prvních 20 článků, přičemž největší důraz byl při tom kladen na praktické využití technik NLP při detekci textových identifikátorů co nejvíce podobných osobním údajům.

Při vyhledávání docházelo ke spojením několika klíčových slov logickými operátory.

Příklady klíčových slov, použitých ve vyhledávacích řetězcích ACM a Google Scholar

NLP, Natural Language Processing, NER, Named Entity Recognition, multilingual, PII, personally identifiable information, Cognitive Data Capture, privacy, data privacy, private, personal, anonymization, pseudonymization, data security

Příklady vyhledávacích řetězců pro ACM a Google Scholar

- (NLP OR Natural Language Processing) AND multilingual
- (NER OR Named Entity Recognition) AND (PII OR personally identifiable information)
- (NER OR Named Entity Recognition) AND (PII OR personally identifiable information) AND (NLP OR Natural Language Processing)
- (NLP OR Natural Language Processing) AND (Cognitive Data Capture)
- (NLP OR Natural Language Processing) AND privacy
- (NLP OR Natural Language Processing) AND data privacy
- (NLP OR Natural Language Processing) AND private
- (NLP OR Natural Language Processing) AND personal
- (NLP OR Natural Language Processing) anonymization

Příklady klíčových slov, použitých ve vyhledávacích řetězcích UKAŽ
Ochrana osobních údajů, automatizovaný, GDPR

Příklady vyhledávacích řetězců pro UKAŽ:

- jakékoliv pole obsahuje Ochrana osobních údajů
- jakékoliv pole obsahuje Ochrana osobních údajů AND jakékoliv pole obsahuje automatizovan
- jakékoliv pole obsahuje GDPR
- jakékoliv pole obsahuje GDPR AND jakékoliv pole obsahuje automatizovan

V nesystematické části rešerše došlo k prohledávání internetu se zaměřením na firmy, zabývající se digitalizací dokumentů a na konkrétní informace týkající se ochrany osobních údajů (například definice samotného osobního údaje). Pozorně prostudována byla například webová stránka Evropské komise[6], která srozumitelnou formou čtenáře seznamuje se základními pojmy v oblasti ochrany údajů v Evropské unii. Došlo také na osobní konzultace s odborníkem, ve kterých byla doporučena odborná literatura týkající se ochrany osobních údajů.

V následujících podkapitolách je provedena rešerše, vztahující se k uvedeným problémům popsaných v části *1.1 Vymezení problému*. Ke každému z těchto problémů je přiřazena jedna z následujících podkapitol a v závěrečné podkapitole *2.5 Shrnutí* je nastíněn postup dalšího řešení těchto problémů v dalších částech této práce.

2.2 Absence jednoznačné definice osobního údaje

Termín osobní údaj, definovaný již v kapitole *1.1 Vymezení problému*, může mít ze své podstaty mnoho významů. Informace je totiž za osobní údaj považována až ve chvíli, kdy vede k přímé a jednoznačné identifikaci jednotlivce. Často se tedy stává, že osobním údajem se údaj stává až ve chvíli, když se vyskytuje v textu společně s dalšími údaji, které dohromady tvoří ucelenou informací o konkrétní osobě (Mihulková, 2018).

Do tohoto vymezení se mísí i koncepce práva na soukromí, která je ukotvena v evropských, kontinentálních právních kulturách. V této koncepci se jako osobní údaj uvažuje i jakákoliv informace, kterou jednotlivec sám o sobě nechce sdílet ve veřejném prostoru (Kubica, 2019).

Jako jednotlivá podkategorie osobních údajů se rozlišují tzv. zvláštní osobní údaje. Tyto údaje mohou být buď citlivějšího charakteru a vypovídat o rasovém původu, politických názorech nebo sexuálním vyznání a celkově vzato jsou klasifikovány tak, že mohou subjekt poškodit ve společnosti. Mezi tyto údaje potom patří i biometrické údaje, jakožto jednoznačné fyzické identifikátory subjektu (MVČR).

Problém vzniká ve státní správě, kde může kolidovat 106/1999 Sb. Zákon o svobodném přístupu k informacím a 101/200 Sb. Zákon o ochraně osobních údajů. Občan má totiž právo

[6] https://ec.europa.eu/info/law/law-topic/data-protection_en

přístupu k informacím, které nejsou omezené nařízeními o ochraně osobních údajů. Tyto informace potom může dostat v anonymizované formě. Ve chvíli, kdy ale údaj, jako je například křestní jméno není považován za osobní údaj, nespadá pod zákon o ochraně osobních údajů a měl by se dostat k žadateli v plné, neanonymizované podobě (Gealfow et al., 2019).

Na základě těchto tezí můžeme tušit, že pojem osobní údaj se liší podle konkrétní aplikace využití informací, a i podle sektoru, ve kterém ke zpracování dat dochází. Ostatně i například na stránce Evropské komise[7] a v článku Jitky Mihulkové (2018) se uvedené příklady osobních údajů liší.

2.3 Limitující závislost na jazyce trénovacích datasetů

V oblasti strojového učení, a zejména pak v podoblasti NLP je často adresován problém nelokalizovaných vstupních dat. Vzhledem k dnešnímu stavu vědeckého světa se totiž výzkumné práce píšou ve většině případech v angličtině a v důsledku toho existuje i největší množství předtrénovaných modelů, které jsou založeny právě na anglických korpusech (Pod pojmem korpus se rozumí množina slov, která může být využita pro trénink modelu. Většinou jsou slova v korpusu spojená vazbami k doméně, ke které je korpus plánovitě vytvořen). Angličtina je také výhodná z pohledu Named Entity Recognition (NER), protože například oproti češtině se v ní nevyskytují ve větším množství pády, rody či větší množství nepravidelných tvarů množných čísel podstatných jmen.

Možným řešením problému s nedostatkem kvalitních korpusů různých jazyků mohou být moderní modely, jako je například model ELECTRA, který je vhodný pro trénování neuronové sítě architektury transformer i menším výpočetním výkonem a korpusem menší velikosti (Clark et al., 2020). Populárním řešením je také využití revolučního modelu BERT, publikovaného týmem výzkumníků z Google v roce 2018 (Devlin et al., 2019), zejména pak jeho multilingvní varianty M-BERT, která v současné době podporuje 104 jazyků, včetně češtiny[8]. Problém v tomto případě může být imbalance korpusu, na kterém byl model trénován (celosvětové záznamy Wikipedie). Kvůli rozdílnému poměru celkového množství textu každého jazyka v trénovacím datasetu se tak u modelu M-BERT dají u méně zastoupených jazyků čekat horší výsledky (Chau et al., 2020).

Jako další state-of-the-art multilingvní modely se dají označit modely XLM a jeho upravená varianta XLM-R, které mají ve srovnání provedeném Johnem Mobergem na 15 jazycích o několik procent lepší úspěšnost (Moberg, 2020).

[7] https://ec.europa.eu/info/law/law-topic/data-protection_en
[8] https://github.com/google-research/bert

2.4 Nemodulární implementace existujících nástrojů

Jak již bylo dříve zmíněno, pro anonymizaci osobních údaje je vhodné zvolit metod NLP, které jsou součástí technik hlubokého učení (Qu el al., 2021; Silva et al., 2020; Ellman, 2018). Jednotlivé metody lze pro dosažení optimálních výsledků potom různě kombinovat. Většina existujících prací se však zabývá pouze zkoumáním a implementací těchto metod zvlášť.

Silva ve své práci využívá NER a porovnává v ní vhodné NLP toolkity, které se hodí pro klasifikaci osobních identifikátorů v smlouvách. S největším F1 score mu v jeho případě funguje toolkit Stanford CoreNLP (Silva et al., 2020). Jeho výsledek pak rozporuje např. Mendels, kterému v jeho případě nejlepší F1 score vychází u toolkitu Flair (Mendels, 2020).

Mendels sám uvádí další vhodné metody vhodné pro detekci identifikátorů, jako je využití regulárních výrazů uvnitř klasifikačních vrstev nebo vytvoření blacklistů s textovými řetězci s větší pravděpodobností výskytu hledaného výrazu.

Zajímavá je práce Mathiase Ellmanna, který se pomocí NLP snaží detekovat duplikáty „issue trackerů". Narozdíl od předchozích autorů jde více do hloubky (vzhledem k tomu, že pracuje pouze s binárním klasifikátorem si to může dovolit) a tím dokáže lépe zpracovat samotnou extrakci příznaků z textu na základě sémantické analýzy textu originálního „issue". Tím potom dochází k rozšířenému chápání širšího kontextu a menšímu zaměření na samotné identifikátory (Ellman, 2018).

Širším kontextem se zabývá také Jake Williams, který ve své práci vytváří „interní" a „externí" model. Interní model je určen pro lokální klasifikaci identifikátorů jako je název města nebo jméno. Externí model potom pracuje s již předzpracovaným textem připraveným interním modelem. Tento text obsahuje klasifikované identifikátory nahrazené klíčovými slovy, která by měly být z hlediska širšího kontextu relevantními pro lepší pochopení textu (Williams, 2017).

Chen Qu se zabývá konceptem typických architektur anonymizačních modelů a pokládá otázku, jak s anonymizovanými daty dokážou pracovat další vrstvy hlubokých neuronových sítí, které byly předučeny na neanonymizovaných datech (Qu el al., 2021).

Ve všech výše zmíněných pracích se vyskytuje jeden problém. Ač všechny trénované klasifikátory dosahují kvalitních výsledků, chybí zde kvalitnější uzpůsobení modelu konkrétní doméně. Silva například trénuje model, který dokáže klasifikovat osobní údaje ve smlouvách, jenže šíře kategorií osobních údajů je moc malá na to, aby došlo k detekci všech různých druhů osobních údajů (Silva et al., 2020).

Texty, na kterých byl model trénován totiž obsahují anotace pouze pro základní druhy jmenných entit, jako jsou jména, adresy, lokace a podobně. To pramení především z nejběžněji dostupných a v akademické sféře používaných veřejných datasetů, které jsou často anotovány dle standardu, který definuje pravidelná konference CONLL (Yadav et al., 2019).

Zároveň ve vědeckých pracích zcela chybí důraz na modularitu vyvíjených řešení. To je většinově dáno tím, že práce se zabývají implementací konkrétního modelu, spíše než implementací konkrétního řešení.

2.5 Shrnutí

Z provedené rešerše vychází, že zde existuje prostor pro vytvoření modulárního nástroje pro klasifikaci a anonymizaci osobních údajů, který bude možné upravovat pro konkrétní aplikační využití.

Přitom je nejprve je na základě provedené analýzy potřeba definovat konkrétní typy osobních údajů pro konkrétní aplikaci. Tyto typy osobních údajů se přitom mohou pro různé typy doménově specifických aplikací lišit.

Z rešerše také vychází, na co je třeba dávat si pozor při výběru modelu a při vytváření architektury nástroje a kombinaci různých NLP metod. Je také potřeba vybrat vhodný dataset, který musí reflektovat nejen jazyk analyzovaného textu při detekci osobních údajů, ale i vybraný model hluboké neuronové sítě, který tomu musí být přizpůsoben.

3 Metodika práce

Rešeršní strategie, která byla pro tuto práci zvolena, je popsána v předchozí kapitole, kde je popsán současný stav poznání pro zkoumanou oblast. Stejná strategie bylo zvolena i pro další části práce, které se zabývají konkrétnějšími způsoby řešení uvedených problémů. Následující kapitoly potom kombinují rešerši existujícího stavu poznání s praktickou aplikací zjištěných poznatků v kontextu obsahu práce.

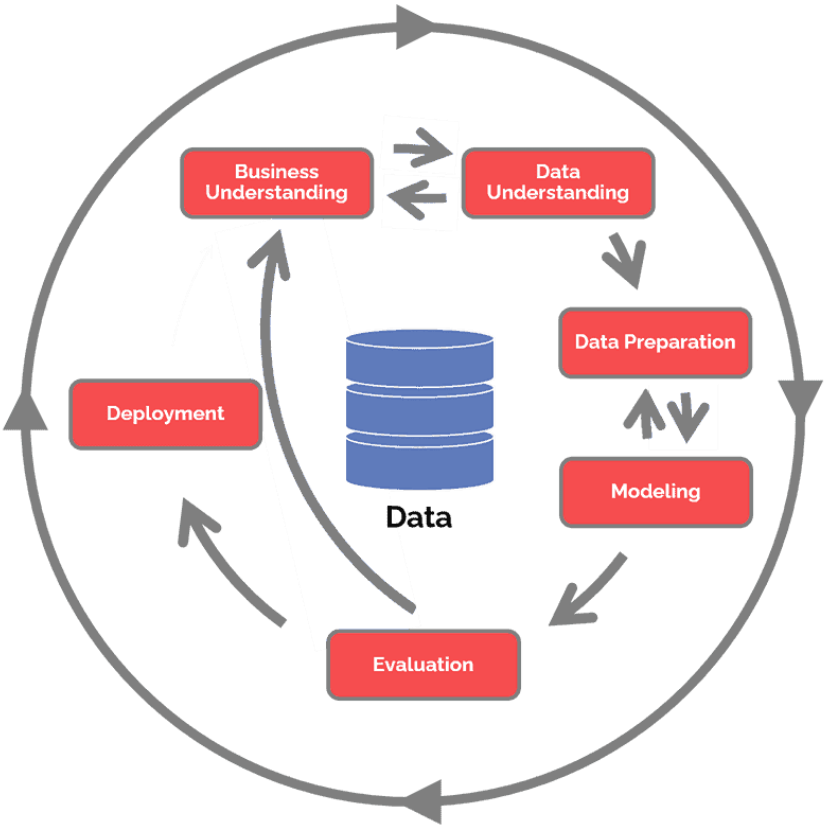
Výběr vhodných technologií byl proveden definováním seznamu důležitých kritérií a následným porovnáním zkoumaných frameworků a nástrojů vůči tomuto seznamu. Vzhledem k tomu, že dále použitá metodika CRISP-DM nedefinuje výběr programového vybavení a nástrojů, je zde tato část uvedená samostatně. Výběr vhodných technologií obsahuje kapitola 6.

3.1 CRISP-DM

Pro vývoj prototypu a jeho evaluaci bylo využito upravené metodiky CRISP-DM (Cross-industry standard process for data mining) (Chapman et al., 2000). Ta je vzhledem k charakteru vývoje nástrojů založených na strojovém učení často využívána a je vhodnější alternativou než některé v jiných oblastech hojněji využívané agilní metodiky (Pinhasi, 2021).

Tato metodika je také dostatečně flexibilní, aby se dala jednoduše přenést na řešený problém. Zároveň v kontextu této práce došlo k upravení metodiky tak, aby odpovídala vývoji prototypu nástroje, a ne jeho komplexní komerční verze. Metodika není závislá na konkrétním odvětví, ani na konkrétním využitém programovém vybavení (Chapman et al., 2000).

Procesní model metodiky CRIPS-DM se dělí na 6 fází: Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation a Deployment. Pořadí těchto fází při vývoji produktu je vyobrazeno na obrázku 1. Je zde také vidět, že celá metodika je postavená kolem práce se zdrojovými daty, v případě této práce kolem zdrojového datasetu, který musí být správně vybrán a pochopen. Šipka, která na obrázku uzavírá všechny fáze a tvoří tak kompletní kruh naznačuje, že má celý proces cyklický charakter.



Obrázek 1: Fáze metodiky CRISP-DM

3.1.1 Fáze Business Understanding

Tato fáze se zaměřuje na pochopení business cílů a požadavků na vyvíjený konečný produkt. Tvoří tak základ pro pochopení celého produktu a specifikuje doménovou oblast, do které bude konečný produkt zařazen. Dochází zde také k určení konečné podoby výstupu projektu a k stanovení metrik pro vyhodnocení projektu jako úspěšného (Chapman et al., 2000).

V kontextu této práce je tato fáze metodiky splněna úvodní kapitolou, ve které jsou definovány cíle, omezení a přínos práce. Také zde dochází k zařazení práce do doménové oblasti ochrany osobních údajů a je zde popsáno, jak tato práce přispívá k řešení popsaných problémů.

Dále lze do této fáze zařadit i části kapitol 4 a 5, které se zabývají doménovou problematikou osobních údajů a jejich kategorizací a analýzou vhodných technik hlubokého učení.

3.1.2 Fáze Data Understanding

V této fázi jde nejprve o sběr vstupních dat a jejich popis. Následuje zkoumání vstupních dat a ověření kvality a vztahů mezi jednotlivými prvky. Tato fáze slouží ke zodpovězení otázky, zda se dá vstupní dataset považovat jako kompletní a zda je reprezentace dat dostatečně kvalitní k dalšímu zpracování (Chapman et al., 2000).

V této práci je tato fáze metodiky splněna v kapitole *7 Dataset*. V této kapitole je z několika variant vybrán a popsán vhodný dataset. Jsou zde diskutována kritéria pro nejvhodnější způsob reprezentace dat a dochází zde k vytvoření statistiky o zastoupení různých kategorií jmenných entit v datasetu.

3.1.3 Fáze Data Preparation

Po pochopení, popsání a prozkoumání vstupních dat následuje v metodice fáze Data Preparation, ve které se jedná o další přípravě těchto dat pro další zpracování. Dochází zde k úpravě dat takovým způsobem, aby došlo k jejich očištění a zpřesnění vzhledem k předchozím stanoveným cílům. Data jsou také transformována do vhodného formátu pro další zpracování (Chapman et al., 2000).

Stejně jako předchozí je tato fáze adresována v kapitole *7 Dataset*. Po předchozím zkoumání dat zde dochází k jejich očištění a k jejich transformaci do vhodného formátu.

3.1.4 Fáze Modeling

Tato fáze zachycuje vytváření a testování různých statistických modelů včetně výběru konkrétních vhodných algoritmů. Výsledek testování jednotlivých vytvořených modelů může vést zpět k fázi Data Preparation a k doupravení vstupních dat za účelem optimalizace dosaženého výsledku.

Vytváření a testování modelů založených na technikách hlubokého učení je popsáno v kapitole *8 Vývoj prototypu*. V této kapitole je popsán proces trénování modelů a výběru jednotlivých dílčích kandidátů, stejně tak jako jejich průběžné testování a závěrečné zhodnocení jejich kvality. Následně zde pak dochází k integraci nejlépe ohodnoceného modelu do dále vyvíjeného modulárního anonymizačního nástroje.

3.1.5 Fáze Evaluation

Oproti předchozí fázi, která v sobě také obsahuje dílčí testování modelů (a odpovídá tak spíše termínu verifikace) se tato fáze zaměřuje více na uživatelské testování konečného výsledku. Dochází zde k zodpovězení otázky, zda projekt dosáhl předem stanovených cílů nebo je třeba zahájit další iteraci vývoje.

V kapitole *9 Evaluace anonymizačního nástroje* je provedena celková evaluace finálního vytvořeného produktu, a to pomocí automatizovaného testování na nově vytvořeném validačním datasetu, představujícím produkční data (smlouva). Dále je zde provedeno závěrečné zhodnocení kvality výsledného anonymizačního nástroje a následně jsou diskutovány další způsoby zlepšení. To vše je provedeno v návaznosti na určené cíle projektu, popsané v kapitole 1.

3.1.6 Fáze Deployment

Tato fáze popisuje uvedení výsledného produktu do produkčního prostředí a následné monitorování provozu a údržbu. Zároveň je v této fázi zařazena retrospektiva, která má pomáhat v efektivnější přípravě dalších projektů.

Vzhledem k tomu, že tato práce popisuje pouze vývoj prototypu, není tato fáze metodiky v práci využita. Dá se nicméně říci, že část popisující retrospektivu je splněna v kapitole 9 a v závěru práce.

4 Kategorizace osobních údajů

Jak již bylo zmiňováno výše, druh informace, který se dá považovat za osobní údaj se liší v závislosti na kontextu zpracovávaného textu. Dle domény, v které má anonymizační nástroj fungovat, je tedy potřeba identifikovat konkrétní druhy osobních údajů pro danou sadu zpracovávaných dokumentů.

Ač je cílem této práce vyvinout prototyp nástroje, který bude umožňovat jeho úpravu pro různé doménově specifické aplikace, je v následující kapitole provedena analýza a kategorizace osobních údajů vyskytujících se v dokumentech běžně nahrávaných do Veřejného registru smluv. Lze zde tak vidět kategorizaci typu osobních údajů pro konkrétní doménu, která je v případě této práce vybrána tak, aby se na ní dalo aplikací vyvíjeného anonymizačního nástroje dokázat jeho funkčnost a otestovat jeho spolehlivost.

4.1 Osobní údaje v kontextu Veřejného registru smluv

4.1.1 Zákon o registru smluv

340/2015 Sb. Zákon o registru smluv, který nabyl platnosti 1. 7. 2016 a který byl 2x novelizován, nejprve v roce 2017 (zákonem 249/2017 Sb.), dále v roce 2019 (zákonem 177/2019 Sb.), hovoří o zvláštních podmínkách účinnosti některých smluv a o uveřejňování těchto smluv ve Veřejném registru smluv (Bouda, 2019).

Tento zákon stanovuje povinnost k uveřejňování soukromoprávních smluv a smluv o poskytnutí dotace nebo návratné finanční výpomoci, jejichž alespoň jednou stranou je některý z okruhu vymezených povinných subjektů. Tímto subjektem je běžně míněn některý veřejnoprávní subjekt, který je definován v uzavřeném výčtu dle §2 tohoto zákona. Jsou zde také definované podmínky pro povinnost uveřejnění smlouvy, jako je například minimální výše hodnoty plnění smlouvy 50 000 Kč nebo plnění smlouvy převážně na území České republiky. Subjekt musí dle tohoto zákona také splňovat podmínky platnosti uveřejnění, jako je například nutnost uveřejnění 30 dnů od jejího uzavření (Kraus, 2022).

4.1.2 Veřejný registr smluv

Veřejný registr smluv je informačním systém veřejné správy, který slouží k uveřejňování smluv dle platného zákona. Správcem tohoto IS je Ministerstvo vnitra, které ale žádným způsobem neodpovídá za správnost nahraných smluv. Samotná kontrola správnosti smlouvy, a především kontrola znečitelnění obsahu (neboli anonymizace či úplné odstranění některých údajů) spadá na subjekt, který smlouvu uveřejňuje (Kraus, 2022).

Smlouvy je možné do registru zasílat přes grafické rozhraní skrz elektronický formulář, dostupný na Portálu veřejné správy nebo přes systém datových schránek.

Prostřednictvím registru smluv není možné zveřejňovat informace, které nelze poskytnout dle 106/1999 Sb. Zákona o svobodném přístupu k informacím. Těmi jsou v kontextu této práce a možností algoritmického zpracování dat pomocí hlubokého učení osobní údaje a obchodní tajemství, jehož detekce však není cílem této práce.

Práce s osobními údaji je dále omezená 110/209 Sb. Zákonem o zpracování osobních údajů a potažmo tedy i nařízením Evropského parlamentu a rady EU GDPR. Vzhledem k tomu, že při uveřejnění smlouvy není k dispozici oprávněný zájem ani jiný právní titul, který by tuto operaci ospravedlňoval (ÚOOÚ, 2016) a získání souhlasu by znamenalo další interakci se subjektem smlouvy, je nutné všechny osobní údaje dle jejich platné definice v uveřejňovaných smlouvách řádně anonymizovat. V případě neexistující či nekompletní anonymizace může Úřad pro ochranu osobních údajů (ÚOOÚ) udělit odpovědnému subjektu pokutu nebo se případně poškozená fyzická osoba může domáhat náhrady způsobené újmy (Kraus, 2022).

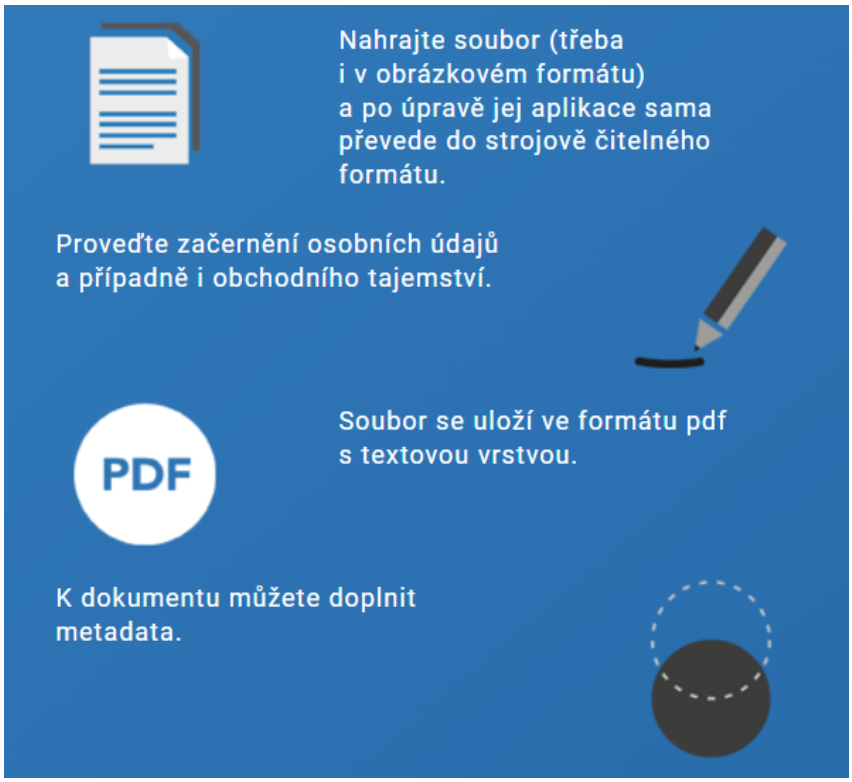
4.1.3 Nástroj pro anonymizaci dokumentů

Ministerstvo vnitra poskytuje přes Portál veřejné správy pro účely anonymizace textů uveřejňovaných smluv Nástroj pro anonymizaci dokumentů[9], který však mohou využívat pouze orgány veřejné moci.

Tento nástroj umožňuje řádným způsobem (znečitelněním všech vrstev textu) anonymizovat nahraný dokument. Dle textového popisu pro nepřihlášené uživatele však nenabízí automatickou detekci klíčových identifikátorů, a to ani v podobě jejich vyhledávání přes předdefinované šablony bez znalosti širšího kontextu textu. Nástroj vyvíjený v další části této práce by se tedy mohl stát dalším rozšířením tohoto nástroje a pomoci tak pracovníkům orgánů veřejné moci rychleji a správněji anonymizovat nahrávané dokumenty.

Na obrázku 2 lze vidět část úvodní intrografiky, dostupné na webu Portálu veřejné správy. Lze z něho vyčíst, že strojové zpracování textu, neboli Optical Character Recognition (OCR) při nahrání textu v obrazovém formátu nástroj již zvládá. Začernění osobních údajů a obchodního tajemství se ale stále musí provádět ručně.

[9] https://anonymizace.gov.cz/crossroad/



Obrázek 2: Nástroj pro anonymizaci dokumentů – intrografika

4.2 Typy vstupních dokumentů

Smlouvy smí do registru smluv být nahrávané pouze v elektronické podobě v několika povolených formátech. Elektronický obraz textového obsahu smlouvy se do registru zveřejňuje v otevřeném a strojově čitelném formátu, případně v otevřeném formátu umožňujícím úplně strojové zpracování textového obsahu (Kraus, 2022).

Otevřený a strojově čitelný formát definuje zákon 106/1999 Sb. o svobodném přístupu k informacím.

Otevřený formát nesmí být závislý na konkrétním technickém a programovém vybavení. To obvykle splňují formáty, které jsou udržovány neproprietárními standardizačními organizacemi a které nemají právní omezení pro jejich používání (např. PDF, ODT).

Strojově čitelný formát obsahuje takovou vnitřní strukturu, která umožňuje při dalším elektronickém zpracování extrahovat ze souboru čitelný a nijak nepozměněný text od originálu.

Do 31. prosince 2023 je možné do registru smluv nahrát i dokumenty, které sice umožňují úplné strojové zpracování textového obsahu, ale které v sobě neobsahují další informace o vnitřní struktuře dat, jako jsou informace o nadpisech, podnadpisech nebo poznámkách pod čarou. Vždy je ale nutné, aby dokument obsahoval čitelnou textovou vrstvu.

Tato informace je důležitá pro implementaci modelu hlubokého učení, který bude text zpracovávat. Znamená totiž, že metadata u jednotlivých textů nemusí být vždy dostupná a nelze se na ně tedy spolehnout.

Do registru smluv se nahrává smlouva ve stejném formátu, v jakém byla v originále podepsána, a to včetně všech jejích dodatků, které by měly být součástí jednoho dokumentu. Subjekt, který do registru smlouvu nahrává musí vyplnit i metadata. Ta by měla obsahovat základní informace o smlouvě, jako je identifikace smluvních stran, vymezení předmětu smlouvy, datum jejího uzavření a případně, pokud jí lze určit, i cenu předmětu vyjednávání. Informace uvedené v metadatech však musí splňovat stejná nařízení o nezveřejnitelných informacích jako samotný text smlouvy.

Součástí nahrání smlouvy mohou být i přílohy. Obrazové přílohy obvykle povinnost pro nahrávání nenaplňují. Textové přílohy musí splňovat stejné formální požadavky pro správný formát jako samotná smlouva.

4.3 Kategorizace osobních údajů

Na základě předpokládaného výskytu osobních údajů, běžně se vyskytujících ve smlouvách nahrávaných do registru smluv podle Metodického návodu k aplikaci Zákona o registru smluv (Kraus, 2022) a na základě úvahy autora byl vytvořen seznam specifických osobních údajů, které se pravděpodobně budou vyskytovat ve zpracovávaných dokumentech.

Údaje jsou roztříděné do několika základních kategorií, primárně dle formátu, ve kterém se údaj v textu vyskytuje (sekvence čísel, znaků, apod.).

Jmenné identifikátory

- Jméno
- Příjmení
- Příjmení za svobodna
- E-mail osobní
- E-mail pracovní
- Přihlašovací údaje (login)

Číselné identifikátory

- Rodné číslo/číslo pojištěnce
- Číslo občanského průkazu (OP)
- Číslo pasu
- Číslo řidičského průkazu
- Číslo platební karty
- Číslo jiné osobní karty (např. členské karty)
- Daňové identifikační číslo (DIČ)
- Číslo bankovního účtu
- Sdružené inkaso plateb obyvatelstva (SIPO)
- IBAN

- Telefonní číslo osobní
- Telefonní číslo pracovní
- Datum narození
- Věk

Znakové identifikátory

- Státní poznávací značka (SPZ)
- MAC adresa
- IP adresa
- Internetová doména
- Podpis
- Elektronický podpis
- Elektronický klíč
- Biometrická data
- Identifikační číslo vozidla (VIN)
- Parcelní číslo
- Sériové číslo
- GPS souřadnice

Nezařazené

- Adresa
- Pohlaví
- Rasa
- Náboženské vyznání
- Sexuální vyznání
- Titul
- Pracovní zařazení
- Politická příslušnost
- Státní příslušnost
- Obchodní tajemství
- Cena zakázky

Identifikátory nepodléhající anonymizaci

- Název firmy
- Identifikační číslo osoby (IČO)
- Id datové schránky
- Právnická osoba (PO)/Fyzická osoba podnikající (FOP)

Tento základní seznam ale není k dalšímu zpracování dostatečný, protože nereflektuje zvláštní situace, kdy některý údaj povinné anonymizaci může a nemusí podléhat. Zároveň nevystihuje jednotlivé vazby údajů mezi sebou a také v něm chybí rozpad na konkrétnější identifikátory nižší úrovně.

4.3.1 Kompletní seznam

V další fázi tedy byla po hlubší analýze vytvořena tabulka, která obsahuje podrobnější seznam jednotlivých identifikátorů.

Každému identifikátoru bylo přiděleno unikátní číslo ID, název, příznak, zda je identifikátor primární nebo sekundární, seznam závislých identifikátorů a příznak, zda je identifikátor specifický využitím v dokumentech typu smlouva.

Pole **Typ identifikátoru** vyjadřuje, zda je daný identifikátor v kontextu vždy osobním údajem (v tomto případě je označen jako primární) nebo se osobním údajem stává až při vzniku vazby na další identifikátor (v tomto případě je označen jako identifikátor sekundární). To si lze jednoduše představit na jednoduchém příkladu křestního jména a příjmení. Samo o sobě není křestní jméno tak silným identifikátorem, aby dokázalo identifikovat konkrétní osobu na kterou odkazuje. Ve spojení s příjmením se ovšem množina lidí, která může být tímto identifikátorem označena značně zmenšuje a v tomto kontextu lze tedy již prohlásit, že se jedná o osobní údaj dle platné definice.

Existují samozřejmě výjimky, kdy se jedním celým jménem (křestní jméno + příjmení) dá označit více osob, a naopak křestní jméno nebo příjmení může být tak unikátní, že je přiřazeno jen jedné osobě. Hranice může být v tomto případě velmi tenká. Tato práce se však ve své základní podobě zabývá standartními situacemi.

V poli **Závislé identifikátory (ID)** jsou v případě sekundárních identifikátorů uvedeny odkazy na další identifikátory, na kterých je v kontextu textu daný identifikátor závislý ve smyslu toho, že pokud daná vazba v textu existuje, stává se kombinace těchto identifikátorů osobním údajem.

V poli **Identifikátor specifický pro dokument typu smlouva** se nachází informace, zda je nutná detekce tohoto identifikátoru specifická přímo pro využití detektoru v kontextu smluv. Například identifikátor, který odkazuje na adresu bydliště je v kontextu běžného textu téměř vždy osobním údajem. V kontextu dokumentu typu smlouva zde však existuje závislost na tom, jestli se jedná o adresu-bydliště fyzické osoby či o adresu-sídlo právnické osoby. V druhém jmenovaném případě totiž tento údaj nepodléhá povinné anonymizaci.

To, jestli je identifikátor specifický jen pro smlouvy, je také důležité rozdělení jednotlivých detektorů do vhodných modulů tak, aby se v případě využití modulárního přístupu k vývoji aplikace daly některé obecné části znovu využít i v případě přepracování modelu na detekci osobních údajů v jiném kontextu.

V případě, kdy identifikátor nepodléhá povinné anonymizace, nejsou vyplněny hodnoty v žádném dalším sloupci kromě ID a název identifikátoru.

Kompletní seznam identifikátorů lze naleznout v příloze *A: Kompletní mapa kategorií osobních údajů*.

4.4 Mapa kategorií

V kompletním seznamu se pracuje s počtem 59 identifikátorů. Tento počet je v další fázi redukován pomocí postupného logického zjednodušování, kdy je bráno v potaz konkrétní využití aplikace na detekci osobních údajů na dokumentech typu smlouva. Také zde dochází k odstranění některých duplicitních příznaků u jednotlivých identifikátorů.

Kvůli své podstatě sekundárních identifikátorů vede tato mapa k následnému vytvoření n-rozměrné vazební matice binárních hodnot, kdy n je rovno celkovému počtu zpracovávaných identifikátorů.

Zjednodušenou mapu kategorií osobních údajů lze najít v tabulce 1.

Mapa kategorií osobních údajů			
ID	**Identifikátor**	**Typ identifikátoru**	**Závislé identifikátory (ID)**
Jmenné identifikátory			
1	Křestní jméno	sekundární	2, 7, 9, 53
2	Příjmení	sekundární	1, 7, 9, 53, 56
4	Jméno + Příjmení	primární	1, 2
6	E-mail s unikátním jménem	primární	
7	E-mail pracovní s obecným jménem	sekundární	1, 2, 9, 53
9	Přihlašovací údaje (login)	sekundární	1, 2, 7, 10, 32, 33
10	Název společnosti	-	-
Číselné identifikátory			
12	Rodné číslo/číslo pojištěnce	primární	
13	Číslo OP	primární	
14	Číslo pasu	primární	
15	Číslo platební karty	primární	
16	Číslo jiné karty	sekundární	vydavatel karty, 10
17	Číslo bankovního účtu	sekundární	59
19	IBAN	sekundární	59
21	Telefonní číslo	primární	
23	Číslo řidičského průkazu	primární	
24	Věk (sec)	sekundární	1, 2, 3, 53
25	SIPO	sekundární	59
Znakové identifikátory			
30	SPZ	sekundární	1, 2, záznam o jízdě
31	MAC adresa	primární	
32	IP adresa	sekundární	9, 10, 53
33	Doména	sekundární	9, 10, 53
38	VIN	sekundární	1, 2, záznam o jízdě
40	Sériové číslo	sekundární	1, 2, 10
42	Daňové identifikační číslo (DIČ)	sekundární	59
Lokalizační identifikátory			
45	Adresa – bydliště	primární	
46	Adresa – sídlo PO nebo FOP	-	-
Ostatní identifikátory			
49	Náboženské vyznání	sekundární	nepřímá závislost
52	Titul	sekundární	nepřímá závislost
53	Pracovní zařazení	sekundární	1, 2, 7, 10, 32, 33
54	Politická příslušnost	sekundární	nepřímá závislost
55	Státní příslušnost	sekundární	nepřímá závislost
56	Datum narození	sekundární	2, 3, 10
59	Právnická osoba (PO)/Fyzická osoba podnikající (FOP)	-	-

Tabulka 1: Mapa kategorií osobních údajů

4.4.1 Jmenné identifikátory

V této skupině došlo ke sjednocení všech identifikátorů vyjadřujících jméno osoby do 3 kategorií: křestní jméno, příjmení a celé jméno, které vzniká spojením 2 předchozích. Kvůli tomu, že všechna celá jména budou v konečném důsledku stejně podléhat anonymizaci, identifikátor označující jednatele/zástupce firmy nedává v samostatném významu smysl.

E-mail byl zařazen do 2 kategorií: E-mail s unikátním jménem (ať už pracovní nebo osobní) a e-mail s obecným jménem, který si lze představit například jako info@vse.cz. Ten se dá za osobní údaj uvažovat jen pokud jsou známé dodatečné informace jako například příjmení osoby.

Název společnosti sám o sobě anonymizaci nepodléhá. Je ho ovšem potřeba brát v potaz kvůli vazbám dalších sekundárních identifikátorů, které jsou na tomto údaji závislé.

4.4.2 Číselné identifikátory

Primární identifikátory uvedené v kompletním seznamu zůstávají principiálně ve stejné podobě, jen dochází ke sjednocení osobního a pracovního telefonního čísla, a především k vytvoření vazby na identifikátoru *Právnická osoba (PO)/Fyzická osoba podnikající (FOP)*.

Identifikátory označující identifikační číslo osoby (IČO) byly zcela odstraněny, protože tento údaj v daném kontextu nepodléhá anonymizaci.

4.4.3 Znakové identifikátory

Z této skupiny byl odstraněn identifikátor vyjadřující podpis. Vyvíjený program totiž nezpracovává obrazová data a k vyfiltrování podpisu v obrazové podobě by tak mělo dojít ještě v předchozím zpracování. Pokud je podpis uveden jen v textovém formátu, spadá již pod identifikátor vyjadřující jméno osoby.

Stejně tak byla vyřazena i textová reprezentace elektronického podpisu a klíče. U těchto údajů se totiž tento formát zobrazení nepředpokládá. Stejně tak jsou vyjmuta i biometrická data, která by se ve smlouvě vůbec neměla v žádné formě vyskytovat.

4.4.4 Lokalizační identifikátory

Adresa obecně vždy podléhá anonymizaci, pokud není v kontextu smlouvy zároveň sídlem PO nebo i FOP. Z toho se dá tedy odvodit, že tento identifikátor je ve své základní podobě primární, avšak dá se na něj pohlížet i jako na sekundární, přičemž jako sekundární anonymizaci nepodléhá.

4.4.5 Ostatní identifikátory

Identifikátor *Rasa* byl z této skupiny odstraněn kvůli velmi nepravděpodobnému výskytu ve smlouvách. To stejné platí i o identifikátoru *Sexuální vyznání*. *Pohlaví* bylo odstraněno

kvůli malé identifikační vypovídající hodnotě a zároveň kvůli složitostem při detekci pohlaví v českém jazyce.

Cena zakázky by měla být dle metodických doporučení uvedena přímo v předmětu smlouvy a anonymizaci tak podléhá jen ve velmi specifických případech, které jsou velmi složité a značně nekonkrétní. Cena může souviset i s obchodním tajemstvím, jehož detekce není v této verzi anonymizačního nástroje umožněna.

4.4.6 Redukovaný seznam identifikátorů

Po redukci kompletního seznamu identifikátorů se podařilo snížit celkový počet identifikátorů z 59 na 34. Do vytvořené mapy se dají v budoucnu při rozšíření stávajícího využití nebo při novém stanovení typu analyzovaných dokumentů přidávat i další identifikátory, které mohou být navázané na již stávající. Díky modulární struktuře vyvíjeného anonymizačního nástroje by mělo být možné relativně jednoduše rozšířit jeho funkcionalitu pro klasifikaci i dalších více doménově specifických identifikátorů.

5 Použité techniky hlubokého učení

Vyvíjený anonymizační nástroj je z velké části založený technikách hlubokého učení. V následující kapitole jsou tedy probrány vybrané oblasti hlubokého učení, a to od širších základů po konkrétnější témata, zabývající se přímo oblastí Named Entity Recognition (NER). Důraz je dále kladen na klasifikaci jmenných entit v českém jazyce a na doplňující morfologické znaky, přispívající lepší kvalitě klasifikace. Nakonec jsou zde zkoumány některé state-of-the-art modely, které jsou vhodné pro vlastní trénink v další části práce.

5.1 Hluboké učení

Hlubokým učením (Deep Learning) se nazývá podoblast strojového učení (Machine Learning). Hluboké neuronové sítě, tedy neuronové sítě patřící právě do oblasti hlubokého učení vynikají především větším stupněm abstrakce, která je způsobena větším počtem skrytých vrstev (hidden layers).

Známým příkladem, který ukazuje, jak větší počet skrytých vrstev dokáže ovlivňovat složitost požadované operace, je trénink neuronové sítě tak, aby dokázala provádět operaci XOR. Zatímco binární operace jako AND a OR jsou lineárně separovatelné v dvourozměrném prostoru jednou polorovinou, u XOR už toto tvrzení neplatí a pro úspěšné vyjádření této funkce je potřeba do neuronové sítě přidat skrytou vrstvu.

Zároveň je dokázáno, že jakákoliv funkce vyjádřená ve dvourozměrném prostoru může být neuronovou sítí simulována dostatečným počtem výpočetních jednotek (computation units) v pouze jedné skryté vrstvě, přičemž toto tvrzení vytváří paralelu například s Fourierovou transformací. Tento přístup je však pro praktické použití nevýhodný, protože s narůstající přesností simulace funkce roste exponenciálně i počet výpočetních jednotek ve skryté vrstvě a tím pádem i celková výpočetní náročnost (Wang, 2017).

Jako hluboké neuronové sítě jsou definovány takové neuronové sítě, které mají 3 a více skrytých vrstev a pracují s více úrovněmi reprezentace informací. Princip většího počtu skrytých vrstev je bližší fungování lidského mozku, který také tíhne k vytváření větší abstrakce a postupnému spojování dílčích informací (Bengio a Delalleau, 2011).

Dá se tedy zjednodušeně říct, že zatímco mělké neuronové sítě (s jednou skrytou vrstvou) se budou při úloze detekce automobilu v obrázku snažit hledat automobil jako celek, hluboké neuronové sítě budou v jednotlivých vrstvách nejdříve detekovat dílčí části automobilu, jako jsou kola, karoserie, a podobně.

Eldan a Shamir ve své práci *The Power of Depth for Feedforward Neural Netoworsk* (2015) zjistili a dokázali, že více skrytých vrstev neuronové sítě dosahuje z hlediska výpočetní náročnosti v poměru s přesností modelu lepší výsledky. Zároveň však mohou nastat situace, kdy je přidávání dalších vrstev spíše na škodu, protože model může v důsledku ve vyšších

vrstvách vytvářet i abstraktní vazby takové úrovně, že z hlediska řešeného problému nemusejí dávat smysl.

Zároveň obecně platí, že čím je model neuronové sítě hlubší, tím je i náročnější na trénování algoritmem zpětné propagace chyby (backpropagation algorithm), protože trénování se značně zpomaluje a zároveň se zvyšuje problém uváznutí v lokálních minimech kvůli špatné volbě počátečních vah. Také je v těchto případech pro trénování potřeba většího množství trénovacích dat (Hinton, 2006).

Tento problém adresoval v roce 2006 ve své práci *A fast learning algorithm for deep belief nets* Geoffrey Hinton, který přišel s greedy algoritmem, který umožňuje trénovat jednotlivé vrstvy odděleně, a tím otevřel cestu k řadě dalším výzkumům v oblasti hlubokého učení pomocí jemného ladění (fine tuning) ve vyšší vrstvách, bez přístupu k obrovskému množství trénovacích dat.

5.2 Natural Language Processing

Techniky patřící do oblasti Natural Language Processing (dále NLP) se zabývají strojovým zpracováním jazyka. Jedná se tedy o činnost, která je pro člověka relativně jednoduchá (ve většině případech), ale v případě strojového zpracování zde věda naráží na obtížnost vyjádření myšlenky, kontextu či emocí pomocí matematických pravidel (Johri et al., 2021).

NLP samo o sobě není přímou součástí hlubokého učení. Výzkum strojového zpracování jazyka začal ještě dávno před tím, než se objevily první záznamy o hlubokém učení a o hlubokých neuronových sítích. Nejprve se svět NLP dlouho zabýval vytvořením automatického překladače, který by dokázal spolehlivě automaticky překládat mezi různými jazyky. To vedlo k vytvoření časově náročného projektu, který si dal za cíl vytvořit překladové strojové slovníky. To se podařilo Georgesi Artstrounimu, který si nechal roku 1933 patentovat zařízení s názvem „Mechanical brain“, které sloužilo jako mechanický překladač (machinetranslate.org, nedatováno).

Tento přístup se ale z praktického používání v reálném světě ukázal jako nepoužitelný, protože tradiční slovníkový přístup si nedokáže poradit s gramatikou. V roce 1957 přišel Noam Chomsky s teorií o lingvistických strukturách, která měla obecně definovaným setem pravidel definovat matematické vztahy mezi jednotlivými slovy ve větách. Později se ale ukázalo, že tento přístup nedokáže nikdy komplexně vystihnout gramatiku kvůli syntaktickým rozdílům v jednotlivých jazycích (Johri et al., 2021).

Dalším evolučním krokem byla tokenizace slov ve větě, jejich klasifikace a zpracovávání kontextu dle pořadí tokenů ve stromové struktuře představující větu. Hojně se při tom využívala rekurze, kdy docházelo k postupnému zpracovávání všech tokenů, dokud nebyl rozpoznán význam textu. Taková technika NLP se nazývá Augmented Transition Network (ATN) a byla popsána roku 1970 Williamem Woodsem (Woods, 1970).

Všechny výše popsané metody měly jednu zásadní nevýhodu: všechna pravidla pro zpracování textů musela být přesně definována a algoritmicky popsána, což z tohoto úkolu

dělalo nesmírně časově a intelektuálně náročnou činnost (nehledě na to, že způsob, jak některá pravidla programaticky vyjádřit ani nebyl nikdy nalezen).

V 80. letech 20. století se oblast NLP začala pomalu prolínat se světem strojového učení, které pomocí například rozhodovacích stromů dokázalo přesněji interpretovat jednotlivá gramatická a syntaktická pravidla na základě tréninku modelu na reálných datech. Stále zde ale existovala velká oblast nevyřešených problémů s určováním významu slov na základě kontextu nebo rozdílného chápání textu na základě vztahu pisatele.

Nejnovějšími technikami NLP se tedy staly techniky hlubokého učení, které umožňují právě vyšší abstrakci chápání textu.

Klíčovou technikou se stalo vytváření matic reprezentujících vztahy mezi jednotlivými tokeny (těmi můžou být jak jednotlivá slova, tak například i dvojice slov, atd.). Samotné vytvoření této matice se provádí trénováním modelu na korpusu slov, které by se měly týkat stejné či příbuzné problematiky jako na kterou bude využitý i výsledný NLP model.

Běžnými technikami jsou například TF-IDF, One-Hot Encodings, Word Embeddings, Word2Vec a další. Cílem je vytvořit takovou n-rozměrnou matici, aby slova, která jsou si významem a využitím ve větách příbuzná, měla k sobě co nejbližší algebraickou vzdálenost (Nigam, 2021).

Základních typů neuronových sítí, které jsou v současné době považované jako nejvhodnější pro použití v NLP je několik. V první řadě jde o rekurentní neuronovou síť (Recurrent Neural Network – RNN), která dokáže zpracovat vstupní data (jedno slovo/jeden token) s ohledem na data zpracovaná při předchozím průchodu. Tím pádem dokáže při zpracování např. věty brát v potaz i kontext předchozích slov. Problém RNN spočívá v tom, že fungují pouze jednosměrně. To znamená, že slova následující ve větě později už nemohou ovlivnit zpracovaný význam slov předcházejících dříve ve větě. Zároveň zde existuje problém takzvaného mizejícího gradientu, kdy v delších textech postupně mizí vazba na předchozí zpracovaná data (Goldberg, 2017).

V současné době se tedy spíše využívají modifikované verze RNN, které tyto problémy adresují. Nejznámějšími jsou modely Gated Recurrent Unit (GRU) a Long Short Term Memory (LSTM), které řeší problém mizejícího gradientu představením takzvaných bran (gate), které umožňují lepší kontrolu nad oddělenou pamětí, představující kontext předchozích tokenů (Goldberg, 2017).

Problém nemožnosti vytvoření vazby na tokenech zpracovávaných později ve větě je řešen obousměrným RNN (Bidirectional RNN – BRNN). V tomto případě probíhá zpracovávání tokenů obousměrně, takže při zpracování konkrétního tokenu BRNN pracuje s gradienty obou RNN (Goldberg, 2017).

5.3 Named Entity Recognition

Named Entity Recognition (NER) je jedním z hlavních dílčích úkolů NLP, který představuje detekci a klasifikaci jmenných entit v analyzovaném textu. Tato technika má potom využití

jak sama o sobě, tak i jako předzpracování textu před dalšími NLP úlohami jako je porozumění textu, vytvoření shrnutí textu nebo strojového překladu (Li et al., 2020).

Jmenné entity jsou obecně rozdělovány do dvou skupin: generické jmenné entity (generic NE) a doménově specifické jmenné entity (domain-specific NE).

V literatuře lze identifikovat 4 hlavní přístupy v řešení NER (Li et al., 2020). Prvním je rule-based approach, založený na definici konkrétních pravidel pro každý typ jmenné entity. V tomto případě se často využívají prostředky jako jsou slovníky (dictionaries), regulární výrazy, syntakticky-lexikální vzory nebo ručně vytvořená syntaktická pravidla (Petasis et al., 2001). Systémy vytvořené tímto způsobem se vyznačují vysokou přesností (v terminologii strojového učení pojem *precision* = poměr true positives/(true positives + false positives)), ale malým recallem (poměr true positives/(true positives+ false negatives)) (Li et al., 2020).

Dalším přístupem je tzv. Unsupervised Learning neboli učení bez učitele, konkrétně potom využití clusteringu, kdy dochází k segmentaci jmenných identifikátorů do různých skupin dle obecně se vyskytujících lexikálních vzorů ve větším textovém korpusu (Nadeau, 2007). V případě unsupervised learning není potřeba anotovat data v trénovacím datasetu. Tím pádem je potřeba menší práce s přípravou trénovacích dat a tím i větší flexibilita při aplikování této techniky pro konkrétní doménu.

Přístup aplikující *Supervized Learning* neboli učení s učitelem je ve velké míře závislý na vektoru příznaků. Příznaky mohou být mnoha druhů. Některý vektor může označovat všechna velká písmena nebo číslice v textu, jiný speciální znaky či fráze s větší důležitostí (Settles, 2004), apod. Tyto vektory pak slouží jako abstraktní vrstva nad vektorem reprezentujícím samotný text. Klasifikátor je potom vytvořen jako multi-class (neuronová síť umožňuje klasifikaci několika jmenných entit) některým z tradičních algoritmů strojového učení, jako jsou rozhodovací stromy (decision trees), SVM (Support Vector Machine) nebo CRF (Conditional Random Fields) (Li et al., 2020).

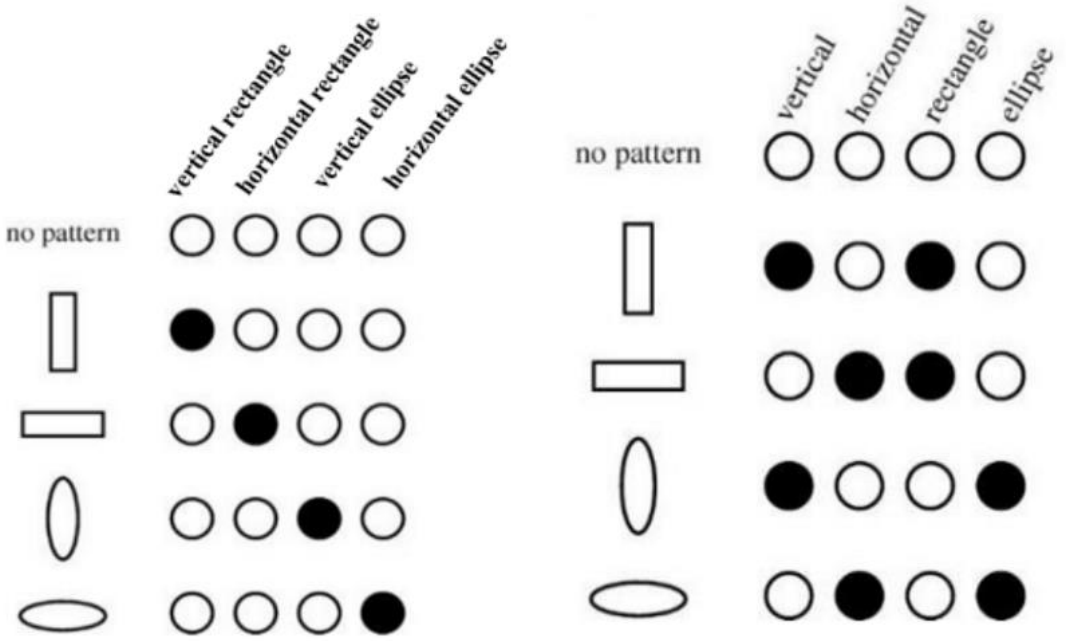
Posledním přístupem je využití algoritmů hlubokého učení. V tomto případě se jedná (ve většině případech) také o učení s učitelem. Oproti předchozímu přístupu se však využití hlubokého učení liší ve vytváření příznakových vektorů (tato část se běžně označuje jako feature engineering). Hluboká neuronová síť totiž dokáže odhalovat příznaky již sama. To značně zmenšuje nutnou odbornost v doméně zpracování daného textu, protože odpadá práce s odvozováním toho, jaké příznaky nejvíce souvisí s analyzovaným textem a jak je z textu derivovat.

Běžně využívaná architektura moderního NER implementovaného pomocí hlubokého učení je následující (jednotlivé části jsou uvedeny od vstupní vrstvy představující analyzovaný text po výstupní vrstvu, která představuje označené jmenné identifikátory v textu) (Li et al., 2020):

- Vrstva představující distribuovanou reprezentaci vstupních dat (distributed vector representation)
- Context encoder
- Tag decoder

Pod vrstvou představující **distribuovanou reprezentaci vstupních dat** si lze představit N-rozměrnou matici, která je uspořádána takovým způsobem, který reprezentuje vztahy mezi významy jednotlivých slov. Slova, která jsou si příbuzná, pak mají mezi sebou kratší vzdálenost.

Rozdíl mezi nedistribuovanou a distribuovanou reprezentací dat ukazuje obrázek 2: *Rozdíl mezi distribuovaným a nedistribuovaným rozdělením vstupních dat*. Vlevo je vidět řídká matice s nedistribuovanou reprezentací dat (one-hot vektor), ve které neexistuje žádný vztah mezi vstupními daty (geometrické obrazce). Vpravo se nachází matice s distribuovanou reprezentací, kde již lze vztahy mezi obrazci odvodit.



Obrázek 3: Rozdíl mezi distribuovaným a nedistribuovaným rozdělením vstupních dat (Ganesh, 2019)

Matice představující distribuované rozdělení vstupních dat se v kontextu NLP a NER nazývá word embeddings. Vrstvu word embeddings je nutné před zařazením do architektury NER neuronové sítě vytrénovat na vhodném korpusu tak, aby došlo ke správné distribuci reprezentace slov.

Správně vytrénovaná vrstva word embeddings se ukazuje v posledním vývoji NER jako klíčová. Moderní NER modely tedy pracují s word embeddings předtrénovaným na obrovských korpusech, které je následně pomocí techniky fine-tuning doučeno na konkrétnějších datech více závislých na dané doméně (Li et al., 2020).

Zajímavou alternativou (postavený na této architektuře je například populární model BERT) je využití architektury **Deep transformer**. Modely založené na této architektuře jsou v současné době považovány za state-of-the-art v oblasti NLP. Pomocí transformerů a techniky fine-tuning lze funkci předtrénovaných modelů založených na transformerech poupravit pro využití v NER (Giacaglia, 2021). Vrstva embeddings architektury transformer vykazuje větší citlivost na kontext než klasické word embeddings.

Context encoder, jak již název této části NER architektury naznačuje, se zabývá zpracováním kontextu. To znamená, že se tato část hluboké neuronové sítě stará o vyšší abstrakci nad textem předzpracovaným pomocí word embeddings. Vytvářený model v tomto případě vyjadřuje syntaktické, gramatické či významové závislosti, které se neuronová síť učí skrz supervised learning z anotovaných jmenných entit vyskytujících se v textu. Nejběžněji využívanými architekturami pro tuto část jsou obousměrné rekurentní neuronové sítě (BRNN), společně s jejich odvozenými variantami LSTM nebo GRU.

Výstupní vrstvou typické moderní NER architektury je **Tag decoder**. Ten si bere jako vstup kontextově závislou reprezentaci textových dat a produkuje původní text proložený anotovanými jmennými identifikátory.

Typicky se používá jednoduchý multi-layer perceptron se softmax vrstvou, případně pak Conditional Random Fields (CRF). Některé novější práce pak pracují i s RNN, případně potom s upravenou RNN nazývající se Pointer Network (Li et al., 2020).

5.4 NER v českém jazyce

Většina výzkumů týkajících se NER a NLP se tradičně provádí v angličtině (model je trénován na datasetech v anglickém jazyce). Dá se tedy říct, že jako state-of-the-art stav poznání současného NER se dají považovat modely, které se běžně evaluují na anglických korpusech.

Anglický jazyk se však dá svojí složitostí z pohledu sémantiky považovat jako spíše jednodušší. Celkové množství tvarů a variací jednotlivých slov (neboli morfologická složitost) je zde výrazně menší než v češtině. Stejně tak i například skladba vět je v češtině typicky složitější a vyžaduje větší povědomí o širším kontextu. Příkladem může být větší návaznost na rody podstatných jmen a na odkazování se na ně ve složitějších větách.

Dalším konkrétním příkladem může být porovnání celkového množství použitých slov v jejich různých tvarech na 2 korpusech, obou zhruba o velikosti 10 milionů slov, přičemž jeden z těchto korpusů je turecký a druhý anglický. Turečtina je přitom morfologickou složitostí češtině podobná. V anglickém korpusu lze dohledat celkové množství různých morfémů (tvarů slov) 97 734. V tureckém je to potom 474 957. Pokud se ale morfémy převedou do svého základního tvaru, vychází v obou těchto korpusech počet morfémů na hodnotu kolem 90 000. Průměrně tak na jeden základní tvar slova v turečtině vychází 5 různých morfémů (Demir, 2014).

Při trénování NER klasifikátoru operujícím na morfologicky bohatých jazycích, do kterých čeština (společně třeba se zvýše zmiňovanou turečtinou) patří, tak nabývá na důležitosti struktura trénovacích dat.

Ta totiž může ke každému jednotlivému slovu v datasetu nést i dodatečné informace, které mohou v konečném důsledku zpřesnit klasifikaci jmenných entit právě v případě výskytu většího množství formátů daného slova (Konkol a Konopík, 2014).

5.4.1 Stematizace a lematizace

Jednou z užitečných dodatečných informací, kterou mohou jednotlivá slova v datasetu nést je tzv. lemma. Toto označení popisuje slovo ve svém základním slovníkovém tvaru, tedy například lemmou výrazů *jsi, jsem* nebo *byli jste* je slovo *být*. Lematizací se potom nazývá proces vytváření lemm z původních morfémů.

Jako stematizace se označuje algoritmicky jednodušší proces převádění morfémů do základních tvarů, avšak bez dodatečné morfologické analýzy. Typicky se tedy jedná o odstraňování přípon a předpon slov a následné převádění do slovníkové podoby

Lematizace i stematizace jsou složitými operacemi, které se řeší například Porterovým algoritmem (použitelným pro angličtinu), rule-based algoritmy nebo různými statistickými modely. Jako příklad state-of-the-art nástroje využitelného pro lematizaci v češtině lze uvést nástroj MorphoDiTa (Konkol a Konopík, 2014).

5.4.2 POS

Part-of-speech tag, neboli POS označuje, je-li v datasetu dostupný, slovní druh každého morfému. I v tomto případě je většina dnešních taggerů (modelů umožňujících určování POS) implementována pomocí modelů strojového učení. Největší výzvu v tomto případě představuje fakt, že různá slova ve stejném tvaru mohou být v závislosti na kontextu rozdílným slovním druhem (například slovo *jez* může být v různém kontextu podstatným jménem nebo slovesem).

Ve světě NLP se typicky pracuje s POS tagy ve stejném, standardizovaném formátu, definovaném rámcem Universal Dependencies[10]. Tento rámec se snaží celosvětově sjednotit značení morfologických a sémantických příznaků v anotovaných datasetech.

5.5 Multilingvní modely

Využití multilingvních modelů, přesněji řečeno pak cross-lingual word embeddings je částečným řešením problému s nedostatečně velkými a kvalitními datasety v cílovém jazyce.

Typický přístup k trénování NLP modelu spoléhá na anotovaná data pro danou úlohu, kterou má výsledný model vykonávat. To vede k technice *supervised learning* (viz nadpis 5.3 Named Entity Recognition). Trénovací anotovaná data jsou potom typicky poskytována v jednom jazyce (Conneau et al., 2018).

V případě, kdy má model fungovat pro více jazyků současně (což je případ mezinárodně dostupných produktů jako například webových vyhledávačů) je ale potřeba, aby byl model natrénován zároveň pro více jazyků najednou. Vzhledem k tomu, že rozdílné jazyky, a především pak skupiny jazyků se vyznačují rozdílnou vnitřní logikou a sémantickými

[10] https://universaldependencies.org/

pravidly, mělo by při správném trénování tohoto modelu docházet k vytváření jazykově nezávislých příznaků (language-independent features) (Tsai et al., 2016).

Stejný přístup se dá využít i při trénování NLP modelu v jazyce, v kterém není dostupné větší množství anotovaných dat pro danou NLP úlohu (např. NER). Váhy multilingvního modelu potom mohou být využity v počátku trénování monolingvního jazykově specifického modelu a tím omezit celkově potřebné množství dat potřebných pro úspěšné natrénování modelu (Tsai et al., 2016).

Kapitolou samou o sobě je potom trénování multilingvních word embeddings. Tato běžně využívaná část modelu dokáže zachytit logické vztahy mezi jednotlivými slovy trénovacího datasetu. Velkou výhodou je, že kontextuální word embeddings se dá trénovat pomocí techniky *unsupervised learning*, tedy učením bez učitele. Tím vznikl prostor pro trénink word embeddings na obrovských neanotovaných korpusech, jako jsou například kompletní záznamy z wikipedie ve všech na wikipedii dostupných jazycích (Conneau et al., 2018).

Dále existuje rozdíl mezi kontextově závislými a nezávislými Word Embeddings. Modely jako Word2Vec[11] nebo GloVe[12] totiž vytvářejí kontextově nezávislou reprezentaci mezi jednotlivými zpracovávanými slovy. Modely typu BERT[13], ELMo[14] či podobné dokážou pracovat i s kontextem textu na úrovni vět a kontinuálních textových částí (Miaschi et al., 2020). Například pokud bude některé slovo znít stejně, ale bude mít více významů (tedy bude homonymem), ve word embeddings vygenerovaném modelem Word2Vec bude mít toto slovo stejnou vektorovou reprezentaci. Ve word embedding vygenerovaném modelem BERT však může mít stejné slovo v závislosti na významu takových reprezentací víc.

Z hlediska úlohy NER, ve které je širší kontext textu klíčový se tedy tato práce dále zaměřuje na modely, které pracují s kontextově závislými word embeddings, založenými na architektuře transformer.

5.5.1 mBERT

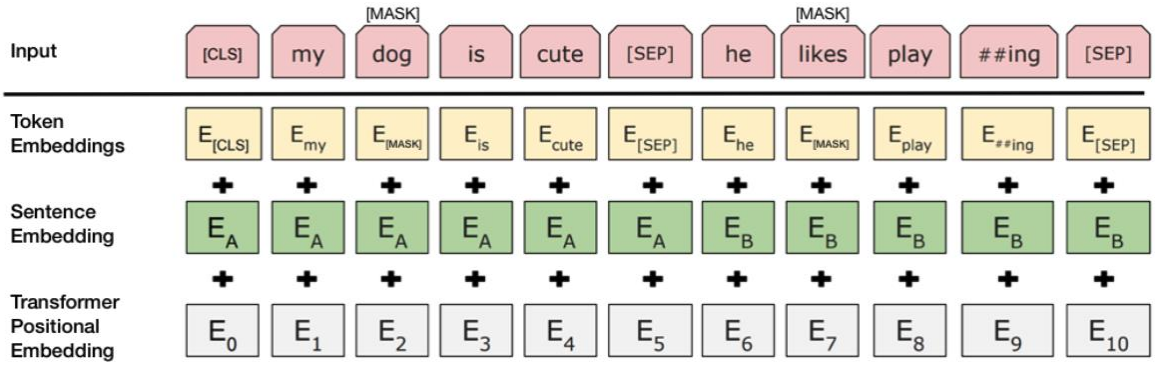
Poslední verze (6. 5. 2022) modelu mBERT[15], neboli multilingual BERT (Bidirectional Encoder Representations from Transformers) podporuje 104 jazyků (mezi nimi i češtinu), skládá se z 12 vrstev transformerů se společným počtem 768 skrytých vrstev.

Tento model byl vytrénován na korpusu obsahujícím kompletní texty z internetové encyklopedie Wikipedie.

Tento model (převážně tedy jeho ne-multilingvní varianta BERT) je považován jako současný state-of-the-art ve většině NLP úloh. BERT využívá ve své architektuře principu MLM, neboli Masked Language Modeling, kdy záměrně maskuje některá vstupy tak, aby

[11] https://github.com/tmikolov/word2vec
[12] https://github.com/stanfordnlp/GloVe
[13] https://github.com/google-research/bert
[14] https://github.com/HIT-SCIR/ELMoForManyLangs
[15] https://github.com/google-research/bert

model dokázal zvyšovat svojí schopnost abstrakce a vyhnul se tak jeho přeučení – viz *obrázek 4: Zpracování vstupu pomocí modelu mBERT* (Devlin et al., 2019).



Obrázek 4: Zpracování vstupu pomocí modelu mBERT (Devlin et al., 2019)

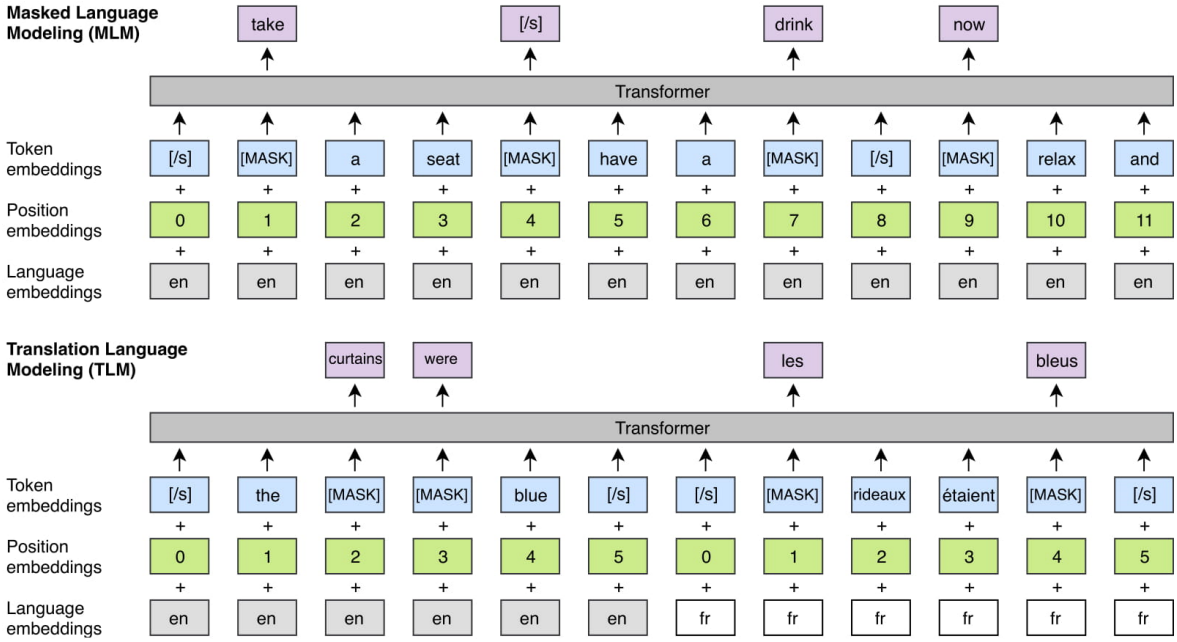
5.5.2 XLM

Model XLM[16] byl již vyvinut jako cross-lingual, tedy s cílem objevovat jazykově nezávislé příznaky. Mimo principu MLM ve své architektuře využívá i dalšího principu označeného jako Translation Language Modeling (TLM) – viz *obrázek 5: Zpracování vstupu modelem XML*. To ve své podstatě znamená, že model může při predikování využít další dostupný nezamaskovaný token, který je ovšem dostupný pouze v jiném jazyce. Tím vzniká větší provázanost mezi těmito jazyky (Lample a Conneau, 2019).

Tento model byl trénován na stejném datasetu jako předchozí model mBERT.

Nevýhoda tohoto modelu spočívá v tom, že jeho trénink není možný kompletně bez učitele. Vyžaduje totiž mít v datasetu označenou závislost mezi paralelními příklady tak, aby dokázal pomocí TLM zaměňovat jednotlivé tokeny.

[16] https://github.com/facebookresearch/XLM



Obrázek 5: Zpracování vstupu modelem XML (Lample a Conneau, 2019)

5.5.3 XML-RoBERTa

Model XML-RoBERTa [17]dělá oproti XML krok zpátky a zavrhuje TLM. Dělá to kvůli podpoře kompletního učení bez učitele.

V podstatě se tedy jedná o podobný princip jako u modelu mBERT, ovšem model dle svého jména používá architekturu RoBERTa. Ta funguje na stejném principu jako BERT, ovšem je řádově košatější. Dle původní práce *RoBERTa: A Robustly Optimized BERT Pretraining Approach* totiž původní model BERT podléhá underfittingu. RoBERTa tedy tento model rozšiřuje a trénuje na řádově větším datasetu. Zároveň však zvětšuje jeho velikost a zvyšuje výpočetní náročnost pro jak trénování modelu, tak i následnou inferenci. Celková přesnost modelu se však dá označit oproti modelu BERT jako vyšší (Liu et al., 2019).

Model XML-RoBERTa je tak po vzoru modelu RoBERTa vytrénován na datasetu CommonCrawl, který obsahuje až 250 000 unikátních tokenů (slov) (Conneau et al., 2020).

[17] https://github.com/facebookresearch/fairseq

6 Výběr technologií

Vzhledem k tomu, že cílem této práce není vytváření nových architektur NLP a NER modelů, je třeba vybrat nejvhodnější již existující technologie pro vytváření modelů hlubokého učení, které se pro tyto úlohy nejvíce hodí.

Existuje mnoho frameworků, které se používají pro vývoj modelů založených na hlubokém učení. Jako příklad některých nejpopulárnějších lze uvést například TensorFlow[18], Keras[19], Caffe[20] či PyTorch[21]. Tyto frameworky umožňují relativně jednoduše vytvářet, trénovat a evaluovat jakékoliv modely založené na hlubokém učení bez vlastní implementace nízko úrovňové logiky těchto modelů. Tato implementace totiž může být velmi složitá a je ve velké míře závislá na vhodně zvolených optimalizačních technikách či upravení celého trénovacího workflow na konkrétní HW.

Společnosti, které tyto frameworky vyvíjí, se do nic snaží neustále pronášet nejmodernější poznatky z oblasti AI, a tím pádem lze obecně říci, že pokud není vyvíjený model velmi specifického charakteru, je lepší některý z těchto frameworků použít, protože to znamená využít všechny state-of-the-art principy, které jsou zrovna dostupné.

6.1 NLP frameworky

Oblast NLP je v rámci hlubokého učení specifická typickým využíváním prvků, které se v ostatních oblastech (např. oblasti počítačového vidění) nevyužívají. V důsledku toho existují knihovny specificky zaměřené právě na NLP, které utilizují sílu předchozích zmiňovaných Deep Learning frameworků a přidávají ke své funkcionalitě i dodatečné funkce, které se zaměřují na NLP specifické části modelů, jako jsou například word embeddings či na sémantickou analýzu textu.

[18] https://www.tensorflow.org/
[19] https://keras.io/
[20] https://caffe.berkeleyvision.org/
[21] https://pytorch.org/

6.1.1 Kritéria pro výběr NLP frameworku

Při výběru NLP frameworku použitého pro vývoj prototypu anonymizačního nástroje jsou autorem práce určena následující kritéria:

- Framework musí mít dostatečně kvalitní a obsáhlou dokumentaci.
- Framework musí být volně dostupný (a pokud možno open source) pro nekomerční využití.
- Framework musí podporovat programovací jazyk C++ nebo Python (a v ideálním případě, pokud je framework open source by měla být většina jeho částí i v těchto jazycích implementována).
- Framework by mělo být možné využít ve vývojovém prostředí Google Colab out-of-the-box, tzn. měl by být dostupný jako samostatná knihovna, kterou není nutné pro toto vývojové prostředí samostatně kompilovat.
- Framework musí obsahovat co nejrozsáhlejší podporu úlohy NER.
- Framework musí podporovat rozšířenou znakovou sadu UTF-16 či podobnou, aby v něm bylo možné natrénovat model založený na českém jazykovém korpusu.

6.1.2 Stanford CoreNLP (Stanford NER)

Knihovna Stanford CoreNLP[22] obsahuje řadu nástrojů pro různé úlohy pojící se s NLP. Pro oblast NER, která je pro tuto práci nejvíce relevantní pak Stanford CoreNLP nabízí úžeji zaměřenou samostatně využitelnou knihovnu Stanford NER (která je jinak součástí větší CoreNLP knihovny).

Samotná knihovna Stanford NER je implementována v jazyce Java, nativně tedy podporuje tento programovací jazyk. Existuje však řada implementací nadřazených programových rozhraní pro řadu dalších programovacích jazyků, včetně Pythonu, pro který existuje samostatná knihovna Stanza[23], která je vyvíjená týmem zodpovědným i za Stanford NER a která využívá v pozadí právě knihovny Stanford NER.

Stanford NER podporuje již v základu fine-grained hierarchickou klasifikaci entit (viz 7.1.1 Granularita NER datasetů).

6.1.3 NLTK

NLTK[24] je NLP knihovna využitelná s programovacím jazykem Python (je součástí i balíčkovacího systému pip). Oproti Stanford CoreNLP je tato knihovna snadněji využitelná, ale méně vhodná například pro vytváření modelů s experimentálními architekturami.

[22] https://stanfordnlp.github.io/CoreNLP/
[23] https://stanfordnlp.github.io/stanza/
[24] https://www.nltk.org/

Chybí v ní podpora některých pokročilejších technik využitelných pro NER, jako je syntaktická analýza. U NLTK je ovšem dostupná kvalitní rozsáhlá dokumentace včetně mnoha příkladů konkrétního využití.

6.1.4 Flair

Flair[25] je jednou z novějších NLP knihoven. Je vyvinutá vývojovým týmem společnosti Zalando a v práci Sergeye Vychegzhanina a Evgenye Kotelnikova (2019), která porovnává přesnost klasifikace jmenných entit různých NLP knihoven při využití defaultních out-of-the-box modelů, dosahuje celkově nejkvalitnějších výsledků ze všech porovnávaných knihoven.

Flair je psaná v programovacím jazyce Python a důraz klade na možnost rozsáhlejších úprav vrstvy Word Embeddings. Zároveň poskytuje speciální podporu například pro zpracování biomedicínských datasetů.

Bohužel, dokumentace k této knihovně není moc rozsáhlá ani kvalitní a spíše připomíná ukázku využití jejich základního přiloženého modelu.

6.1.5 SpaCy

Knihovna SpaCy cílí na zákazníky, kteří tuto knihovnu mohou využít jako end-to-end řešení pro implementaci komerčních projektů (nicméně knihovna samotná je distribuována jako open source). Nabízí tedy i další funkcionality potřebné pro nasazení NLP modelu do komerčního prostředí, jako je relativně jednoduché vytváření kompletních NLP pipelines či rozsáhlou knihovnu předtrénovaných modelů využitelných out-of-the-box v komerčních aplikacích. Dá se tak říct, že SpaCy je celý ekosystém, v kterém je vše potřebné pro implementaci komplexního NLP řešení.

SpaCy[26] nabízí programovací rozhraní v jazyce Python, přičemž knihovna samotná je pak implementována kombinací Pythonu a Cythonu, což je programovací jazyk se syntaxí ve stylu Pythonu, avšak s větším důrazem na optimalizaci a správu paměti po vzoru programovacího jazyka C. Tím SpaCy dosahuje proti předchozím zmiňovaným knihovnám větší rychlosti (Vychegzhanin a Kotelnikov, 2019).

Dokumentace této knihovny je rozsáhlá a dostatečně kvalitní. Zároveň je zde kladen velký důraz na upravitelnost konkrétní implementace. Tato customizovatelnost je možná na různých úrovních znalostí programátora. Je tedy možné využít některé předdefinované komponenty, které je možné i upravit konkrétnímu use case a zároveň i vytvořit zcela nový model hlubokého učení dle vlastní experimentální architektury, který lze přímo vytvořit v závislé knihovně PyTorch.

[25] https://github.com/flairNLP/flair
[26] https://spacy.io

Zajímavá je také integrace s databází hlubokých modelů společnosti HuggingFace[27]. SpaCy tak umožňuje využít nejnovější state-of-the-art modely a upravit jejich funkčnost pro doménově specifické použití.

6.1.6 Porovnání

Porovnání frameworků z pohledů přesnosti klasifikace jmenných entit je obtížné. Ač se některé práce touto problematikou zabývají (Vchegzhanin a Kotelnikov, 2019; Al Omran et al., 2017), výsledek těchto porovnání nemá velkou vypovídající hodnotu. Porovnávají se zde totiž defaultní modely poskytnuté knihovnami, které nutně nemusí být totožné (např. spaCy v defaultním stavu nevyužívá architekturu transformer). Zároveň se v těchto pracech nepočítá se zpřesňujícími příznaky jako POS, LEMMAs či syntaktickou analýzou, a to z důvodu že ne všechna tato rozšíření vstupních dat nejsou všemi frameworky podporovány.

Obecně lze ale z těchto prací říct (ač zde platí zmiňovaná omezení), že framework spaCy je považován z pohledu klasifikace jmenných entit za nejrychlejší, flair za nejpřesnější a StanfordNLP za nejvíce používaný ve vědeckých kruzích.

Z pohledu této práce se jako nejvhodnější pro implementaci prototypu nástroje pro anonymizaci jeví framework spaCy, který se od ostatních odlišuje nejkvalitnější dokumentací a nejrozsáhlejšími možnostmi z hlediska úpravy kompletní klasifikační pipeline.

Na druhém místo lze dosadit framework flair, který je velmi nadějným adeptem pro další rozvoj a který podporuje řadu NER specifických funkčností (například kombinovatelnou vrstvu embeddings), ale který je v tuto chvíli špatně dokumentovaný. Alternativou také může být framework stanza.

6.2 Vývojové prostředí

Jako vývojové prostředí bylo zvoleno Google Colab. Toto cloudové řešení je nástrojem pro vývoj aplikací pracujících se strojovým učením, který nabízí připojení na vzdálené servery s dostatečným výkonem jak pro trénink nových modelů hlubokého učení, tak pro jejich následné využití (inferenci).

Google Colab nabízí připojení k virtuálním strojům, které v závislosti na zvolené variantě poskytují buď pouze CPU, kombinaci CPU a GPU nebo dokonce i TPU (Tensor Processing Unit). Toto prostředí má v sobě již předinstalované některé nástroje (např. knihovnu TensorFlow) vhodné pro hluboké učení, takže je příprava pracovní instance o to jednodušší.

Samotné IDE je potom téměř totožné s populárním IDE Jupyter Notebook a nabízí tedy bohaté formátování společně s možností spouštět kód po jednotlivých buňkách. Prostředí

[27] https://huggingface.co

lze propojit s Google osobním účtem a osobním úložištěm ve službě Google Drive, do kterého se dají při trénování modelů ukládat hotové prototypy.

Limitací tohoto prostředí je omezený přidělený výpočetní čas vzdáleného virtuálního stroje pro pracovní instanci. Tento čas není nikde definován a je vypočítáván dle vytížení zdrojů a času připojení k virtuálním stroji (a to ne jednorázově, ale v delším časovém horizontu).

Google Colab nabízí 3 druhy účtů: Colab, Colab Pro a Colab Pro+. Každý z těchto druhů má možnost využít i GPU, ovšem Colab Pro a Colab Pro+ nabízí rychlejší GPU a k tomu i více paměti RAM, společně s delším přiděleným výpočetním časem v jednom sezení (i když konkrétní časový interval stále není nikde definován). Účet Colab Pro+ nabízí navíc ještě možnost spouštět kód i v pozadí, bez nutnosti spuštěného a zobrazeného webového prohlížeče.

Pro tuto práci byl vybrán druh účtu Colab Pro. Ten sice oficiálně není v České republice podporovaný, nicméně si ho i tak lze zakoupit. Tím si lze lépe zajistit větší přidělený GPU výkon a více výpočetního času v jednom sezení.

7 Dataset

Výběr vhodného datasetu pro trénování a evaluaci vyvíjeného modelu je nejen nezbytnou, ale i klíčovou a velmi důležitou činností pro správné konečné fungování celého nástroje.

V kontextu této práce se jedná o dataset, který bude obsahovat anotované jmenné entity v českém jazyce a který tedy bude možné využít pro trénování modelu s učitelem. Distribuce kategorií jmenných entit by v nejlepším případě měla být v datasetu rovnoměrná, přičemž by dataset neměl obsahovat žádné chybně kategorizované entity (false positives).

Vzhledem k co největší snaze o modularitu vytvářeného nástroje je vhodné najít dataset s co nejširším záběrem klasifikovaných jmenných entit. V tom případě se totiž vytvářené klasifikátory budou moct později lépe upravovat i bez dalšího přetrénování modelu pro upravení nástroje pro konkrétní aplikaci. Kategorie entit by také měly být svým charakterem co nejblíže kategoriím osobních údajů (viz kapitola 4 Kategorizace osobních údajů).

V neposlední řadě by vhodný dataset měl obsahovat i dodatečné informace o jednotlivých morfémech, jako dříve zmiňované lemmy či POS tagy.

7.1 Anotace NER datasetů

7.1.1 Granularita NER datasetů

V případě běžně využívaných a veřejně dostupných datasetů, používaných pro úlohu Named Entitity Recognition lze datasety rozdělit do 2 základních kategorií dle granularity klasifikace jmenných entit.

Datasety s nižší granularitou jmenných entit se označují jako coarse-grained. Tento způsob anotace se dá obecně označit jako více běžný, nese však s sebou typicky problém nedostatečné možnosti rozlišení jmenných entit ve více specifikovaných aplikacích. Zpracování výstupu modelu, který je natrénován na coarse-grained datasetu je však jednodušší jak na logiku, tak na výpočetní náročnost.

Datasety s vyšší granularitou jmenných entit se označují jako fine-grained, neboli také hierarchické datasety. Tento způsob nabízí efektivnější rozdělení kategorií do funkčních celků a zároveň nabízí možnost kaskádování dalších modelů pro každý supertyp. Označení supertyp potom označuje hlavní kategorii, která může mít pod sebou ve stromové struktuře další příbuzné atomické jmenné entity (např. supertyp OSOBA může mít pod sebou další kategorie jako je křestní jméno nebo příjmení). Problém v tomto případě nastává, pokud jeden druh jmenné entity patří do více kategorií zároveň (Mai et al., 2018).

7.1.2 Způsoby anotace jmenných entit

Jmenné enity se běžně v datasetech anotují 2 způsoby. Prvním z nich je BIO (někdy označován taky jako IOB), což je zkratka pro Begin, In, Out. Tento způsob rozlišuje, zda je daný token jmennou entitou (pokud ne, je tokenu přiřazen tag O) a dále jestli je prvním tokenem jmenné entity (B) anebo pokračujícím tokenem (I).

Dalším způsobem je BILUO, což značí Begin, In, Last, Unit a Out. Tento způsob mimo předchozího značí i poslední token jmenné entity tagem L. Pokud se jmenná entita skládá pouze z jednoho tokenu, je jí potom přiřazen tag U. Dle práce Rationova a Rotha (2009) je tento způsob anotace vhodnější pro efektivnější trénování modelu, avšak novější práce jako je například článek *Named Entity Recognition Systém for Postpositional languages* od Kamrana a Mansoora (2016) tuto hypotézu vyvrací.

7.2 Dostupné datasety

7.2.1 Czech Named Entity Corpus (CNEC)

Asi nejznámější český dataset jmenných entit je v postupném vývoji již od roku 2007, kdy byl vytvořen jako součást práce *Named Entities in Czech: Annotating Data and Developing NE Tagger* (Ševčíková et al., 2007). Tento originální dataset je označován jako CNEC 1.0 a obsahuje 2000 manuálně anotovaných českých vět.

Od té doby došlo k rozšíření datasetu o další věty, POS tagy, lemmy i dodatečné morfologické informace. Poslední dostupná verze datasetu, označená jako CNEC 2.0 obsahuje 8993 manuálně anotovaných českých vět s celkovým množstvím 35220 jmenných entit (ÚFAL, nedatováno).

Tento dataset je proti ostatním českým NER datasetům specifický tím, že je hierarchický. Definované jmenné entity jsou v něm rozdělené do 8 supertypů, přičemž každý ze supertypů má pod sebou ještě další podkategorie. Celkově potom tento dataset obsahuje 46 kategorií atomických jmenných entit.

CNEC 2.0 je dostupný pod licencí Creative Commons BY-NC-SA 3.0, což znamená, že je možné ho volně použít pro jakékoliv nekomerční účely.

7.2.2 BSNLP

BSNLP je dataset, který je periodicky aktualizován před výročním workshopem *Balto-Slavic Natural Language Processing*[28], který tradičně pořádá soutěž o nejlepší multilingvní NER model, který dokáže pracovat s pobaltskými slovanskými jazyky. Ty čítají bulharštinu, češtinu, polštinu, ruštinu, slovenštinu a ukrajinštinu. Poslední vydaná verze datasetu je

[28] http://bsnlp.cs.helsinki.fi/

BSNLP-2021[29]. BSNLP-2021 rozlišuje pouze 5 základních typů jmenných entit, dodatečně však obsahuje i anotace lemm.

Dataset BSNLP-2021 je dostupný pod licencí Creative Commons BY-NC-SA 4.0.

7.2.3 SumeCzech-NER

Dataset SumeCzech-NER[30] je odvozený od velkého datasetu SumeCzech 1.0[31], který obsahuje až 1 milion českých novinových článků společně s jejich titulky a který je určen pro úlohu sumarizace textu.

SumeCzech-NER odkazuje na texty datasetu SumeCzech a přidává k nim jmenné entity, přičemž při tom rozlišuje 7 jejich kategorií (které se shodují se supertypy definovanými v CNEC 2.0 kromě typu Number expressions).

Anotace SumeCzech-NER není vytvořena manuálním způsobem, ale programaticky, což otevírá prostor pro více chybně definovaných či nenalezených jmenných entit v textu. Zároveň tento dataset neobsahuje žádné dodatečné informace jako jsou lemmy či POS tagy.

7.2.4 Shrnutí

Z dostupných českých anotovaných datasetů, vhodných pro úlohu NER se pro využití z hlediska této práce jeví jako nejvhodnější dataset CNEC 2.0., který odpovídá všem předem stanoveným kritériím. Zároveň je pro tuto práci vhodnější než řada běžně používaných anglických datasetů, a to právě díky většímu množství anotovaných kategorií jmenných entit.

Dataset je také neustále udržován Ústavem formální a aplikované lingvistiky (ÚFAL), což zároveň do jisté míry i zaručuje jeho kvalitu, především pak tedy jeho přesnost a komplexitu co se týče nejednostranného doménově specifického zaměření.

7.3 Zpracování datasetu

Vybraný dataset CNEC 2.0 je dostupný v několika formátech:

- **Html** – formát vhodný pro vizuálně efektivnější zobrazení anotovaných textů. Neobsahuje žádné dodatečné informace.
- **Plaintext** – formát, který obsahuje pouze anotované entity bez dodatečných informací. Tokeny, které nejsou jmennými entitami zde nejsou nijak značeny.
- **XML** – je anotován stejně jako plaintext, pouze v notaci XML.

[29] http://bsnlp.cs.helsinki.fi/shared-task.html
[30] https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-3505
[31] https://ufal.mff.cuni.cz/sumeczech

- **Treex** – proprietární formát určený pro zpracování pomocí nástroje Treex, vyvíjeného ÚFAL. Obsahuje dodatečné informace.

Speciální varianta datasetu CNEC_extended (pozor, dataset navzdory názvu o žádné další entity ani informace rozšířený není) je potom dostupná ve formátu **CONLL**, což je ve světě běžně používaný formát, který je spojený s výroční konferencí se stejným jménem. Tento formát obsahuje jak informaci o kategorii každého z tokenů, tak i dodatečné informace jako jsou lemmy. Problém je v tom, že CNEC_extended rozlišuje pouze coarse-grained kategorie, tedy vlastně jednotlivé supertypy definované v CNEC 2.0.

Knihovna spaCy vyžaduje pro trénování modelu buď vlastní proprietární binární formát (.spacy) nebo dataset převedený do JSONu podle definovaných pravidel. Současně ve verzi knihovny spaCy v3.0 je podporovaný binární formát. JSON formát je potom spojený s verzí knihovny v2.0, nicméně je novější verzí knihovny stále podporován.

SpaCy nabízí nástroje pro konvertování běžně se vyskytujících formátů datasetů do jejich zpracovávatelného formátu. Mezi těmito formáty se vyskytuje CONLL-2003 i CONLL-U, který obsahuje rozšířené morfologické informace.

7.3.1 Načtení CNEC_extended

Pro další zpracování byl do .spacy binárního formátu načten pomocí nástroje spacy convert dataset CNEC_extended (anotovaný způsobem BIO), který obsahuje menší množství kategorií jmenných entit (pouze supertypy).

7.3.2 Načtení CNEC 2.0

Kvůli potřebě většího počtu kategorií pro dosáhnutí větší modularity vyvíjeného nástroje byl dále dataset CNEC 2.0 (anotovaný způsobem BIO, obsahující hierarchické fine-grained kategorie) pomocí skriptu *treex2conll2003.pl*, dostupného jako součást práce *Neural Networks for Featureles Named Entity Recognition in Czech* (Straková et al., 2016) konvertován do formátu CONLL-2003.

Tento dataset ve formátu CONLL-2003 byl pomocí nástroje spacy convert s parametrem -c conll převeden do binární .spacy podoby, avšak bez dodatečných informací jako je lemma či POS. Spacy convert totiž tyto dodatečné informace při konvertování z CONLL-2003 neumí do svého formátu převést.

Pro načtení lemm je však možné postupně převést CNEC 2.0 dataset v CONLL-2003 formátu pomocí spacy convert s parametrem -t json do staršího spaCy JSON formátu a potom algoritmicky (pomocí funkce transform_lemmas) opravit chybně konvertovaná data tak, aby byly v JSON struktuře lemmy dostupné.

Zároveň je při této operaci potřeba odstranit speciální úpravy na lemmách, které jsou specifické pro CNEC 2.0 dataset a které dodatečně morfologicky rozšiřují lemmu i například o různé vysvětlivky či reference (Prague Dependency Treebank, nedatováno). Tyto dodatečné informace sice mohou být pro trénování modelu potenciálně velmi cenné, avšak

jsou jazykově specifické a knihovna spaCy není na zpracování těchto zvláštních morfologických atributů připravena.

Zpětným konvertováním pomocí spacy convert již v tomto případě lze dojít k použitelnému datasetu ve vyžadovaném binárním formátu .spacy s dodatečnou informací navíc ve formě potřebných lemm.

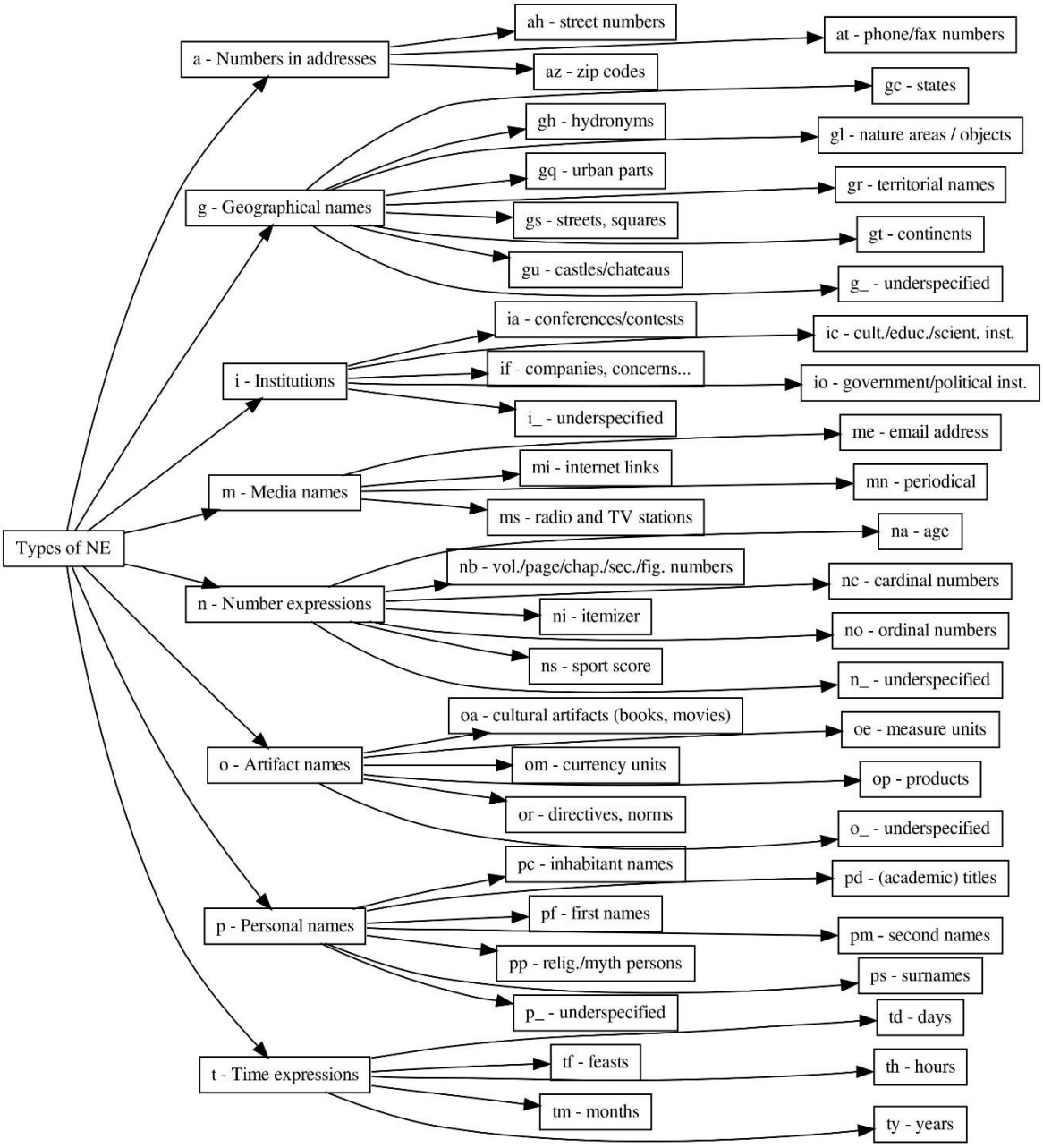
Při požadavku ke zpracování všech dostupných a pomocí knihovny spaCy zpracovatelných dodatečných informací, které dataset CNEC 2.0 nabízí, je potřeba načíst i POS tagy. Ty však nelze načíst při konverzi pomocí spacy nástrojů, pokud se zdrojový dataset nachází ve formátu CONLL-2003. Z tohoto důvodu byl dataset programaticky transformován do rozšířeného formátu CONLL-U. CNEC 2.0 poskytuje i POS tagy ve zvláštní rozšířené formě (stejně jako lemmy). Ta však není nijak standardizovaná (rámcem Universal Dependencies) a tedy ani dále zpracovatelná neproprietárními nástroji, vázanými přímo k CNEC 2.0 datasetu. Dalším krokem tedy došlo k namapování standartních POS tagů na CNEC 2.0 specifické tagy.

Bohužel, v posledním kroku, kterým mělo být převedení CNEC 2.0 v CONLL-U formátu pomocí spacy convert do .spacy binárního formátu, byla nalezena chyba ve zdrojovém kódu konverzního spacy nástroje, která zabraňuje správnému převedení BIO (i BILUO) anotací při využití zdrojového formátu CONLL-U.

Tuto chybu lze (k 8. 5. 2022) dohledat v spacy repositáři, a to v souboru spacy/spacy/training/converters/conllu_to_docs.py (Spacy GitHub, 2022). Nachází se v něm konstanta MISC_NER_PATTERN, která v sobě nese regulární výraz, který by měl sloužit pro detekci jmenných entit, ale který funguje jen pro detekci tokenu O, tedy pro token, který nese informaci, že daný morfém není jmennou entitou.

7.4 Analýza dat

7.4.1 Kategorie klasifikovaných jmenných entit



Obrázek 6:Hierarchie jmenných entit CNEC 2.0

Vybraný dataset CNEC 2.0 rozlišuje 46 kategorií jmenných entit, které jsou hierarchicky přiřazené pod 8 supertypů (viz obrázek 6).

V této práci však bylo rozhodnuto o využití coarse-grained struktury, tedy struktury, která je plochá a lze u modulárního nástroje lépe využít pro další přiřazení nadřazených recognizerů (viz kapitola 8 Vývoj modelu). Využití fine-grained struktury by přicházelo

v úvahu při implementaci hierarchických modelů, které by bylo možné načítat dle nadřazených recognizerů, to ale není součástí této práce.

Při analýze datasetu bylo zjištěno, že některé jmenné entity jsou zastoupené s minimální četností, a tedy u trénování modelu pro jejich klasifikaci může docházet buď k přeučení modelu nebo naopak k vytváření chybné reprezentace. Tento problém konkrétně nastává u jmenné entity mi, tf nebo i_, což bylo ověřeno i při evaluaci prvního natrénovaného modelu. Zároveň zde nastává problém s rozdělením datasetu na trénovací, testovací a evaluační část, kdy se absolutní hodnoty ještě více zmenšují. Konkrétní hodnoty absolutního a procentuálního zastoupení jmenné entity v datasetu lze vidět na obrázku 7.

7.4.2 Úprava datasetu

Jmenná entita *mi* vyjadřuje identifikované URL adresy. Celkově se však v datasetu vyskytuje pouze ve 22 případech, což je velmi málo pro správné vytrénování modelu. Tato jmenná entita (URL) je pro detekci osobních údajů důležitou, proto ji nelze zcela odstranit. Při trénování NER modelu však není tento dataset z tohoto pohledu dostatečně kompletní, proto je entita *mi* z datasetu pro další zpracování vyřazena. Potřeba klasifikace URL z pohledu anonymizace osobních údajů je potom tento problém adresován přidáním dalšího nadřazeného recognizeru (viz kapitola 8).

Jmenná entita *pp*, která představuje označení mýtické bytosti je při dalším zpracování z datasetu odstraněna. Pro žádané využití modelu je tento údaj spíše irelevantní a mohl by způsobovat chyby při tréninku modelu.

Jmenné entity kategorie *tf* (entita představující názvy svátků a významných dnů) ani kategorie i_ (která představuje instituce, které nejsou blíže specifikované) z datasetu odstraněny nejsou, a to kvůli možnému využití ve vyšších vrstvách nástroje. Také se u těchto entit předpokládá, že budou ve vyšších vrstvách sjednoceny s dalšími entitami pro detekci jednoho typu osobního údaje.

Seznam kategorií jmenných entit s kódovým označením jednotlivých kategorií, názvy kategorií, procentuálním zastoupením výskytu po úpravě datasetu a s doplňujícím vysvětlujícím popisem nebo příklady slov, které do dané kategorie patří je vyobrazen v tabulce 2.

Graf s procentuálním rozložením jmenných entit v upraveném datasetu lze potom vidět na obrázku 8. Z obrázku lze vidět, že rozložení stále není rovnoměrné. To je nicméně v případě datasetů určených k trénování NER modelů zcela běžná záležitost a pramení z podstaty nerovnoměrného výskytu určitých informací ve zdrojových textech.

Některé názvy kategorií jsou pojmenovány podle původní práce, ve které byl dataset navržen (Ševčíková et al., 2007). Další názvy a popisy jsou autorem upraveny nebo rozšířeny dle aktuálního stavu datasetu CNEC 2.0 v roce 2022.

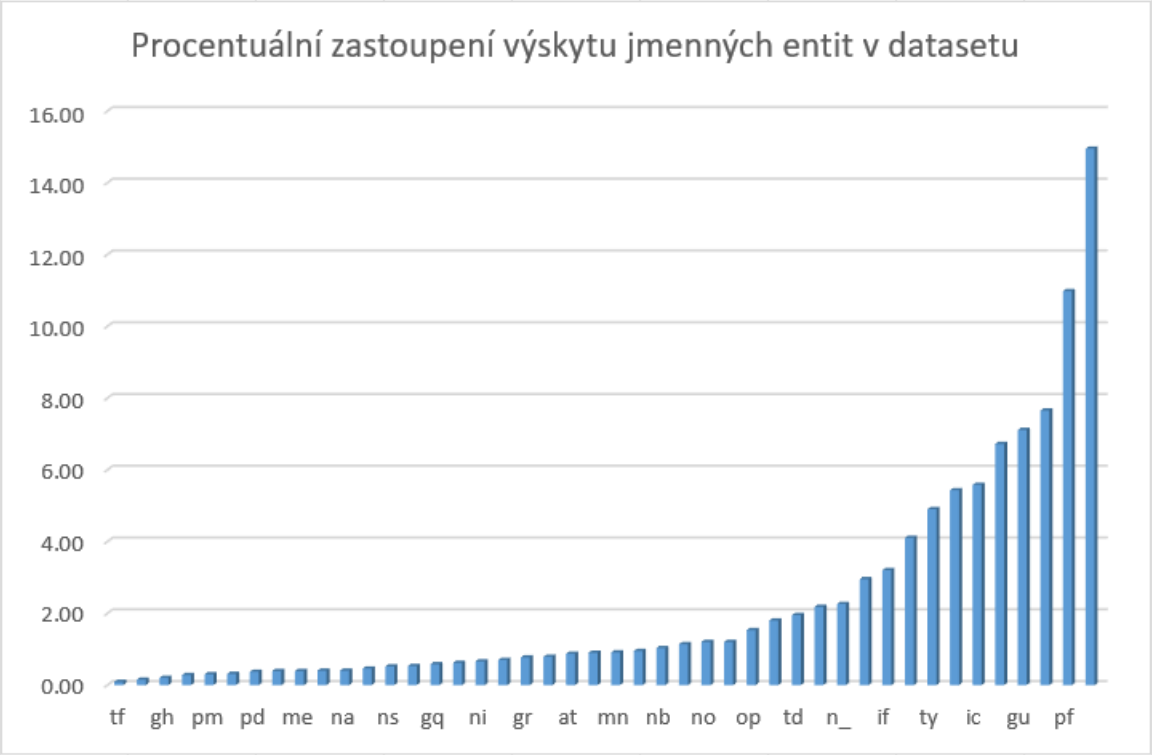
```
total detected named entity categories: 46
total named entites count: 26555
NE      count   percent
mi      22      0.08
tf      24      0.09
i_      40      0.15
gh      54      0.20
ms      73      0.27
pm      79      0.30
gt      81      0.31
pd      98      0.37
g_      102     0.38
pp      103     0.39
me      104     0.39
gl      106     0.40
na      106     0.40
az      121     0.46
ns      137     0.52
or      141     0.53
gq      153     0.58
o_      165     0.62
ni      175     0.66
ah      184     0.69
gr      204     0.77
om      209     0.79
at      229     0.86
ia      238     0.90
mn      240     0.90
pc      251     0.95
nb      273     1.03
gs      301     1.13
no      316     1.19
p_      317     1.19
op      404     1.52
oe      475     1.79
td      515     1.94
tm      577     2.17
n_      596     2.24
io      776     2.92
if      846     3.19
gc      1083    4.08
ty      1295    4.88
th      1436    5.41
ic      1474    5.55
oa      1776    6.69
gu      1878    7.07
nc      2021    7.61
pf      2901    10.92
ps      3856    14.52
```

Obrázek 7:Zastoupení jmenných entit v datasetu CNEC 2.0

Označení kategorie	Název kategorie	Procentuální zastoupení	Doplňující popis
tf	Svátky	0.09	Názvy různých svátků (Vánoce, Halloween,...)
i_	Nespecifikované instituce	0.15	Instituce nespecifikovaného konkrétního druhu
gh	Vodní útvar	0.20	Název jakéhokoli vodního útvaru (název řeky, jezera, moře,...)
ms	Stanice	0.28	Jméno TV nebo radio stanice
pm	Druhé jméno	0.30	Část jména osoby, která není křestním jménem či příjmením
gt	Kontinent	0.31	
pd	Titul	0.37	Titul, navázaný na jméno osoby
g_	Nespecifikovaný geografický název	0.39	
me	E-mail	0.39	
gl	Přírodní oblasti/útvary	0.40	Např. Krkonoše, Sibiř,...
na	Věk	0.40	
az	PSČ	0.46	Poštovní směrovací číslo
ns	Sportovní výsledek	0.52	Např. 2:0, vítězství Sparty,...
or	Předpis, norma	0.53	Např. sbírka zákonů, ISO 20000
gq	Místní název	0.58	Např. část obce, místní pojmenování místa,...
o_	Nespecifikovaný název věci	0.62	
ni	Položka seznamu	0.66	
ah	Číslo popisné	0.70	
gr	Menší územní celky	0.77	Např. kraje, oblasti,...
om	Měna	0.79	
at	Telefonní číslo	0.87	V datasetu se vyskytují i čísla faxu
ia	Přednáška, konference, soutěž	0.90	
mn	Periodikum	0.91	Může vyjadřovat i název redakce či mediální společnosti
pc	Obyvatelské jméno	0.95	Např. Afričan, Pražan,...
nb	Sekce	1.03	Např. kapitola, strana, vyjádření části celku
gs	Ulice	1.14	Názvy ulic a náměstí

no	Číslovky	1.20	
p_	Nespecifikované jméno osoby	1.20	
op	Výrobek, produkt	1.53	
oe	Jednotka	1.80	
td	Den	1.95	
tm	Měsíc	2.18	
n_	Nespecifikované číselné vyjádření	2.26	
io	Státní nebo politická instituce	2.95	
if	Firma	3.20	
gc	Stát	4.10	
ty	Rok	4.90	
th	Čas	5.43	
ic	Vzdělávací, kulturní nebo sportovní instituce	5.58	
oa	Umělecké dílo	6.72	Kniha, obraz, film,...
gu	Město, hrad, zámek	7.11	Praha, Konopiště,...
nc	Číselné vyjádření	7.65	
pf	Křestní jméno	10.98	
ps	Příjmení	14.95	

Tabulka 2: Seznam kategorií jmenných entit upraveného datasetu CNEC 2.0



Obrázek 8: Graf rozložení výskytu jmenných entit v upraveném datasetu CNEC 2.0

8 Vývoj prototypu

V následující kapitole bude postupně popsán proces návrhu a implementace prototypu anonymizačního nástroje. Tento proces se skládá z dvou hlavních částí.

V první části jde o vytrénování dostatečně kvalitního NER modelu založeného na technologii hlubokého učení, který bude sloužit jako hlavní komponenta anonymizačního nástroje.

V druhé části se tento model integruje do samotného anonymizačního nástroje. Tento nástroj bude umožňovat namapování doménově specifických jmenných entit relevantních pro danou aplikaci. Tímto způsobem je tento nástroj možné využít pro více druhů dokumentů i pro více kategorií jmenných entit. V případě této práce dojde k namapování na osobní údaje identifikované v kapitole 4 a vyobrazené v tabulce *1 Mapa kategorií osobních údajů*.

Pro dokonalejší funkci nástroje budou dále vyvinuty další samostatné klasifikátory, které budou dále utilizovat vyšší přesnost a kvalitu anonymizačního nástroje. Zároveň bude tímto ukázána modularita nástroje, která dokáže, že je možné funkčnost tohoto prototypu přenést na další druhy dokumentů a kategorií doménově specifických jmenných entit.

8.1 Trénování NER modelu

Pro trénování NER modelu bylo využito dříve zmiňovaného frameworku spaCy, který se dá nainstalovat přes Python balíčkovací systém pip včetně všech svých závislostí. To však platí pouze pro verzi knihovny, která utilizuje trénování pouze pro CPU. Vzhledem k tomu, že se dále v této práci využívá jak trénování jen pomocí CPU, tak ale i pomocí GPU, bylo potřeba doinstalovat i verzi spaCy s podporou CUDA a korespondující verzí knihovny PyTorch. V případě této práce byla vybrána verze CUDA 11.1 a k tomu kompatibilní verze PyTorch 1.10.1+cu111. Knihovna spaCy pak byla nainstalována s podporou modelů s architekturou transformer. Následně muselo dojít k exportu některých globálních proměnných pro další správné fungování frameworku.

```
!pip install torch==1.10.1+cu111 torchvision==0.11.2+cu111 torchaudio==0.10.1 -
f https://download.pytorch.org/whl/torch_stable.html
```

Zdrojový kód 8.1.1: Instalace CUDA + PyTorch

```
!pip install -U spacy[cuda111,transformers]
!export CUDA_PATH=/usr/local/cuda-11.1
!export PATH=/usr/local/cuda-11.1/bin${PATH:+:${PATH}}
!export LD_LIBRARY_PATH=/usr/local/cuda-11.1/lib64${LD_LIBRARY_PATH:+:${LD_LIBRARY_PATH}}
!pip install cupy-cuda111
```

Zdrojový kód 8.1.2: Instalace spaCy pro GPU

8.1.1 Konfigurace spaCy pipeline

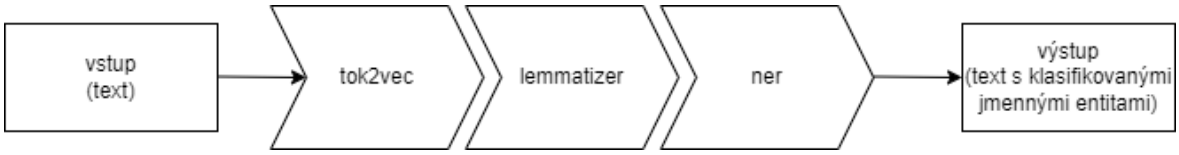
K vytvoření funkčního kompletního spaCy modelu připraveného k použití je potřeba vytvořit spaCy pipeline. Pipeline si lze představit jako řadu navazujících komponent, které postupně zpracovávají vstupní data (tedy nestrukturovaný text) a generují různé výstupy. Tyto výstupy se pak dají použít jako vstupy pro další komponenty, které jsou součástí pipeline nebo jako součásti konečného výstupu celé pipeline. Tímto způsobem může spaCy sloužit jako jediná knihovna potřebná pro vytváření kompletních komerčních end-to-end řešení.

SpaCy nabízí řadu mnoho hotových již předtrénovaných pipelines, které lze jednoduše využít pro řadu jednodušších use cases. Například v angličtině spaCy nabízí pipeline en_core_web_sm, která obsahuje komponenty typu tagger, parser, atrribute_ruler, lemmatizer a ner a je vhodná pro většinu běžných NER úloh v anglickém jazyce. Pro stroje, které dokážou utilizovat GPU pak nabízí pipeline en_core_web_trf, která má v sobě navíc komponentu transformer a dokáže využívat modelů založených na architektuře transformer.

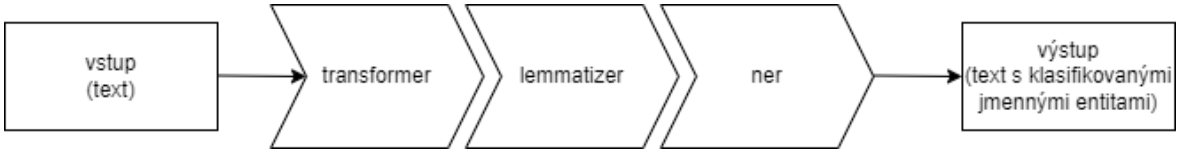
Žádnou pipeline určenou pro český jazyk spaCy nenabízí, bylo proto potřeba ji vhodně navrhnout a natrénovat od začátku na datasetu zpracovaném v předchozí kapitole.

Návrh pipeline lze provést ve verzi spaCy v3 pomocí speciálního konfiguračního souboru config.cfg, který si spaCy knihovna načítá před spuštěním trénovacího skriptu. V tomto souboru je nutné nadefinovat všechny klíčové parametry potřebné pro trénování modelu. To platí jak u každé jednotlivé komponenty modelu, tak i u globálních parametrů (těmi je například cesta k trénovacímu a testovacímu datasetu nebo batch size). Struktura konfiguračního souboru je tvořena řadou sekcí a podsekcí, které dohromady vytvářejí konkrétní architekturu všech jednotlivých částí pipeline.

Na obrázku 9 lze vidět zvolenou architekturu spaCy pipeline zvolenou pro trénování modelů utilizujících pouze CPU. Na obrázku 10 pak architekturu spaCy pipeline modelů utilizujících GPU. Tyto pipelines se liší v komponentě tok2vec a transformer. Tato představuje vrstvu word embeddings, přičemž v GPU modelu vrstva transformer využívá architektury modelu transformer a kontextuálního word embeddings (tedy fakticky spíše jen embeddings).



Obrázek 9: CPU spaCy pipeline



Obrázek 10: GPU spaCy pipeline

Vytvořené modely, trénované na datasetu bez dostupných lem pak ve své architektuře neobsahují komponentu lemmatizer (čímž se značně snižuje čas potřebný pro jejich vytrénování).

Konkrétní konfigurační soubory v plné délce, použité pro trénování jednotlivých modelů jsou dostupné v příloze B jako součást zdrojového kódu. V následující části tak budou popsány jen některé jejich části. Každý konfigurační soubor pak obsahuje řadu dalších zajímavých vlastností pipeline a jednotlivých komponent, které dále nejsou popsány, a to z důvodu zjednodušení přehlednosti a srozumitelnosti následujícího popisu.

V následujícím zdrojovém kódu (Zdrojový kód 8.1.3) lze vidět definované absolutní cesty ke zdrojovému datasetu. Také je zde vidět, že nebyly použité žádné předtrénované váhy pro komponentu tok2vec.

```
[paths]
train =
"/content/drive/MyDrive/DP/datasets/fine_grained_conll/spacy_lemmas/train_transformed.spac
y"
dev =
"/content/drive/MyDrive/DP/datasets/fine_grained_conll/spacy_lemmas/dtest_transformed.spac
y"
vectors = null
init_tok2vec = null
```

Zdrojový kód 8.1.3: Cesty k externím vstupním datům

Následující zdrojový kód obsahuje specifikaci, že trénink modelu bude probíhat s využitím GPU a knihovny PyTorch (Zdrojový kód 8.1.4)

```
[system]
gpu_allocator = "pytorch"
seed = 0
```

Zdrojový kód 8.1.4: Definice podřízené ML knihovny

Dále je potřeba specifikovat, že dochází k trénování pipeline v českém jazyce. Tím pak spaCy může vytvářet slovník pouze s českými výrazy. Zároveň je zde definováno pořadí jednotlivých komponent a batch size (počet vzorků využitých pro trénink jedné iterace modelu), která je v případě následujícího konfiguračního souboru určeného pro trénink modelu na CPU stanovena na 1000 vzorků (Zdrojový kód 8.1.5).

```
[nlp]
lang = "cs"
pipeline = ["tok2vec","trainable_lemmatizer","ner"]
batch_size = 1000
…
```

Zdrojový kód 8.1.5: Ukázka některých NLP globálních nastavení

Vlastnosti každé komponenty modelu je potom nutné dále detailněji specifikovat. Vše lze vidět na příkladu definice komponenty *ner*. V té je třeba definovat například objekt *scorer*,

který je odpovědný za evaluaci dané iterace modelu (jde tedy přeneseně řečeno o objekt, který spravuje cost funkci).

Každé komponentě modelu je také třeba přiřadit architekturu podřazeného modelu. U *ner* je to například spacy.TransitionBasedParser.v2, tedy knihovnou spaCy předdefinovaná architektura, která se stará o správné propojení jednotlivých tokenů (které již mohou být předzpracovány předchozími komponentami) tak, aby došlo ke kontext citlivé klasifikaci jmenných entit. Je také nutné definovat velikost skryté vrstvy, která byla v případě této komponenty stanovena na 64 jednotek (neuronů) (Zdrojový kód 8.1.6).

```
[components.ner.model]
@architectures = "spacy.TransitionBasedParser.v2"
state_type = "ner"
extra_state_tokens = false
hidden_width = 64
…
```

Zdrojový kód 8.1.6: Ukázka přiřazení vlastností podřazeného modelu *ner*

V komponentě *ner*, konkrétně potom v její embeddings vrstvě, která slouží ke zpracování vstupních vektorů je potřeba definovat s jakými příznaky bude tato vrstva pracovat. Jde tedy vlastně o vytvoření spojení s nižšími vrstvami (v případě této práce s vrstvou *lemmatizer*), které mohou předzpracovávat vstupní tokeny (jednotlivá slova vstupního textu). U každého tohoto příznaku se musí stanovit rozměr vstupního vektoru, který je typicky stejný jako rozměr výstupního vektoru z předřazené komponenty. Dá se zjednodušeně říct, že tímto nastavením se určují dostupné morfologické či syntaktické vlastnosti tokenu, které mají být brány v potaz při konečné klasifikaci jmenných entit komponentou *ner*. V následující ukázce je tedy vidět, že atribut „LEMMA“ je společně s atributem „NORM“, který označuje původní tvar tokenu (slova), příznakem při trénování komponenty *ner (*Zdrojový kód *8.1.7)*.

```
[components.ner.model.tok2vec.embed]
@architectures = "spacy.MultiHashEmbed.v2"
width = ${components.tok2vec.model.encode.width}
attrs = ["NORM","LEMMA","PREFIX","SUFFIX","SHAPE","SPACY"]
rows = [5000,5000,1000,2500,2500,50]
include_static_vectors = false
```

Zdrojový kód 8.1.7: Nastavení *embedding* vrstvy komponenty *ner*

Další zajímavou částí je popis samotného trénovacího procesu. V této části se dá pracovat například s hodnotou dropout rate, která zabraňuje přeučení modelu nebo s hyperparametrem learn_rate, který definuje velikost jednotlivých kroků při optimalizaci cost funkce. Dá se lze také nastavit optimalizační funkce (v případě této práce byl použitý optimizer Adam) (Zdrojový kód 8.1.8).

```
[training]
accumulate_gradient = 3
dropout = 0.1
patience = 1600
max_steps = 20000
eval_frequency = 200
…

[training.optimizer]
@optimizers = "Adam.v1"
beta1 = 0.9
beta2 = 0.999
L2_is_weight_decay = true
L2 = 0.01
grad_clip = 1.0
use_averages = false
eps = 0.00000001

[training.optimizer.learn_rate]
@schedules = "warmup_linear.v1"
warmup_steps = 250
total_steps = 20000
initial_rate = 0.00005
```

Zdrojový kód 8.1.8: Ukázka nastavení některých trénovacích parametrů aplikovaných pro celou pipeline

V neposlední řadě je zde potřeba zmínit načtení již předtrénovaného multilingvního modelu založeného na transformer architektuře v případě trénování pipeline utilizující GPU. V následující ukázce dojde k načtení modelu „bert-base-multilingual-uncased“. Tento model, více popsaný v části 5.5.1 mBert, je předtrénovaný na vícejazyčných textech z Wikipedie. Knihovna spaCy strukturu i váhy tohoto modelu stahuje z dříve zmiňované databáze modelů HuggingFace. Tato vrstva v případě této práce není uzamčená, a tedy při trénování pipeline dochází i k dalšímu trénování tohoto modelu na vstupním českém datasetu (Zdrojový kód 8.1.9).

```
[components.transformer.model]
@architectures = "spacy-transformers.TransformerModel.v3"
name = "bert-base-multilingual-uncased"
mixed_precision = false
```

Zdrojový kód 8.1.9: Definice předtrénovaného modelu mBert v komponentě *transformer*

Po doinstalování dodatečného balíčku spacy-lookups-data[32] je možné využít některé již vytvořené lookup tabulky, které se mohou týkat lematizace slov, ale i jiných fenoménů. V

[32] https://github.com/explosion/spacy-lookups-data

případě této práce je načtena i lookup tabulka pro normalizaci, tzn. tabulka, která dokáže sjednotit různé tvary jednoho slova se stejným významem (tedy např. gymnázium a gymnasium) (Zdrojový kód 8.1.10).

```
[initialize.lookups]
@misc = "spacy.LookupsDataLoader.v1"
lang = "cs"
tables = ["lexeme_norm","lemma_lookup"]
```

Zdrojový kód 8.1.10: Načtení externích lookup tabulek

8.1.2 Evaluace trénovaných modelů

V první fázi trénování modelů byly vytrénovány modely utilizující pouze CPU a tedy modely, které nejsou založené na architektuře transformer. Ačkoli ve finální verzi anonymizačního nástroje bude použitý model utilizující GPU, na těchto modelech, méně výpočetně náročných pro trénování bylo provedeno vyhodnocení připravených datasetů.

Natrénován byl tedy CPU model na coarse-grained datasetu derivovaném z CNEC_extended, model nevyužívající dodatečné morfologické informace lemma a model, který informace o lemmách využívá. Poslední 2 modely byly přitom vytrénovány na transformované vyčištěné verzi datasetu CNEC2.0.

Jako způsob pro evaluaci modelu vybráno kritérium F1 score, které vyjadřuje vztah mezi dalšími základními ML evaluačními kritérii Precision a Recall. Toto kritérium je při evaluování modelů velmi populární a dobře vyjadřuje obecnou kvalitu modelu, a to s menší závislostí na imbalanci datasetu než třeba základní kritérium Accuracy.

Bylo vybráno 5 předtrénovaných modelů založených na architektuře transformer. Každý z těchto předtrénovaných modelů byl vytrénován na jiném textovém korpusu.

Vytvořené modely **GPU_bert_uncased** a **GPU_bert_cased** používají známý model mBERT vytrénovaný na textech Wikipedie a jsou tedy multilingvními modely. Varianta modelu GPU_bert_cased je oproti GPU_bert_uncased[33] předtrénována na datasetu obsahujícím velká i malá písmena[34]. To je výhodné například z hlediska lepší detekce různých jmen.

Model **GPU_small_e_czech** používá v pozadí model Small-E-Czech[35], který je architekturou modelem ELECTRA a je předtrénovaný na datasetu Czech web corpus, který je spravován společností Seznam.cz. Tento model i dataset jsou poměrně nové, článek o nich byl publikován v prosinci 2021 (Kocián et al., 2021).

[33] https://huggingface.co/bert-base-uncased
[34] https://huggingface.co/bert-base-cased
[35] https://huggingface.co/Seznam/small-e-czech

Model **GPU_robeczech** je předtrénovaný na textech z různých českých periodik a článků o celkové velikosti až 450 milionů tokenů. Svou architekturou se jedná o model RoBERTa[36]. Tvůrci tohoto modelu tvrdí, že svojí kvalitou při použití pro české texty překonává známé neanotované multilingvní i další české datasety (Straka et al., 2021).

Model **GPU_czert_b_based** je založený na modelu Czert-B-base-cased[37]. Ten je vytvořený na kombinaci modelů BERT a ALBERT a natrénovaný je na menším, ale zato kvalitním datasetu, který obsahuje 340 000 českých vět. Dle autorů by měl ve většině NLP úloh předčít většinu multilingvních modelů (Sido et al., 2021).

V konfiguračních souborech všech trénovaných pipelines byla stanovena pevná hranice pro celkový maximální počet kroků (max_steps) 20 000 a posuvná hranice pro zastavení tréninku modelu v případě absence dalšího zlepšování (patience) 1600 kroků. Maximální počet epoch nebyl definován.

Pojmem jeden krok (step) se v tomto případě označuje 1 kompletní průchod neuronovou sítí pro danou várku (batch) z trénovacího datasetu. Pojem epocha potom označuje 1 kompletní průchod neuronovou sítí pro všechny vzorky z trénovacího datasetu.

V prostředí Google Colab nelze dopředu vyčíst, jak dlouho je možné využít aktivní připojení k jednomu virtuálnímu stroji. Může se tak proto stát, že dojde v průběhu tréninku k odpojení. Aby se v tomto případě zabránilo ztrátě již vypočtených vah modelu, je model v intervalech automaticky průběžně ukládán na Google Disk autora. Zároveň byla maximální délka trénování jednoho modelu stanovena na 8 hodin. Dále prezentovaná evaluace kvality jednotlivých vytrénovaných modelů je prováděna vždy na iteraci modelu, vykazujícího největší kvalitu při tréninku, přičemž platí následující pravidla:

- Došlo k dokončení tréninku modelu podle výše uvedených parametrů.
- Trénink modelu byl zastaven po 8 hodinách.

Evaluace každého modelu probíhá na speciálním evaluačním datasetu se vzorky nevyužitými pro samotné trénování modelu.

[36] https://huggingface.co/ufal/robeczech-base
[37] https://huggingface.co/UWB-AIR/Czert-B-base-cased

Evaluace NER modelů – porovnání				
Model	lemmatizer accuracy	ner precision	ner recall	ner F1 score
CPU_coarse		64,43	62,21	63,30
CPU_fine_nomorph		64,42	63,56	63,99
CPU_fine_lemmas	91,49	65,70	61,40	63,48
GPU_bert_uncased	92,39	79,08	79,53	79,30
GPU_bert_cased	**94,43**	**78,76**	**82,89**	**80,77**
GPU_small_e_czech	86,82	66,39	66,28	66,34
GPU_robeczech	-	-	-	-
GPU_czert_b_based	93,96	77,47	81,12	79,25

Tabulka 3: Evaluace NER modelů – porovnání

V tabulce 3 lze vidět výsledky evaluace jednotlivých modelů. Model GPU_robeczech se nepodařilo správně vytrénovat, a tedy konkrétní hodnoty evaluace tohoto modelu v této tabulce chybí.

Z výsledků evaluace lze vyčíst, že nejlépe uspěly multilingvní typy modelů GPU_bert_uncased a GPU_bert_cased, oba založené na modelu mBERT. Z těchto dvou pak lépe vychází varianta GPU_bert_cased, což vzhledem k charakteru klasifikovaných kategorií jmenných entit, které jsou často názvy institucí či jmény, dává smysl. GPU_bert_uncased má oproti GPU_bert_cased vyšší hodnotu metriky precision a menší hodnotu metriky recall. To znamená, že GPU_bert_uncased dokáže klasifikovat detekovanou jmennou entitu s vyšší úspěšností správného rozeznání konkrétní entity. GPU_bert_cased má potom větší schopnost samotné detekce jmenné entity. Z hlediska využití v anonymizačním nástroji je pro tuto práci důležitější parametr recall, protože je spíše důležité danou entity detekovat než správně kategorizovat. Krom toho byla jako hlavní metrika určená pro porovnání kvality všech modelů zvolená F1 score. Tato hodnota je vyšší u modelu GPU_bert_cased.

Dalším v pořadí je model GPU_czert_b_based, který je následován modelem GPU_small_e_czech. V tomto případě je tento výsledek taky spíše očekávaný. GPU_small_e_czech totiž využívá architektury modelu ELEKTRA, který je spíše modelem lehčím a známým spíše pro jeho rychlost. Ta v kontextu této práce není brána v potaz, a proto není v tabulce 4 uvedena (jde o vývoj prototypu nástroje s co největší přesností detekce a klasifikace). Přesto z dalšího porovnání vychází, že model GPU_small_e_czech je z vytrénovaných modelů nejrychlejší (metrikou je počet slov za vteřinu) a má nejlepší poměr NER F1 score a rychlosti.

O mnoho méně přesné jsou potom modely utilizované pro CPU. Jejich F1 score se pohybuje kolem hodnoty 63,5.

V dalším pokračováním práce byly na základě této analýzy zvoleny 2 modely: GPU_bert_cased pro GPU variantu a CPU_fine_nomorph pro CPU variantu.

Samotné binární soubory, představující vytrénované modely s vypočítanými finálními vahami jednotlivých výpočetních jednotek (neuronů) jsou dostupné na vyžádání od autora této práce. Soubory nejsou součástí ani přílohy, ani autorova GitHub repositáře[38], a to kvůli své velikosti a kvůli omezené velikosti dostupných úložišť (GitHub a Insis).

8.2 Návrh a implementace anonymizačního nástroje

Pro vývoj anonymizačního nástroje byl použit open source software development toolkit (SDK) Presidio[39]. Ten nabízí v programovacím jazyce python nástroje na vytvoření aplikace sloužící k anonymizaci osobních údajů.

8.2.1 SDK Presidio

Presidio nabízí 3 moduly: *analyzer*, *anonymizer* a *image redactor*.

Analyzer slouží k detekci osobních údajů ve vstupním textu. K této operaci přitom používá řadu definovaných python objektů, nazývajících se recognizery. Každý z těchto recognizerů dokáže v textu klasifikovat jednu či několik jmenných entit. Recognizery mohou fungovat na různých principech, jako je například detekce entit pomocí regulárních výrazů nebo pomocí slovníků. Presidio nabízí k využití buď několik již hotových recognizerů od jeho tvůrců (vhodných pro jazykově nezávislé entity typu IP adresa) nebo možnost vytvořit si recognizery vlastní, dle potřeby doménově specifického využití vyvíjené aplikace.

Klíčovým typem recognizeru je potom recognizer založený na hlubokém učení. Ten je potřeba vytvořit tak, aby mohl využívat v pozadí některého z podporovaných NLP frameworků. V případě této práce jde tedy o jakýsi wrapper na NER model vytrénovaný v předchozí části, který využívá *AnalyzerEngine* pro spouštění klasifikace každého vstupního tokenu knihovnou spaCy.

Dále je nutno zmínit, že Presidio nemá žádnou podporu českého jazyka. Ačkoli je tedy možné využít některé předdefinované jazykově nezávislé recognizery, je potřeba naimplementovat několik dalších jazykově závislých a aplikačně specifických recognizerů tak, aby byl splněn požadavek na anonymizaci důležitých osobních údajů běžně se vyskytujících ve smlouvách nahrávaných do Veřejného registru smluv.

Anonymizer slouží k anonymizaci osobních údajů, poskytnutých předchozím modelem *analyzer*. Nabízí přitom různé možnosti anonymizace, jako je prosté začernění údajů či pseudonymizace. V kontextu této práce je v navrženém nástroji použitá pouze základní funkčnost tohoto modulu, která nahrazuje ve vstupním textu konkrétní jmennou entitu jménem klasifikované kategorie.

[38] https://github.com/ondrasekd/DP
[39] https://github.com/microsoft/presidio

Image redactor je v tuto chvíli spíše experimentální částí Presidia, která slouží k anonymizaci osobních údajů v obrázcích. V kontextu této práce není tento modul využit.

Před využitím Presidia je potřeba nejprve nainstalovat pomocí balíčkovacího systému pip moduly presidio_analyzer a presidio_anonymizer. Následně, aby SDK Presidio mohl správně referencovat vytrénovaný NER model, je potřeba z vytrénované pipeline vygenerovat pomocí nástroje spacy package python balíček, které je nutné dále znovu pomocí balíčkovacího systému pip nainstalovat (Zdrojový kód 8.2.1).

```
!pip install presidio_analyzer
!pip install presidio_anonymizer

!python -
m spacy package /content/drive/MyDisk/PIIAnonymizer/models/CPU_fine_lemmas ./packages --
name CPUFineLemmas
!pip install /content/packages/cs_CPUFineLemmas-0.0.0
```

Zdrojový kód 8.2.1: Instalace Presidia a vytvoření balíčku z vytrénované NER pipeline

8.2.2 Vývoj a přiřazení recognizerů

Pro integraci vytrénovaného NER modelu do SDK Presidio byl vytvořen wrapper *SpacyRecognizerCustom*, který zachytává výsledky klasifikace jmenných entit z podřazeného spaCy modelu a přiřazuje jednotlivým kategoriím správné konečné labely na základě analýzy typů osobních údajů na začátku této práce. Přiřazení fine-grained kategorií lze vidět na ukázce (Zdrojový kód 8.2.2). V případě využití tohoto nástroje pro jinou doménově specifickou aplikaci (tedy ne detekce osobních údajů ve smlouvách) lze tuto tabulku upravit tak, aby odpovídala konkrétním analyzovaným dokumentům. Tím je dosaženo jednoho z pilířů modularity tohoto nástroje.

```
CHECK_LABEL_GROUPS = [
    ({"PERSON"}, {"pd", "pf", "pm", "ps"}),
    ({"EMAIL_ADDRESS"}, {"me"}),
    ({"LOGIN_NICK"}, {"p_"}),
    ({"INSTITUTION"}, {"ia", "ic", "if", "io", "i_"}),
    ({"PHONE_NUM"}, {"at"}),
    ({"MEDIA_NAME"}, {"mn", "ms"}),
    ({"NUMBER_EXPR"}, {"nb", "nc", "ni", "no", "ns", "n_"}),
    ({"LOCATION"}, {"ah", "az", "gc", "gh", "gl", "gq", "gr", "gs", "gt", "gu", "g_"}),
    ({"PRODUCT"}, {"op"}),
    ({"DATE_TIME"}, {"td", "tf", "th", "tm", "ty"}),
    ({"OTHER"}, {"oa", "or", "o_", "pc"})
]
```

Zdrojový kód 8.2.2: Ukázka možného nastavení konečné kategorizace jmenných entit v závislosti na vytrénovaném NER modelu

Společně s recognizerem *SpacyRecognizerCustom* byly vyvinuty jako ukázka možného dalšího modulárního rozšiřování nástroje i další doménově specifické recognizery. Tyto

doménově specifické recognizery mohou sloužit i jen jako doplňující prvek anonymizačního nástroje, který zvyšuje konečnou přesnost klasifikace. Například vyvinutý recognizer *CSRCRecognizer*, který slouží pro klasifikaci rodného čísla ve své podstatě není pro aplikaci na klasifikaci osobních údajů ve smlouvách nutně potřebný. Rodná čísla totiž dokáže klasifikovat i zmiňovaný předchozí recognizer *SpacyRecognizerCustom* a to s labelem „NUMBER_EXPR“. Pro zvýšení přesnosti a pro zvýšení rozlišovací schopnosti anonymizačního nástroje však byl tento recognizer jako součást prototypu naimplementován a dokazuje tím existenci druhého pilíře modularity tohoto nástroje. Tedy možnosti relativně jednoduše vytvářet doménově specifické recognizery pro konkrétní aplikaci anonymizačního nástroje.

CSRCRecognizer využívá definované délky a formy rodného čísla a využívá pro jeho spolehlivou detekci regulární výraz (Zdrojový kód 8.2.3).

```
PATTERNS = [
    Pattern("rodne cislo (high)", r"\d{2}(0[1-9]|1[0-2]|5[1-9]|6[0-2])(0[1-9]|1[0-9]|2[0-9]|3[0-1])\/?\d{3,4}", 0.5)
]
```

Zdrojový kód 8.2.3: Regulární výraz použitý pro detekci rodného čísla

V případě, ve kterém by se vyvíjený anonymizační nástroj stal komerčním produktem, by pravděpodobně došlo k dalšímu vývoji řady dalších doménově specifických recognizerů, které by si konečný uživatel (tedy někdo, kdo potřebuje uspokojit potřeby anonymizace osobních údajů) mohl volitelně zapnout. Zároveň by měl stejný koncový uživatel možnost vytvořit si vlastní pravidla a implementovat si tak zcela vlastní a zcela specifické recognizery.

Ve vyvíjeném nástroji byly dále použité některé předdefinované jazykově nezávislé recognizery, které nabízí SDK Presidio. Těmito recognizery jsou:

- CreditCardRecognizer – detekce čísel platebních karet
- CryptoRecognizer – detekce adres bitcoinových peněženek
- EmailRecognizer – detekce e-mailových adres
- IbanRecognizer – detekce čísla IBAN
- IpRecognizer – detekce IP adresy
- UrlRecognizer – detekce internetové domény

Například využití předdefinovaného *EmailRecognizer*u je v případě anonymizačního nástroje vhodnější, a to proto, že ve zdrojovém datasetu, na kterém byl trénován NER model není výskyt jmenných entit vyjadřujících e-mailovou adresu dostatečný. Zároveň lze e-mailová adresa typicky dobře detekovat přes správně vytvořený regulární výraz.

8.2.3 Namapování kategorií osobních údajů na Presidio recognizery

V tabulce 4 lze vidět kategorie jmenných entit, které byly sumarizovány v části 4.4 *Mapa kategorií,* doplněné o konkrétní jim přidělené recognizery a výsledné labely, kterými budou tyto údaje označovány v anonymizačním nástroji. Jak je z tabulky vidět, ke všem identifikovaným osobním údajům je přiřazen některý recognizer a výsledný label,

s výjimkou identifikátoru 53 – pracovní zařazení. Tento typ jmenné entity by byl vhodný pro klasifikaci pomocí recognizeru pracujícího s hlubokým učením, bohužel ale zdrojový dataset neobsahoval žádné anotace, které by tuto jmennou entitu popisovaly.

Mapa kategorií osobních údajů – výsledná kategorizace				
ID	**Identifikátor**	**Typ identifikátoru**	**Přiřazený Presidio recognizer**	**Přiřazený výsledný label**
Jmenné identifikátory				
1	Křestní jméno	sekundární	SpacyRecognizerCustom	PERSON
2	Příjmení	sekundární	SpacyRecognizerCustom	PERSON
4	Jméno + Příjmení	**primární**	SpacyRecognizerCustom	PERSON
6	E-mail s unikátním jménem	**primární**	EmailRecognizer	EMAIL_ADDRESS
7	E-mail pracovní s obecným jménem	sekundární	EmailRecognizer	EMAIL_ADDRESS
9	Přihlašovací údaje (login)	sekundární	SpacyRecognizerCustom	LOGIN_NICK
10	Název společnosti	-	SpacyRecognizerCustom	INSTITUTION
Číselné identifikátory				
12	Rodné číslo/číslo pojištěnce	**primární**	CSRCRecognizer	CS_RC
13	Číslo OP	**primární**	SpacyRecognizerCustom	NUMBER_EXPR
14	Číslo pasu	**primární**	SpacyRecognizerCustom	NUMBER_EXPR
15	Číslo platební karty	**primární**	CreditCardRecognizer	CREDIT_CARD
16	Číslo jiné karty	sekundární	CreditCardRecognizer	CREDIT_CARD
17	Číslo bankovního účtu	sekundární	SpacyRecognizerCustom	NUMBER_EXPR
19	IBAN	sekundární	IbanRecognizer	IBAN_CODE
21	Telefonní číslo	**primární**	SpacyRecognizerCustom	PHONE_NUM
23	Číslo řidičského průkazu	**primární**	SpacyRecognizerCustom	NUMBER_EXPR
24	Věk (sec)	sekundární	SpacyRecognizerCustom	NUMBER_EXPR
25	SIPO	sekundární	SpacyRecognizerCustom	NUMBER_EXPR
Znakové identifikátory				
30	SPZ	sekundární	SpacyRecognizerCustom	NUMBER_EXPR
31	MAC adresa	**primární**	SpacyRecognizerCustom	NUMBER_EXPR
32	IP adresa	sekundární	IpRecognizer	IP_ADDRESS
33	Doména	sekundární	UrlRecognizer	DOMAIN
38	VIN	sekundární	SpacyRecognizerCustom	NUMBER_EXPR
40	Sériové číslo	sekundární	SpacyRecognizerCustom	NUMBER_EXPR
42	Daňové identifikační číslo (DIČ)	sekundární	SpacyRecognizerCustom	NUMBER_EXPR
Lokalizační identifikátory				
45	Adresa - bydliště	**primární**	SpacyRecognizerCustom	LOCATION
46	Adresa - sídlo PO nebo FOP	-	SpacyRecognizerCustom	LOCATION
Ostatní identifikátory				
49	Náboženské vyznání	sekundární	SpacyRecognizerCustom	OTHER
52	Titul	sekundární	SpacyRecognizerCustom	PERSON

53	Pracovní zařazení	sekundární	-	-
54	Politická příslušnost	sekundární	SpacyRecognizerCustom	OTHER
55	Státní příslušnost	sekundární	SpacyRecognizerCustom	OTHER
56	Datum narození	sekundární	SpacyRecognizerCustom	NUMBER_EXPR
59	Právnická osoba (PO)/Fyzická osoba podnikající (FOP)	-	Může být vyčteno z metadat	

Tabulka 4: Mapa kategorií osobních údajů - výsledná kategorizace

8.3 Grafické rozhraní

Aby byl vyvíjený anonymizační nástroj dobře prezentovatelný, byla společně s ním vyvinuta jednoduchá grafická webová aplikace, ve které je možné naživo sledovat proces anonymizace vstupního textu.

Pro vývoj této grafické webové aplikace byl použit open source framework Streamlit[40], ve kterém je možné psát v programovacím jazyce Python jednoduché grafické aplikace s použitím poskytovaných grafických komponent. K tomu, aby šel Streamlit framework použít v prostředí Google Colab, je potřeba nainstalovat správnou verzi 1.7.0, která používá shodnou verzi pythonu jako Google Colab (updatovat v Google Colab verzi pythonu není dobře možné, protože je potom vyžadován restart virtuálního stroje) (Zdrojový kód 8.2.4).

```
!pip install streamlit==1.7.0 pandas
```

Zdrojový kód 8.2.4: Instalace frameworku Streamlit do prostředí Google Colab

Následné spuštění Streamlit aplikace probíhá v prostředí Google Colab přes tunelování portu 8501 skrze utilitu localtunnel. Ta umožňuje jednoduše sdílet webovou aplikaci běžící na localhost na vzdálenou serverovou instanci, ke které je následně přiřazeno URL (Zdrojový kód 8.2.5).

```
!streamlit run /content/DP/src/streamlit_app/presidio_streamlit_GPU_best.py & npx localtun
nel --port 8501
```

Zdrojový kód 8.2.5: Spuštění Streamlit aplikace v prostředí Google Colab

Samotná Streamlit aplikace má dostupné zdrojové kódy v příloze B ve složce streamlit_app. Nejdříve dojde k importování dependencies. Mezi nimi jsou třeba předem nainstalované knihovny jako spaCy nebo Presidio, ale i běžnější python moduly jako pandas. Zároveň je potřeba naimportovat i přímo knihovnu Streamlit (Zdrojový kód 8.2.6).

[40] https://streamlit.io/

```
import pandas as pd
import streamlit as st

from presidio_analyzer import AnalyzerEngine
from presidio_anonymizer import AnonymizerEngine
import spacy
```
Zdrojový kód 8.2.6: Python moduly použité v PIIAnonymizer Streamlit aplikaci

Následně dojde k načtení NER pipeline a definování všech výše zmiňovaných recognizerů, včetně vlastních doménově specifických.

Je nutné vytvořit instanci NlpEngineProvider, která slouží jako mezivrstva mezi spaCy NLP frameworkem a frameworkem Presidio. Stejně tak je potřeba i vytvořit instanci AnalyzerEngine s parametrem registry, který značí kolekci všech vytvořených instancí vybraných recognizerů (Zdrojový kód 8.2.7).

```
analyzer = AnalyzerEngine(
    nlp_engine=nlp_engine_custom,
    supported_languages=["cs"],
    registry=recognizer_registry
)
```
Zdrojový kód 8.2.7: Konstrukce objektu AnalyzerEngine

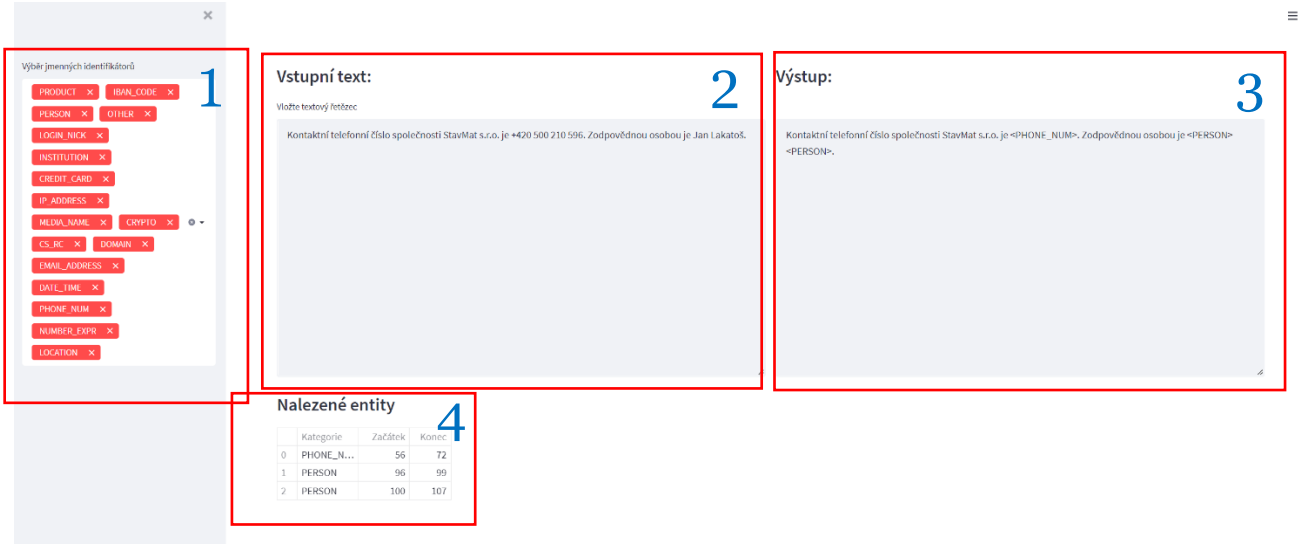
Nakonec jsou definovány zobrazované grafické prvky, jako je postranní panel (sidebar) nebo textová pole, která slouží pro zápis vstupního analyzovaného nestrukturovaného textu. Vytváření grafických komponent je ve frameworku Streamlit velmi přímočaré a intuitivní (Zdrojový kód 8.2.8).

```
col1.subheader("Vstupní text:")
st_text = col1.text_area(
    label="Vložte textový řetězec",
    value="Kontaktní telefonní číslo "
    "společnosti StavMat s.r.o. je +420 500 210 596. Zodpovědnou osobou je Jan Lakatoš.",
    height=400,
)
```
Zdrojový kód 8.2.8: Definice vstupního textového pole

V přiloženém hlavním spouštěcím skriptu *PIIAnonymizer.ipynb* je na výběr z 2 možných variant výsledného anonymizačního nástroje. První z variant je určena pro spuštění na virtuálním stroji bez dostupné GPU. Tato varianta používá model CPU_fine_nomorph, a to především kvůli rychlosti tohoto modelu. Druhá varianta, vhodná pro spuštění pouze na GPU využívá nejlepšího z vytrénovaných modelů GPU_bert_cased.

Na obrázku 11 lze vidět ukázku grafického rozhraní dokončeného anonymizačního nástroje běžícího v Google Colab jako serverová aplikace a zobrazeného ve webovém prohlížeči. Grafické rozhraní se skládá z následujících očíslovaných částí:

1. Panel pro výběr kategorií jmenných entit. V tomto panelu jde vypnout klasifikaci jmenné entity pro každou kategorii.
2. Vstupní textové pole, do kterého se zadává vstupní nestrukturovaný text, který je následně anonymizován.
3. Výstupní pole, ve kterém je zobrazen modifikovaný text vstupního textového pole. Jednotlivé detekované jmenné entity jsou ve výstupu nahrazeny jmény kategorií.
4. Tabulka, která zobrazuje všechny ve vstupním textu detekované jmenné entity, společně s informací, kde přesně byla v textu tato entita nalezena.



Obrázek 11: Ukázka grafického rozhraní anonymizačního nástroje

9 Evaluace anonymizačního nástroje

Cílem této kapitoly je vyhodnotit kvalitu vyvinutého anonymizačního nástroje. Aby došlo ke kvalitnímu zhodnocení, je prototyp evaluován klasifikací jmenných entit v typu dokumentu, pro který byl tento nástroj upraven, tedy na dokumentu typu smlouva.

9.1 Vytvoření doménově specifického evaluačního datasetu

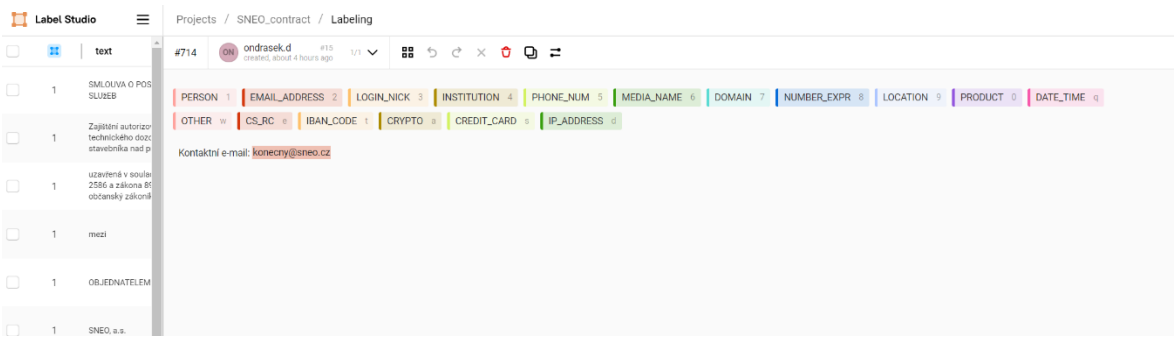
Vzhledem k tomu, že nebyl v době psaní této práce nalezen žádný vhodný dataset, který by obsahoval anotované dokumenty typu smlouva, bylo potřeba tento dataset vytvořit. Autorem vytvořený dataset **contract_eval** vznikl anotováním autorem vytvořené smlouvy, pojednávající o zajištění autorizovaného technického dozoru stavebníka nad prováděním staveb financovaných z veřejného rozpočtu. Tato smlouva vzhledem ke své podstatě podléhá povinnému nahrání do Veřejného registru smluv.

Anotace jmenných entit byla provedena pomocí nástroje Label Studio[41]. Label Studio je open source nástroj, kterým se dají připravovat anotace vstupních dat, které mohou být vloženy v běžném textovém formátu .txt. Je možné zde vytvořit komplexnější projekt, na kterém může vzdáleně pracovat i větší skupina anotátorů, přičemž nástroj Label Studio běží na vzdáleném serveru (který v případě této práce běžel lokálně na stroji autora).

V případě vytváření datasetu contract_eval byla anotace provedena nezávisle dvěma různými osobami. Sporné případy byly následně diskutovány a po souhlasu obou anotátorů byla zvolena vhodnější varianta.

Na obrázku 12 je vyobrazena ukázka práce s nástrojem Label Studio. V horní části hlavního pracovního okna jsou vidět jednotlivé předem pro tento projekt definované kategorie jmenných entit, které jsou následně pomocí označování textu přiřazovány k jednotlivým tokenům.

[41] https://labelstud.io/



Obrázek 12: Grafická ukázka nástroje Label Studio

Po dokončení anotace datasetu byl následně dataset exportován ve formátu CONLL2003. Tento výsledný dataset lze najít v příloze C.

9.2 Automatizovaná evaluace

Pro automatizovanou evaluaci vytvořeného anonymizačního nástroje bylo využito balíčku **presidio-research**[42], který je součástí vývojářských nástrojů SDK Presidio (Zdrojový kód 9.2.1).

```
%cd /content
!git clone https://github.com/microsoft/presidio-research.git
%cd /content/presidio-research/

!pip install -r requirements.txt
!python setup.py install
```

Zdrojový kód 9.2.1: Instalace balíčku presidio-research

Pro úspěšnou evaluaci pomocí presidio-research je potřeba programaticky převést evaluační dataset do kolekce objektů InputSample, přičemž každý z těchto objektů představuje jeden evaluovaný dokument. Nejprve bylo potřeba převést dataset contract_eval do binárního formátu .spacy a následně každý objekt Doc transformovat pomocí funkce *from_spacy_doc* (Zdrojový kód 9.2.2).

[42] https://github.com/microsoft/presidio-research

```
!mkdir '/content/drive/MyDrive/PIIAnonymizer/datasets/eval'
!python -m spacy convert -
n 10 '/content/DP/eval/contract_eval.conll' '/content/drive/MyDrive/PIIAnonymizer/datasets
/eval'

docbin = dataset_docs_from_path('/content/drive/MyDrive/PIIAnonymizer/datasets/eval/contra
ct_eval.spacy')

input_samples = []
for doc in docbin:
  input_samples.append(InputSample.from_spacy_doc(doc=doc))
```
Zdrojový kód 9.2.2: Převedení datasetu contract_eval do kolekce objektů InputSample

Samotná evaluace je provedena třídou Evaluator, která přijímá jako parametr vytvořený vlastní AnalyzerEngine, který je zodpovědný za celou klasifikaci jmenných entit. Objekt evaluator nejprve namapuje kategorie entit datasetu contract_eval na kategorie dostupné v anonymizačním nástroji, následně spustí klasifikaci nad celým datasetem a porovná výsledek s anotací (Zdrojový kód 9.2.3).

```
model = PresidioAnalyzerWrapper(analyzer_engine=analyzer, language=["cs"])

evaluator = Evaluator(model=model)
dataset = Evaluator.align_entity_types(
    deepcopy(input_samples), entities_mapping=presidio_entities_map
)

evaluation_results = evaluator.evaluate_all(dataset)
```
Zdrojový kód 9.2.3: Klasifikace jmenných entit v evaluačním datasetu spuštěná přes třídu Evaluator

Nakonec dojde k výpočtu finálních hodnot metrik precision, recall a F1 score, které jsou následně vypsány do konzole. Na obrázku 13 je vidět tento výsledek pro anonymizační nástroj, který v pozadí využívá modelu CPU_fine_nomorph a na obrázku 14 modelu GPU_bert_cased.

Entity	Precision	Recall	Number of samples
DATE_TIME	50.00%	44.44%	9
EMAIL_ADDRESS	66.67%	100.00%	2
IBAN_CODE	100.00%	100.00%	1
DOMAIN	nan%	0.00%	1
LOCATION	86.67%	66.67%	39
PERSON	66.67%	88.89%	9
INSTITUTION	nan%	0.00%	22
OTHER	10.77%	29.17%	24
PHONE_NUM	57.14%	80.00%	5
NUMBER_EXPR	86.84%	70.21%	94
PII	66.02%	66.02%	206

PII F measure: 66.02%

Obrázek 13: Výsledná evaluace anonymizačního nástroje používajícího model CPU_fine_nomorph

INSTITUTION	nan%	0.00%	22
LOCATION	94.44%	87.18%	39
DOMAIN	nan%	0.00%	1
IBAN_CODE	100.00%	100.00%	1
DATE_TIME	46.67%	77.78%	9
PHONE_NUM	100.00%	100.00%	5
EMAIL_ADDRESS	100.00%	100.00%	2
PERSON	75.00%	100.00%	9
OTHER	46.43%	54.17%	24
NUMBER_EXPR	92.31%	89.36%	94
PII	85.64%	81.07%	206

PII F measure: 81.67%

Obrázek 14: Výsledná evaluace anonymizačního nástroje používajícího model GPU_bert_cased

Výsledné F1 score, které bylo zvoleno jako hlavní kritérium pro hodnocení kvality nástroje bylo vypočteno pro nejlepší CPU model na 66,02 % a pro nejlepší GPU model 81,67 %.

9.3 Zhodnocení výsledků

Ač se zdá konečná kvalita nejlepšího modelu GPU_bert_cased s F1 score 81,67% jako dostatečná, je z výsledků klasifikace jednotlivých druhů jmenných entit vidět, že nástroj má v některých oblastech stále hodně prostoru pro zlepšení.

Konkrétně se jedná o kategorii DOMAIN, která vyjadřuje internetovou doménu a o kategorii INSTITUTION, která klasifikuje názvy různých institucí. U obou těchto kategoriích lze vidět, že Anonymizační nástroj nedokázal správně klasifikovat ani jednu ze jmenných entit tohoto typu v evaluačním datasetu. Tento problém by tak měl být v další iteraci vývoje tohoto nástroje rozhodně vyřešen.

Možným řešením je rozšíření trénovacího datasetu o větší počet těchto entit či například přidáním dalšího recognizeru, pracujícího na principu slovníkového vyhledávání českých firem dotazováním se API Obchodního rejstříku firem. Je také nutno poznamenat, že některé instituce byly klasifikovány pod kategorií OTHER.

Pro jmennou entitu, označující internetovou doménu by potom bylo vhodné opravit funkci předdefinovaného recognizer *UrlRecognizer*, který se zdá že svojí funkcí nevykonává správně.

Vytvořený prototyp anonymizačního nástroje splňuje stanovený cíl, týkající se jeho modularity. Je tak možné jednoduše programaticky upravovat stávající a vytvářet nové doménově specifické recognizery tak, aby mohlo dojít k jednoduché úpravě nástroje pro detekci doménově závislých osobních údajů. Zároveň je možné v případě potřeby dotrénovat spaCy model na dalších, doménově specifických datech, které dále zpřesní přesnost klasifikace.

Vyřešen byl i problém limitující závislosti na jazyce trénovacích datasetů, a to využitím předtrénovaného multilingvního transformer modelu, který tak umožnil dotrénovat lokalizovaný český NER model s vyšší přesností.

Závěr

Tato diplomová práce se zabývala návrhem a implementací modulárního anonymizačního nástroje, který umožňuje anonymizovat osobní údaje v nestrukturovaném textu při využití hlubokého učení a dalších relevantních technik.

Cílem přitom bylo navrhnout anonymizační nástroj tak, aby bylo možné ho jednoduše programaticky upravovat pro různé druhy osobních údajů. Důvodem pro tuto klíčovou požadovanou vlastnost nástroje byl nejednotný význam termínu „osobní údaj“. Osobním údajem totiž může být v závislosti na kontextu nestrukturovaného textu sada rozdílných doménově závislých jmenných entit. Zároveň bylo potřeba vyřešit problém s limitujícím počtem vhodných českých datasetů, anotovaných pro potřebu úlohy Named Entity Recognition.

Vyvíjený prototyp anonymizačního nástroje byl vzorově upraven pro detekci osobních údajů, běžně se vyskytujících v textových dokumentech podléhajících povinnému nahrávání do Veřejného registru smluv.

V kapitole 4 došlo k analýze osobních údajů v kontextu Veřejného registru smluv. Byl prozkoumán existující Nástroj pro anonymizaci dokumentů[43], dostupný na Portálu veřejné správy, typy dokumentů, které se běžně do Veřejného registru smluv nahrávají. Na základě toho byla vytvořena komplexní doménově specifická mapa kategorií osobních údajů, která byla následně logicky redukována. Tato mapa potom sloužila jako seznam doménově specifických osobních údajů, které musí být schopen vyvíjený prototyp anonymizovat.

Následně byla v kapitole 5 vypracována rešerše, zabývající se technikami hlubokého učení, vhodnými pro implementaci anonymizačního nástroje. Nejprve došlo k hrubému představení technik z oblasti hlubokého učení (Deep Learning), následované konkrétněji zaměřenými metodami z oblasti zpracování přirozeného jazyka (Natural Language Processing) a rozpoznávání jmenných entit (Named Entity Recognition). Byla představena typická architektura NER modelů a problematika rozpoznávání jmenných entit byla následně přenesena do českého jazyka. V rámci toho byly představeny a vysvětleny i dodatečné morfologické znaky slov, které je vhodné využít při trénování kvalitního NER modelu.

Došlo k představení možnosti využití state-of-the-art multilingvních modelů, včetně představení a popsání architektury několika konkrétních modelů.

V kapitole 6, zabývající se výběrem technologií, byla stanovena kritéria pro výběr vhodného NLP frameworku. Následně došlo na základě těchto kritérií k porovnání několika vhodných frameworků a k výběru frameworku použitého pro vývoj anonymizačního nástroje (spaCy).

---

[43] https://anonymizace.gov.cz/crossroad/

V další fázi, kterou popisuje kapitola 7, byla zkoumána množina datasetů, vhodných pro trénink vlastního NER modelu. Datasety byly porovnány a na základě definovaných kritérií byl vybrán dataset nejvhodnější pro další zpracování. Ten musel být následně transformován do požadované podoby. Také došlo k jeho detailnějšímu průzkumu a jeho dalším úpravám.

Kapitola 8 je rozdělená na 2 hlavní části. Nejdříve bylo pomocí frameworku spaCy vytrénováno několik variant NER modelů. 5 z těchto modelů je založených na architektuře transformer a potřebují ke svému správnému fungování využít síly grafické karty. Na 3 modelech, utilizujících jen CPU, byly porovnány výsledky klasifikace na coarse-grained datasetu, fine-grained datasetu bez dodatečných morfologických znaků a na fine-grained datasetu s morfologickými znaky. Všechny trénované modely byly následně evaluovány pomocí zvolené metriky F1 score, a na základě tohoto hodnocení byly vybrány 2 nejlepší modely: CPU_fine_nomorph, utilizující sílu CPU a GPU_bert_cased, utilizující i GPU.

Tyto 2 modely byly následně integrovány do samotného vyvíjeného prototypu anonymizačního nástroje. Ten ke své funkci využívá SDK Presidio a umožňuje kombinovat sílu NER modelů, založených na technice hlubokého učení a dalších relevantních klasifikačních technik. S využitím dříve vytvořené mapy kategorií osobních údajů byl přiřazen ke každém identifikovanému osobnímu údaji jeden recognizer, který se v anonymizačním nástroji stará o klasifikaci jmenné entity daného typu.

Dále bylo k anonymizačnímu nástroji vyvinuto s pomocí frameworku Streamlit grafické rozhraní, které umožňuje uživatelsky příjemným způsobem spustit anonymizační nástroj ve webovém prohlížeči a sledovat v reálném čase jeho funkčnost.

Anonymizační nástroj, upravený pro konkrétní doménově specifickou aplikaci anonymizace osobních údajů v dokumentech nahrávaných do Veřejného registru smluv, byl následně evaluován na autorem vytvořeném evaluačním datasetu contract_eval. Tento dataset obsahuje příklad anotované smlouvy, tedy reálný příklad dokumentu nahrávaného do Veřejného registru smluv.

Anonymizační nástroj dosáhl na evaluačním datasetu výsledného F1 score 81,67 %, při využití NER modelu GPU_bert_cased a 66,02 % při použití modelu CPU_fine_nomorph. F1 score 81,67 % je přitom dostatečně vysoké, aby se dal výsledek považovat za uspokojivý. Při analýze výsledků bylo nicméně zjištěno, že má nástroj problémy s klasifikací jmenných entit typu „instituce“ či „doména“.

V případných budoucích iteracích vývoje anonymizačního nástroje stále existuje řada možných zlepšení. Bylo by například vhodné se více zaměřit na doimplementování dodatečných recognizerů či na dotrénování dalších NER modelů. Zároveň by bylo vhodné doplnit mechanismus, který by dokázal na základě primárního či sekundárního typu osobního údaje vytvářet další závislosti, které by šlo dále znovu využít pro kvalitnější celkovou klasifikaci jmenných entit.

I přes výše zmíněné nedostatky se dá výsledný prototyp anonymizačního nástroje označit jako dostatečně kvalitní, aby naplňoval všechny cíle této práce. Nástroj je dostatečně modulární a jednoduše přizpůsobitelný, aby se dal dobře použít pro anonymizaci i dalších

doménově závislých osobních údajů a tím vyřešil absenci jednoznačné definice osobního údaje jako takového. Využitím multilingvního modelu byl také částečně vyřešen problém s limitující závislostí na jazyce trénovacích datasetů. Ač je zřejmé, že pro komerční nasazení tohoto anonymizačního nástroje do produkčního prostředí by ještě musel proběhnout další rozsáhlejší vývoj, dá se říct, že konceptuálně tato myšlenka o přizpůsobitelném modulárním anonymizačním nástroji dává smysl a je tak vhodná k dalšímu zkoumání.

Použitá literatura

AL OMRAN, Fouad Nasser A a Christoph TREUDE, 2017. Choosing an NLP Library for Analyzing Software Documentation: A Systematic Literature Review and a Series of Experiments. In: *2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)*: *2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)* [online]. Buenos Aires, Argentina: IEEE, s. 187–197 [vid. 2022-06-02]. ISBN 978-1-5386-1544-7. Dostupné z: doi:10.1109/MSR.2017.42

BENGIO, Yoshua a Olivier DELALLEAU, 2011. On the Expressive Power of Deep Architectures. In: Jyrki KIVINEN, Csaba SZEPESVÁRI, Esko UKKONEN a Thomas ZEUGMANN, ed. *Algorithmic Learning Theory* [online]. Berlin, Heidelberg: Springer, s. 18–36. Lecture Notes in Computer Science. ISBN 978-3-642-24412-4. Dostupné z: doi:10.1007/978-3-642-24412-4_3

BOUDA, Petr, 2019. *Aktuálně k registru smluv | Právní prostor* [online] [vid. 2022-02-13]. Dostupné z: https://www.pravniprostor.cz/clanky/spravni-pravo/aktualne-k-registru-smluv

CHAU, Ethan C., Lucy H. LIN a Noah A. SMITH, 2020. Parsing with Multilingual BERT, a Small Corpus, and a Small Treebank. *arXiv:2009.14124 [cs]* [online]. [vid. 2022-01-09]. Dostupné z: http://arxiv.org/abs/2009.14124

CLARK, Kevin, Minh-Thang LUONG, Quoc V. LE a Christopher D. MANNING, 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. *arXiv:2003.10555 [cs]* [online]. [vid. 2022-01-09]. Dostupné z: http://arxiv.org/abs/2003.10555

CONNEAU, Alexis, Guillaume LAMPLE, Ruty RINOTT, Adina WILLIAMS, Samuel R. BOWMAN, Holger SCHWENK a Veselin STOYANOV, 2018. XNLI: Evaluating Cross-lingual Sentence Representations. *arXiv:1809.05053 [cs]* [online]. [vid. 2022-05-06]. Dostupné z: http://arxiv.org/abs/1809.05053

CONNEAU, Alexis, Kartikay KHANDELWAL, Naman GOYAL, Vishrav CHAUDHARY, Guillaume WENZEK, Francisco GUZMÁN, Edouard GRAVE, Myle OTT, Luke ZETTLEMOYER a Veselin STOYANOV, 2020. Unsupervised Cross-lingual Representation Learning at Scale. *arXiv:1911.02116 [cs]* [online]. [vid. 2022-05-06]. Dostupné z: http://arxiv.org/abs/1911.02116

DEMIR, Hakan a Arzucan OZGUR, 2014. Improving Named Entity Recognition for Morphologically Rich Languages Using Word Embeddings. In: *Proceedings - 2014 13th International Conference on Machine Learning and Applications, ICMLA 2014* [online]. Dostupné z: doi:10.1109/ICMLA.2014.24

DEVLIN, Jacob, Ming-Wei CHANG, Kenton LEE a Kristina TOUTANOVA, 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]* [online]. [vid. 2022-01-09]. Dostupné z: http://arxiv.org/abs/1810.04805

ELDAN, Ronen a Ohad SHAMIR, 2015. The Power of Depth for Feedforward Neural Networks [online]. [vid. 2022-03-05]. Dostupné z: https://arxiv.org/abs/1512.03965v4

ELLMANN, Mathias, 2018. Natural language processing (NLP) applied on issue trackers. In: *Proceedings of the 4th ACM SIGSOFT International Workshop on NLP for Software Engineering* [online]. New York, NY, USA: Association for Computing Machinery, s. 38–41 [vid. 2022-01-09]. NL4SE 2018. ISBN 978-1-4503-6055-5. Dostupné z: doi:10.1145/3283812.3283825

GANESH, Prakhar, 2019. Distributed Vector Representation : Simplified. *Medium* [online] [vid. 2022-03-21]. Dostupné z: https://towardsdatascience.com/distributed-vector-representation-simplified-55bd2965333e

GEALFOW, John Altair a Christian MAY, 2019. Anonymizace osobních údajů v soudních rozhodnutích. *Revue pro právo a technologie* [online]. **10**(19), 3–39. ISSN 1805-2797. Dostupné z: doi:10.5817/RPT2019-1-1

GIACAGLIA, Giuliano, 2021. Transformers. *Medium* [online] [vid. 2022-03-21]. Dostupné z: https://towardsdatascience.com/transformers-141e32e69591

GOLDBERG, Yoav, 2017. Neural Network Methods for Natural Language Processing. *Synthesis Lectures on Human Language Technologies* [online]. **10**(1), 1–309. ISSN 1947-4040. Dostupné z: doi:10.2200/S00762ED1V01Y201703HLT037

GRISHMAN, Ralph a Beth SUNDHEIM, 1996. Message Understanding Conference-6: a brief history. In: *Proceedings of the 16th conference on Computational linguistics - Volume 1* [online]. USA: Association for Computational Linguistics, s. 466–471 [vid. 2022-01-09]. COLING '96. Dostupné z: doi:10.3115/992628.992709

HINTON, Geoffrey E., Simon OSINDERO a Yee-Whye TEH, 2006. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation* [online]. **18**(7), 1527–1554. ISSN 0899-7667, 1530-888X. Dostupné z: doi:10.1162/neco.2006.18.7.1527

HOTZ, Nick, 2018. What is CRISP DM? *Data Science Process Alliance* [online]. [vid. 2022-06-16]. Dostupné z: https://www.datascience-pm.com/crisp-dm-2/

CHAPMAN, P., J. CLINTON, R. KERBER, T. KHABAZA, T. REINARTZ, C. SHEARER a R. WIRTH, 2000. CRISP-DM 1.0: Step-by-step data mining guide. *undefined* [online]. [vid. 2022-06-16]. Dostupné z: https://www.semanticscholar.org/paper/CRISP-DM-1.0%3A-Step-by-step-data-mining-guide-Chapman-Clinton/54bad20bbc7938991bf34f86dde0babfbd2d5a72

JOHRI, Prashant, Sunil Kumar KHATRI, Ahmad AL-TAANI, Munish SABHARWAL, Shakhzod SUVANOV a Avneesh CHAUHAN, 2021. Natural Language Processing: History, Evolution, Application, and Future Work. In: [online]. s. 365–375. ISBN 9789811597114. Dostupné z: doi:10.1007/978-981-15-9712-1_31

KAMRAN, Muhammad a Syed MANSOOR, 2016. Named Entity Recognition System for Postpositional Languages: Urdu as a Case Study. *International Journal of Advanced Computer Science and Applications* [online]. **7**(10) [vid. 2022-05-07]. ISSN 21565570, 2158107X. Dostupné z: doi:10.14569/IJACSA.2016.071019

KOCIÁN, Matěj, Jakub NÁPLAVA, Daniel ŠTANCL a Vladimír KADLEC, 2021. *Siamese BERT-based Model for Web Search Relevance Ranking Evaluated on a New Czech Dataset* [online]. 3. prosinec 2021. B.m.: arXiv. [vid. 2022-06-05]. Dostupné z: http://arxiv.org/abs/2112.01810

KONKOL, Michal a Miloslav KONOPÍK, 2014. Named Entity Recognition for Highly Inflectional Languages: Effects of Various Lemmatization and Stemming Approaches. In: Petr SOJKA, Aleš HORÁK, Ivan KOPEČEK a Karel PALA, ed. *Text, Speech and Dialogue* [online]. Cham: Springer International Publishing, Lecture Notes in Computer Science, s. 267–274 [vid. 2022-05-06]. ISBN 978-3-319-10815-5. Dostupné z: doi:10.1007/978-3-319-10816-2_33

KRAUS, Martin, 2022. *Metodický návod k aplikaci ZRS*. 5. leden 2022. B.m.: MVČR.

KUBICA, Jan, 2019. *Vybrané problémy technologické realizace evropské ochrany osobních údajů* [online]. B.m. [vid. 2022-01-09]. Univerzita Karlova. Dostupné z: http://invenio.nusl.cz/record/397848

LAMPLE, Guillaume a Alexis CONNEAU, 2019. Cross-lingual Language Model Pretraining. *arXiv:1901.07291 [cs]* [online]. [vid. 2022-05-06]. Dostupné z: http://arxiv.org/abs/1901.07291

LI, Jing, Aixin SUN, Jianglei HAN a Chenliang LI, 2020. A Survey on Deep Learning for Named Entity Recognition. *arXiv:1812.09449 [cs]* [online]. [vid. 2022-03-20]. Dostupné z: http://arxiv.org/abs/1812.09449

LIM, Cheng, Ian TAN a Bhawani SELVARETNAM, 2019. Domain-General Versus Domain-Specific Named Entity Recognition: A Case Study Using TEXT. In: [online]. s. 238–246. ISBN 978-3-030-33708-7. Dostupné z: doi:10.1007/978-3-030-33709-4_21

LIU, Yinhan, Myle OTT, Naman GOYAL, Jingfei DU, Mandar JOSHI, Danqi CHEN, Omer LEVY, Mike LEWIS, Luke ZETTLEMOYER a Veselin STOYANOV, 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]* [online]. [vid. 2022-05-06]. Dostupné z: http://arxiv.org/abs/1907.11692

MAI, Khai, Thai-Hoang PHAM, Minh Trung NGUYEN, Tuan Duc NGUYEN, Danushka BOLLEGALA, Ryohei SASANO a Satoshi SEKINE, 2018. An Empirical Study on Fine-Grained Named Entity Recognition. In: *COLING 2018*: *Proceedings of the 27th International Conference on Computational Linguistics* [online]. Santa Fe, New Mexico, USA: Association for Computational Linguistics, s. 711–722 [vid. 2022-05-07]. Dostupné z: https://aclanthology.org/C18-1060

MENDELS, Omri, 2020. NLP approaches to data anonymization. *Medium* [online] [vid. 2022-01-09]. Dostupné z: https://towardsdatascience.com/nlp-approaches-to-data-anonymization-1fb5bde6b929

MIASCHI, Alessio a Felice DELL'ORLETTA, 2020. Contextual and Non-Contextual Word Embeddings: an in-depth Linguistic Investigation. In: *Proceedings of the 5th Workshop on Representation Learning for NLP* [online]. Online: Association for Computational Linguistics, s. 110–119 [vid. 2022-05-06]. Dostupné z: doi:10.18653/v1/2020.repl4nlp-1.15

MIHULKOVÁ, Jitka, 2018. *Co je, co není a co bude osobní údaj podle GDPR - Frank Bold Advokáti* [online] [vid. 2022-01-09]. Dostupné z: https://www.fbadvokati.cz/cs/clanky/541-co-je-co-neni-a-co-bude-osobni-udaj-podle-gdpr

MISHRA, Aditya, 2018. Metrics to Evaluate your Machine Learning Algorithm. *Medium* [online] [vid. 2022-01-09]. Dostupné z: https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

MOBERG, John, 2020. A deep dive into multilingual NLP models. *Peltarion* [online] [vid. 2022-01-09]. Dostupné z: https://peltarion.com/blog/data-science/a-deep-dive-into-multilingual-nlp-models

MVČR. *Zvláštní kategorie osobních údajů - Ochrana osobních údajů* [online] [vid. 2022-01-09]. Dostupné z: https://www.mvcr.cz/gdpr/clanek/zvlastni-kategorie-osobnich-udaju.aspx

NADEAU, David a Satoshi SEKINE, 2007. A survey of named entity recognition and classification. *Lingvisticæ Investigationes* [online]. **30**(1), 3–26. ISSN 0378-4169, 1569-9927. Dostupné z: doi:10.1075/li.30.1.03nad

NIGAM, vibhor, 2021. Natural Language Processing: From Basics, to using RNN and LSTM. *Analytics Vidhya* [online]. [vid. 2022-03-13]. Dostupné z: https://medium.com/analytics-vidhya/natural-language-processing-from-basics-to-using-rnn-and-lstm-ef6779e4ae66

PETASIS, Georgios, Frantz VICHOT, Francis WOLINSKI, Georgios PALIOURAS, Vangelis KARKALETSIS a Constantine D. SPYROPOULOS, 2001. Using machine learning to maintain rule-based named-entity recognition and classification systems. In: *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics* [online]. USA: Association for Computational Linguistics, s. 426–433 [vid. 2022-03-20]. ACL '01. Dostupné z: doi:10.3115/1073012.1073067

PINHASI, Assaf, 2021. Towards a Development Methodology for Machine Learning — part I. *Medium* [online]. [vid. 2022-06-16]. Dostupné z: https://assaf-pinhasi.medium.com/towards-a-development-methodology-for-machine-learning-part-i-f1050a0bc607

PRAGUE DEPENDENCY TREEBANK., [b.r.]. *2.1. Lemma structure* [online] [vid. 2022-05-08]. Dostupné z: https://ufal.mff.cuni.cz/pdt2.0/doc/manuals/en/m-layer/html/ch02s01.html

QU, Chen, Weize KONG, Liu YANG, Mingyang ZHANG, Michael BENDERSKY a Marc NAJORK, 2021. Natural Language Understanding with Privacy-Preserving BERT. In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* [online]. New York, NY, USA: Association for Computing Machinery, s. 1488–1497 [vid. 2022-01-09]. ISBN 978-1-4503-8446-9. Dostupné z: http://doi.org/10.1145/3459637.3482281

RATINOV, Lev a Dan ROTH, 2009. Design Challenges and Misconceptions in Named Entity Recognition. In: *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)* [online]. Boulder, Colorado: Association for Computational Linguistics, s. 147–155 [vid. 2022-05-07]. Dostupné z: https://aclanthology.org/W09-1119

SETTLES, Burr, 2004. Biomedical Named Entity Recognition using Conditional Random Fields and Rich Feature Sets. In: *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA/BioNLP)* [online]. Geneva, Switzerland: COLING, s. 107–110 [vid. 2022-03-20]. Dostupné z: https://aclanthology.org/W04-1221

SIDO, Jakub, Ondřej PRAŽÁK, Pavel PŘIBÁŇ, Jan PAŠEK, Michal SEJÁK a Miloslav KONOPÍK, 2021. *Czert -- Czech BERT-like Model for Language Representation* [online]. arXiv:2103.13031. B.m.: arXiv [vid. 2022-06-05]. Dostupné z: doi:10.48550/arXiv.2103.13031

SILVA, Paulo, Carolina GONÇALVES, Carolina GODINHO, Nuno ANTUNES a Marilia CURADO, 2020. Using natural language processing to detect privacy violations in online contracts. In: *Proceedings of the 35th Annual ACM Symposium on Applied Computing* [online]. New York, NY, USA: Association for Computing Machinery, s. 1305–1307 [vid. 2022-01-09]. ISBN 978-1-4503-6866-7. Dostupné z: http://doi.org/10.1145/3341105.3375774

SPACY GITHUB,HONNIBAL, Matthew, Ines MONTANI, Sofie VAN LANDEGHEM a Adriane BOYD, 2020. *spaCy: Industrial-strength Natural Language Processing in Python* [online]. Python [vid. 2022-05-08]. Dostupné z: doi:10.5281/zenodo.1212303

STRAKA, Milan, Jakub NÁPLAVA, Jana STRAKOVÁ a David SAMUEL, 2021. RobeCzech Base [online]. [vid. 2022-06-05]. Dostupné z: https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-3691

STRAKOVÁ, Jana, Milan STRAKA a Jan HAJIČ, 2016. Neural Networks for Featureless Named Entity Recognition in Czech. In: Petr SOJKA, Aleš HORÁK, Ivan KOPEČEK a Karel PALA, ed. *Text, Speech, and Dialogue* [online]. Cham: Springer International Publishing, Lecture Notes in Computer Science, s. 173–181 [vid. 2022-05-08]. ISBN 978-3-319-45509-9. Dostupné z: doi:10.1007/978-3-319-45510-5_20

ŠEVČÍKOVÁ, Magda, Zdeněk ŽABOKRTSKÝ a Oldřich KRŮZA, 2007. Named Entities in Czech: Annotating Data and Developing NE Tagger. In: Václav MATOUŠEK a Pavel MAUTNER, ed. *Text, Speech and Dialogue* [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, Lecture Notes in Computer Science, s. 188–195 [vid. 2022-05-07]. ISBN 978-3-540-74627-0. Dostupné z: doi:10.1007/978-3-540-74628-7_26

TSAI, Chen-Tse, Stephen MAYHEW a Dan ROTH, 2016. Cross-Lingual Named Entity Recognition via Wikification. In: *CoNLL 2016*: *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning* [online]. Berlin, Germany: Association for Computational Linguistics, s. 219–228 [vid. 2022-05-06]. Dostupné z: doi:10.18653/v1/K16-1022

ÚFAL., [b.r.]. *Czech Named Entity Corpus 2.0 | ÚFAL* [online] [vid. 2022-05-07]. Dostupné z: https://ufal.mff.cuni.cz/cnec/cnec2.0

ÚOOÚ, 2016. *Ochrana osobních údajů v souvislosti se zákonem o registru smluv: Názory a rozhodnutí Úřadu: Úřad pro ochranu osobních údajů* [online] [vid. 2022-02-14]. Dostupné z: https://www.uoou.cz/ochrana-osobnich-udaju-v-souvislosti-se-zakonem-o-registru-smluv/d-20322/p1=1099

VYCHEGZHANIN, Sergey a Evgeny KOTELNIKOV, 2019. Comparison of Named Entity Recognition Tools Applied to News Articles. In: [online]. s. 72–77. Dostupné z: doi:10.1109/ISPRAS47671.2019.00017

WANG, Haohan a Bhiksha RAJ, 2017. On the Origin of Deep Learning. *arXiv:1702.07800 [cs, stat]* [online]. [vid. 2022-03-03]. Dostupné z: http://arxiv.org/abs/1702.07800

WILLIAMS, Jake a Giovanni SANTIA, 2017. Context-Sensitive Recognition for Emerging and Rare Entities. In: *Proceedings of the 3rd Workshop on Noisy User-generated Text* [online]. Copenhagen, Denmark: Association for Computational Linguistics, s. 172–176 [vid. 2022-01-09]. Dostupné z: doi:10.18653/v1/W17-4423

WOODS, W. A., 1970. Transition network grammars for natural language analysis. *Communications of the ACM* [online]. **13**(10), 591–606. ISSN 0001-0782. Dostupné z: doi:10.1145/355598.362773

XU, Hanchen, Zhenxiang CHEN, Shanshan WANG a Xiaoqing JIANG, 2021. Chinese NER Using ALBERT and Multi-word Information. In: *ACM Turing Award Celebration Conference - China (ACM TURC 2021)* [online]. New York, NY, USA: Association for Computing Machinery, s. 141–145 [vid. 2022-01-09]. ISBN 978-1-4503-8567-1. Dostupné z: http://doi.org/10.1145/3472634.3472667

YADAV, Vikas a Steven BETHARD, 2019. A Survey on Recent Advances in Named Entity Recognition from Deep Learning models. *arXiv:1910.11470 [cs]* [online]. [vid. 2022-05-06]. Dostupné z: http://arxiv.org/abs/1910.11470

Přílohy

Příloha A: Kompletní mapa kategorií osobních údajů

Tato příloha, ve formátu sešitu Excel, obsahuje kompletní a zjednodušený seznam identifikovaných jmenných entit, běžně se vyskytujících v dokumentech běžně nahrávaných do Veřejného registru smluv. První list sešitu obsahuje kompletní seznam, druhý list potom seznam zjednodušený.

Příloha B: Zdrojový kód Anonymizačního nástroje

Příloha B obsahuje v archivu ZIP všechny zdrojové kódy použité při implementaci Anonymizačního nástroje.

Hlavním spustitelným souborem ve formátu Jupyter Notebook je PIIAnonymizer.ipynb. Tento skript, který je určený pro spuštění ve vývojovém prostředí Google Colab, obsahuje kód potřebný pro načtení a zpracování datasetu, vytrénování NER modelu a jeho evaluaci, evaluaci kompletního Anonymizačního nástroje a kód pro načtení a spuštění webové GUI aplikace založené na frameworku Streamlit.

Tento skript dále využívá připojeného vzdáleného úložiště Google Disk, do kterého se při tréninku modelů ukládají váhy trénovaných modelů a kde se postupně vytváří celá struktura potřebná pro spuštění nástroje.

Složka spacy_config_files obsahuje konfigurační soubory potřebné pro trénink jednotlivých modelů. V těchto souborech je definována architektura těchto modelů.

Složka streamlit_app obsahuje zdrojové kódy pro grafickou webovou aplikaci, která umožňuje spustit samotný Anonymizační nástroj v grafické podobě. Jsou zde obsaženy varianty pro spuštění aplikace v prostředí utilizujícím CPU a GPU.

Zdrojové kódy jsou také dostupné v autorově GitLab repositáři DP[44].

Příloha C: Evaluační dataset contract_eval

Příloha C obsahuje autorem vytvořený evaluační dataset contract_eval ve formátu CONLL2003.

[44] https://github.com/ondrasekd/DP