Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informatiky a kvantitativních metod

Zpracování nestrukturovaných dat pomocí IBM Watson

Diplomová práce

Autor: Bc. Radek Salay
Studijní obor: AI2

Vedoucí práce: Ing. Barbora Tesařová Ph.D.

Hradec Králové Listopad 2020

Prohlášení:

Prohlašuji, že jsem diplomovou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 16.11.2020 Bc. Radek Salay

Poděkování:

Tímto bych chtěl poděkovat Ing. Barboře Tesařové, Ph.D. za odborné vedení, poskytnuté informace a cenné rady při zpracovávání mé diplomové práce.

Anotace

Tato diplomová práce je zaměřena na zpracování nestrukturovaného textu pomocí aplikace IBM Watson a využití získaných informací pro vyhledávání v prostředí knihovních archívů. Čtenář je seznámen s problematikou BigDat. Práce nastiňuje teoretické základy pro strojové zpracování textu. Následně se soustředí na aplikaci IBM Watson a představení jejího využití při zpracování textu. Též je představen proces tvorby specifických modelů, pro IBM Watson, využívaných pro zpracování textu. Závěrem je uvedena aplikace Knihovna 2.0, která demonstruje využití zpracovávání textu pro Knihovní systémy.

Annotation

Title: Processing of unstructured data with IBM Watson

This thesis is focused on processing unstructured data with application IBM Watson and using gathered information for search purpose in the environment of library archive. Reader is acquainted with problematics of BigDat. Thesis is outlining theoretical basics for machine processing of text. Then is focusing on application IBM Watson and introducing its use when processing text. There is also introduced a process of creation of specific models for IBM Watson, used for processing text. In conclusion, there is introduced application Knihovna 2.0, which demonstrates use of processing text for library systems.

Obsah

1 Úvod

Informace nám mohou přinést velmi mnoho nebo také velmi málo. Na to, aby však přinesly vůbec něco, je nejprve musíme získat. Za poslední desetiletí se velmi rozšířily možnosti, jak tvořit data. Rozšířila se jak místa, kde se dají data tvořit, ale rozšířily se i techniky pro rychlejší a efektivnější tvorbu dat. Technický rozmach v posledních letech lidem dovolil generovat a skladovat až nepředstavitelné množství dat. Jedná se o takové množství, že lidé samotní již nejsou plně schopni všechna tato data zpracovávat a tvořit z nich informace. Lidé ke své smůle sice často informace z dat získají, ale většinou až v době, kdy má již informace menší či téměř žádnou cenu. Proto je nesmírně důležité uvědomit si, že v okamžiku, kdy jsou používány technologické nástroje pro tvorbu dat, je zároveň ruku v ruce potřeba použít technologických nástrojů i na jejich následné zpracovávání. Bez těchto nástrojů se lidé připravují o možnost využít potenciálu, který se v datech skrývá.

Největším producentem dat, která jsou pro lidi zajímavá, popřípadě cenná, jsou lidé samotní. Lidé však data nejčastěji neukládají v některé lehce strojem zpracovatelné podobě. Je tomu právě naopak. Lidé nejvíce ukládají data v podobě textu psaného některým z jazyků. Naštěstí má každý jazyk svá pravidla a díky těmto pravidlům lze pro daný jazyk vytvořit algoritmus, pomocí kterého i stroj může daný jazyk číst.

Technickým nástrojem pro zpracovávání dat a následnou tvorbu informací bude nadále v této diplomové práci myšlen algoritmus.

Algoritmy odvádí tu těžší práci, kterou je myšleno zpracování dat určené jejich významu. Tudíž tvorba informací z dat. Proces zpracování dat tím však není ukončen, ale ani tímto nezačíná. Zpracování lidské řeči, nebo jakýchkoli jiných nestrukturovaných dat, je daleko rozsáhlejší proces. Pro usnadnění tohoto procesu bude v diplomové práci vybrán produkt IBM Watson (1). Prostředí IBM Watson bude v diplomové práci použito pro ukázku, jak se dají aplikovat konkrétní algoritmy na zpracování nestrukturovaného textu v praxi, kdy IBM Watson získá

data z připraveného zdroje a následně data analyzuje algoritmy. IBM Watson v poslední fázi zobrazí získané informace pro další manipulaci. Touto manipulací je myšleno filtrování či kombinace různých kritérií, potřebných k efektivnímu hledání v datech.

2 Cíl práce

Využití IBM Watson pro zpracování nestrukturovaného textu v aplikaci Knihovna 2.0.

Diplomová práce seznámí čtenáře s problematikou zpracování nestrukturovaného textu. Dále práce představí produkt Watson a jeho pod produkty pro zpracovávání anglického jazyka. V závěru práce bude ukázané praktické využití těchto poznatků v knihovní aplikaci.

3 BigData

BigData je pojem, který se dnes dá interpretovat vícero způsoby. V této diplomové práci bude pojem BigData chápán definicí od společnosti Gartner.

„BigData jsou informačními aktivy o velkém objemu, velké proměnlivosti a/nebo velké různorodosti, která vyžadují nákladově efektivní inovativní formy zpracování k usnadňování rozhodovacího procesu, objevnému vhledu (do problematiky) a procesní optimalizaci.“ (2).

Obecně se ale jedná o problematiku „tří V“ (3).

Problematika objemu (Volume): U BigDat se, jak už sám název napovídá, potýkáme s něčím velkým, proto „big“. Tím je myšleno opravdu velké množství dat, řádově se dostáváme do terabytů (10^{12}) až petabytů (10^{15}). Tyto číslovky je však nutno brát s jistou rezervou. Jelikož pořád do jisté míry platí Morův zákon[1]. Brzy i tyto jednotky budou jen malá zrnka v poušti dat.
Takovéto množství dat často není na jednom místě, myšleno v jednom datacentru. Možnost mít všechna svá data na jednom místě, nejlépe ve firemním sklepě, již není plně možné. Taková varianta však již není ani chtěná. A to jak z bezpečnostních, tak i finančních důvodů. Proto jsou často data rozdistribuována na vícero serverech, které se nachází po celé zemi. Ani vlastník dat již neví, kde se jeho data fyzicky nacházejí. Pokud však chceme z dat získávat informace, je především potřeba mít k nim přístup, z čehož mohou vznikat jistá omezení, způsobená vlivem rychlosti připojení, či při nutných servisních odstávkách serverů. Pro dnešní technologie takové množství dat není velkým problémem, naopak, jsou na to stavěny a zvládly by určitě i více. Problém však je, že množství dat, které máme dnes, se uchovává na technologiích (pevné disky, databázové struktury), které na takové množství dat nebyly připravovány.

[1] Pravidlo o exponenciálním růstu výkonu tranzistorů.

Problematika rychlosti (Velocity): Parametr rychlosti udává, jak rychle, nebo spíše v jakém typu, jsou data do systému přijímána/vkládána.

- Batch: Data se do systému dostávají v určitých dávkách. Tento přísun dat do systému si můžeme představit na cestovních náhradách. Na konci měsíce zaměstnanci předávají své cestovní příkazy, aby za ně dostali náhradu. Následně jsou všechny cestovní příkazy najednou zpracovány účetním oddělením. Tudíž nám data o výdajích na cesty zaměstnanců u firem přibývají v systému vždy jednou měsíčně. Dávky nemusí být pravidelné a jejich velikost se může lišit v závislosti, čeho se data týkají. U cestovných příkazů bude jiná velikost dávky v únoru, který má 28-29 dní a v lednu, který má 31 dní.
- Near-time: Skoro v reálném čase. Tento pojem je velmi subjektivní. Cílem je vždy data zpracovávat co nejrychleji, tudíž v reálném čase. Avšak docílit toho není vždy jednoduché, občas ani možné. Proto se často spokojíme s tím, že něco zpracováváme dostatečně rychle, tudíž skoro v reálném čase. Jako příklad lze uvést sondu, která letí od země na Mars. Nejdříve data dostáváme bez zpoždění, tedy v reálném čase. Sonda se ale stále vzdaluje. Až najednou máme v datech zpoždění půl vteřiny. Zpoždění půl vteřiny si běžný člověk ani nevšimne a připadá mu, že je vše pořád v reálném čase. Systém už toto zpoždění však zaznamená a může ho to i výrazně zpomalit. S postupující vzdáleností, jak sonda letí, se znovu zpoždění zvětší, a to třeba již na pět minut. To následně může být doba, kterou někteří nedočkaví lidé mohou označit za dávku (Batch). Proto záleží na daném systému, co je pro něj dostatečně rychle, aby jeho uživatelé nebyli zpomalováni.
- Streaming: Streamové nebo real time přibývání dat do systému. Data se do systému dostávají v reálném čase. Jakmile jsou data vytvořena, tak je systém okamžitě bez prodlení získá a začíná zpracovávat. Takovýto průchod dat je u systému, kde uživatelé potřebují mít neustálé informace o daném stavu. Příkladem mohou být systémy zpracovávající počasí, popřípadě finanční trhy. Jedná se o systémy, kdy se každou vteřinu může něco stát či změnit.

Čím rychleji se data dostávají do systému, tím dříve nad nimi může být provedena analýza a jejich transformace na informace.

Problematika různorodosti (Variety):

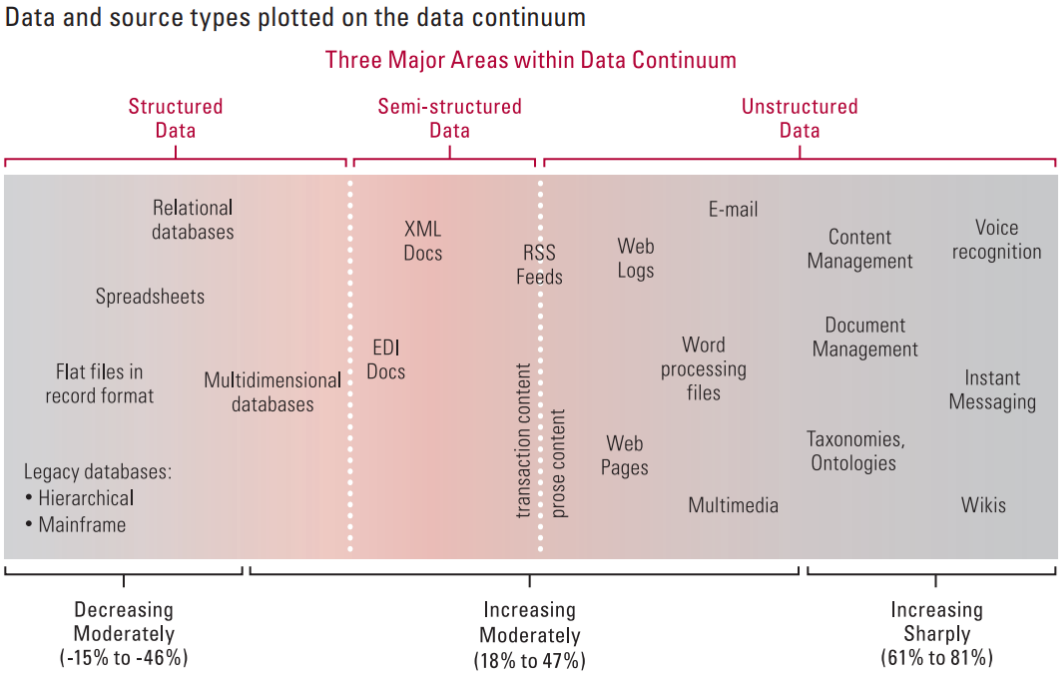
Data nejsou jen věty v dokumentech a číslice v tabulkách, ale jsou to i videa a zvukové záznamy. Informace můžeme získávat ze všech datových formátů. Ať již ze strukturovaných jako jsou tabulky, tak i z nestrukturovaných, jakými jsou dokumenty či zvukové záznamy. (4)

- **Strukturovaná data:** Data, která dokáže zpracovávat relační databázový systém. Tato data se nachází v tabulkách. V tabulkách jsou buňky obsahující samotná data, která jsou vždy daného datového typu. Daný typ je určen sloupcem, ve kterém se buňka nachází. Těmito typy zpravidla bývají písmena nebo číslice.
- **Semi-strukturovaná:** Tato data sice mají svou strukturu a jsou strojově zpracovatelná, ale nejsou zpracovatelná relačními databázovými systémy. Jedná se například o formátovaný text v textových dokumentech, emaily, XML, nebo JSON soubory.
- **Nestrukturovaná:** Nestrukturovaná data tvoří největší část objemu BigDat. Tato kategorie se dále dělí:
    - Textová: Dokumenty psané lidmi, volný text, příspěvky na sociálních sítích.
    - Ne-textová: Obrázky, zvukové záznamy, videa.

  Obě dvě kategorie se vyznačují tím, že data nejsou v předem dané struktuře.

- **Kombinovaná:** Předem zmíněné typy dat se mohou libovolně kombinovat a mezi sebou přecházet.

Procentuální zastoupení typů dat je nastíněno v obrázku níže. Obrázek 1: Rozdělení dat



Obrázek 1: Rozdělení dat (5)

Toto jsou základní vlastnosti BigDat. Mohou však být přidané i další vlastnosti a parametry, které tento model o třech proměnných obohacují. Vzhledem k tomu, že je tato diplomová práce věnována především nestrukturovaným datům, hlavně pak datům tvořenými lidmi ve formě lidské řeči, bude přínosné přidat ještě další parametry definované SAS[2] (6).

Variabilita (Variability): Tato vlastnost souvisí úzce s rychlostí (Velocity) a s různorodostí (Variety). Data se netvoří konstantně ve stejném množství. V noci například bude menší tvorba zpráv na sociálních sítích než přes den, kdy je většina lidí vzhůru.

Popřípadě pokud sledujeme jistý jev v datech. Můžeme například sledovat, jak moc lidé mluví o fotbalu. Jaká je běžná četnost hovorů na téma fotbal a jak

[2] Analytics Software & Solutions

se četnost zvýší například v době pořádání mistrovství světa, kdy o fotbalu najednou hovoří i lidé, které fotbal za normálních okolností vůbec nezajímá. Najednou se však zapojují do hovorů i oni jen proto, že je to v danou dobu trendem. Takovéto trendy mohou silně ovlivnit zpracovávání daných dat a zatížit dané systémy.

Složitost (Complexity): Data mohou pocházet z více zdrojů. Komunikace mezi dvěma účastníky nemusí, a také se málo kdy odehrává, jen pouze pomocí jednoho media. Účastníci mohou svou komunikaci zahájit pomocí emailu. Následně se však mohou dorozumívat telefonicky a ukončit svojí komunikaci opět pomocí emailu. Pokud má však systém správně analyzovat takovouto komunikaci, potřebuje data z obou dvou zdrojů. Jak emailovou korespondenci, tak záznam telefonního hovoru. Pokud by systém nebyl schopen jednu z těchto forem komunikace zpracovávat, tak jeho výsledky mohou být více či méně zkreslené, či dokonce chybné. Lidé používají čím dál více druhů dorozumívajících prostředků, od emailů, chatu, v kterých se často dorozumívají i obrázky, telefonních hovorů, fyzických dokumentů. Systémy zpracovávající BigData potřebují se všemi těmito formáty umět pracovat a být připravené připojovat další. Těmito dalšími formáty v budoucnu budou bezpochyby i data od neživých subjektů. Těmito subjekty budou různé senzory snímající například počasí, provoz na silnicích, či srdeční tep pacientů. Nesmíme však ani zapomínat na rostoucí vliv umělé inteligence v podobě chat-botů, kteří se čím dál tím více zapojují do komunikace mezi lidmi a jsou tedy také zdrojem dat. Přesto však data těchto chat-botů nemusí být pro zkoumání nijak zajímavá a mohou naopak analýze uškodit.

Dále je potřeba také přidat ještě jedno "V".

Důvěryhodnost (Veracity): jedná se o velmi důležitý parametr s ohledem na to, že mnoho dat je tvořeno lidmi a jak už to bývá, lidé jsou občas nespolehliví. Takže i data, která tito lidé vytvoří, mohou být nespolehlivá. Tudíž je potřeba analyzovaným datům přidat určitou důvěryhodnost. Pokud je tedy zpracovávána analýza zpráv o určité události, například o automobilové nehodě, je třeba dát jinou váhu názoru člověka, který byl stovky kilometrů od nehody a jen o události píše

komentář na sociální síti a jinou váhu by měla dostat oficiální zpráva policisty, který nehodu vyšetřoval. Dále je také potřeba počítat i s obyčejnými, avšak velmi častými chybami, jako je chybně napsané telefonní číslo, nadiktování neplatného trvalého bydliště a podobně. A v neposlední řadě nesmí být opomíjeno ani to, že technologické nástroje, kterými byla tato data vytvořena, mohou obsahovat chyby (bugs), které data ovlivňují.

3.1 Využití

Samotná podstata BigDat, představená v předchozí kapitole, vymezuje odvětví, ve kterých je možné se s touto problematikou potkat. Všude tam, kde se shromažďují za nějakým účelem data, se ve větší či menší míře můžeme potkat s problematikou BigDat. Nejčastěji se však setkáváme s BigDaty u větších firem a institucí. Není pravdou, že by menší subjekty data neschraňovali, ale často nemají potřebu, nebo prostředky pro jejich analýzu. Analýza dat v tak ohromném objemu a často zároveň velké složitosti, zvlášť pokud se bavíme o semi-strukturovaných či nestrukturovaných datech, je v dnešní době stále drahou a náročnou disciplínou, navíc s nejistým výsledkem. Zároveň je potřeba si uvědomit, že využití BigDat je stále relativně na počátku své existence, tudíž není samospasné. Analýza BigDat zatím přináší spíše přidanou hodnotu, či konkurenční výhodu, přičemž jako každý jiný nástroj je i analýza BigDat lehce ovlivněná lidským faktorem.

O tom, jak mohou nekompletní data, nebo data, která obsahují chyby, či nedávají smysl, vést ve výsledku ke špatným rozhodnutím, která mohou silně poškodit buď jednotlivce, nebo i ohrozit konkurenceschopnost firem, které na základě takto chybných závěrů vycházejících z chybných dat přizpůsobí svou firemní strategii, hovořil ve své analýze (7) i Gary King z Harvardova ústavu pro kvantitativní společenské vědy.

Ve své analýze popsal projekt s využitím velkého objemu dat. Projekt měl předpovídat míru nezaměstnanosti v USA. Zdrojem dat měly být tweety a příspěvky z ostatních sociálních sítí, ve kterých se vyskytovala klíčová slova "zaměstnání", "nezaměstnanost" a "inzerát".

Při monitoringu tehdy narazili na vysoký počet tweetů, které sice obsahovalo klíčové slovo "jobs", v anglickém jazyce výraz pro zaměstnání, ale jak později zjistili, nemělo nic společného s nezaměstnaností.

„Jednoduše jsme si nevšimli toho, že zemřel Steve Jobs,“ řekl King. *„Podobným problémům se můžete pokusit vyhnout přidáním výjimek, nikdy si však nemůžete být jistí,“* dodal King s tím, že relevanci určitých pojmů si může vyzkoušet každý sám. *„Stačí je zadat do webového vyhledávače. Zobrazí se jak relevantní výsledky, tak ty očividně o něčem úplně jiném.* (7)*“*.

Na problém nerelevantnosti některých dat upozorňoval i Kim Jonese (7), viceprezident společnosti Vantiv. Hovořil o tom, že si někdy lidé mylně myslí, že čím více mají dat, tím přesnější výsledky z nich vyplynou.

Wall Street Journal již v únoru 2013 informoval o tom, že pojišťovny v USA využívají velké množství dat, aby jejich pomocí vytvořili profily svých pojištěnců. Jedním ze zkoumaných parametrů byl například i počet kusů oblečení větších velikostí, dle kterého pojišťovny chtěly identifikovat obezitu. Výzkum však nepočítal s variantou, že oblečení nadměrných velikostí daný pojištěnec nemusí kupovat přímo pro sebe, ale může ho kupovat pro jiného člena rodiny.

Kim Jones v souvislosti nadměrného sběru dat dodává, že tendence vyvozovat chybné závěry na základě špatné analýzy dat, je ve světě čím dál častější a postihuje čím dál více lidí.

Dle Jonese:"*Je vždy špatné stavět svá rozhodnutí pouze na analýze dat. Pokud neberete v úvahu lidský element, musíte nutně dospět k více chybovým závěrům.* (7).

Analyzování BigDat může mít přínos ve všech oborech lidské činnosti.

- **Státní správa:** Největšími zpracovateli dat jsou státní orgány. Prostřednictvím formulářů požadují po občanech, podnikatelích a firmách ohromné množství informací, které pak musí následně každý rok přečíst a zpracovat. Zároveň je však na státní správu, mimo požadované formuláře, směrováno i velké množství dotazů a žádostí. Každá žádost je přečtena a je na ní odpovězeno (8). Toto vše být mohlo být částečně zautomatizováno.

Algoritmy na zpracování lidské řeči dokážou během sekund přečíst desítky stránek formátu A4 a vytvořit z nich výtah na několik řádků, který vystihne danou problematiku.

- **Marketing:** Marketingové společnosti spolu s reklamními společnostmi využívají BigData ve velké míře při utváření profilů zákazníků. To vše dokážou díky relativně malým střípkům informací, které ve velké míře lidé po sobě zanechávají na internetu. Například informace o navštívených stránkách, provedených nákupech či z jiných aktivit. Díky těmto informacím je možné sestavit model dané osobnosti a cíleně soustředit pouze relevantní reklamu.

- **Zdravotnictví:** Zdravotní záznamy představují obrovské množství uskladněných informací nejen o jednotlivých pacientech, ale i o společnosti či určitých skupinách. Zpracováním těchto dat je možné vyhledávat trendy nemocí či skryté účinky, či úskalí léků.

- **Bezpečnost:** Získávání skrytých informací z na první pohled nevyužitelných dat je cílem mnoha bezpečnostních složek. Ať se již jedná o státní či soukromé subjekty. Možnost analyzovat tisíce konverzací ať již písemných či v mluvené podobě, by mohlo vést k velkému navýšení bezpečnosti.

4 Historie zpracování textu

Historicky se zpracováním textu a přístupu k jazyku jako k určitému systému zabýval švýcarský profesor Ferdinand de Saussure. Dílo "Cours de Linguistique Générale" (9) (Kurz obecné lingvistiky), které bylo vydáno (1916) po jeho smrti (1913) a vychází z jeho přednášek na univerzitě v Ženevě. Dílo se zaobírá myšlenkami, že jazyk je určitý systém, kde každá jeho část má své místo, a tudíž je možné ho strukturovat.

V roce 1950, Alan Turing ve své seminární práci „Computing Machinery and Intelligence" (9) zmiňuje stroj, který je schopný komunikovat s lidmi a díky tomu může být nerozpoznatelný od člověka. Tato komunikace vyžaduje právě schopnost zpracovávat a analyzovat řeč, přesněji nějaký jazyk. Úkoly pro takový stroj by se daly kategorizovat do tří odvětví:

- rozpoznání jazyka
- porozumění přirozenému jazyku
- generování přirozeného jazyka.

V sedmdesátých letech dvacátého století vzniklo několik úspěšných projektů pracujících s lidskou řečí, například programovací jazyk LIPS[3]. Známým je též program ELIZA, používající reflexní techniky k imitování konverzace, který je založený na relativně jednoduchých gramatických pravidlech. ELIZA stejně jako většina tehdejších programů pracovala na principu předem napsaných pravidel pro práci s textem. Tento přístup se využívá i dnes pro analýzu a vyhledávání určitého textu v podobě REGEX výrazů (9) (10).

Dle Cvrčka *Regulární výrazy (pojem pochází z teorie formálních jazyků, jeho význam běžně používaný v informatice je však poněkud odlišný) umožňují exaktně popsat množinu textových řetězců odpovídající hledanému výrazu či jevu.* (11)

[3] Locator/Identifier Separation Protocol

Regulární výrazy byly představeny americkým matematikem Stephen Cole Kleene v roce 1951. (12) V původní verzi se jednalo o matematické notace (regular sets).

Následně však přišel útlum způsobený nemožností naplno využít potenciálu zpracování textu. Jelikož strojem zpracovávaný text byl stále dražší, než kdyby stejný text zpracovával člověk. Zájem o tuto doménu se opět vrátil až v letech osmdesátých, kdy pokrok ve výkonu počítačů stoupl natolik, že již byli levnější než lidé. Změny také nastaly v samotném přístupu k algoritmům na zpracování textu. Začaly se více upřednostňovat statistické modely a poznatky ze strojového učení. Tento trend pokračuje až do dnes. Dnes se strojové zpracování textu již využívá běžně. Dobrým příkladem mohou být asistenční služby Siri od společnosti Apple, Alexa od Amazonu či Google Assistant od Googlu.
(9) (10)

5 Zpracování textu

Zpracování textu je postupný proces skládající se z mnoha menších činností. V následující části budou teoreticky představeny některé elementární části. Tyto části jsou obecně používány pro jakoukoli strojovou analýzu textu.

5.1 Tokenization

Abychom mohli zpracovávat text, což je uskupení znaků v určitá slova, která mají význam jak sami o sobě, tak v návaznosti na jiná slova, je nutné nejdříve určit tato jednotlivá slova. Nemusí to být však jen slova. V textu se vyskytují i jiné znaky než jen písmena, ze kterých jsou slova složená. Musíme také například zpracovávat číslice či jiné matematické operace, které se v textu mohou vyskytnout.
Tudíž je nutné určit nejmenší možnou jednotku. Takovouto jednotkou u zpracovávání textu je „Token".
Zpracování textu do jednotlivých tokenů, je samostatný proces. Toto „rozkrájení" textu do jednotlivých, dále nedělitelných části, je prvním krokem při zpracování jakéhokoli textu. (13)

5.1.1 Token

V publikaci *Compilers, principles, techniques, and tools* můžeme nalézt definici tokenu: „Token je dále dělitelná veličina skládající se ze jména a dalších vlastností s vlastní hodnotou. Jméno tokenu je abstraktní symbolická reprezentace lexikálního typu. Jedná se tedy o určité klíčové slovo, nebo seskupení znaků. Znaky jména tokenu jsou vstupem pro syntaktickou analýzu textu. Obecně se doporučuje referovat dále o tokenu právě jeho jménem." (13)

Vlastnosti tokenů

Jednotlivé tokeny musí mít vlastnosti, podle kterých je možné jednotlivé tokeny od sebe odlišit či naopak sdružovat do skupin.
Například text „Vstávám každé ráno v 6 hodin. Na univerzitu jezdím autobusem číslo šest.". Pro vysvětlení se zaměříme pouze na čísla v textu. V textu se nachází číslo „6" a „šest". Každé je napsané jinak, ale obě dvě jsou tokenem druhu číslo. Takovýto token by měl kromě druhu mít určitě ještě dva parametry, a to hodnotu a popis.

V hodnotě "hodnota" by bylo číselné vyjádření a v popisu to, jak se dané číslo zapisuje. Čím více vlastností jednotlivým tokenům umíme dát, tím více informací dokážeme následně vytěžit z celého textu. (13)

Ukázka druhů jednotlivých tokenů

TOKEN	POPIS	UKÁZKA
if	characters i, f	IF
else	characters e, l, s, e	else
comparison	< or > or <= or >= or == or ! =	<=, ! =
id	letter followed by letters and digits	pi, score , D2
number	any numeric constant	3.14159, 0, 6.02e23
literal	anything but ", surrounded by "'s	"core dumped"

Tabulka 1: Tokeny (13)

Tato tabulka ukazuje některých druhů tokenů. Definice toho, jak bude jednotlivý token definován, je čistě na potřebách daného hledání. Tudíž to, co pro tokenizer 1 je slovo, může být pro tokenizer 2 číslo a naopak. Například řetězec znaků „:, -,)“ by před třiceti lety byl nedával nikomu velký smysl, ale dnes má tento řetězec však význam jako „emoji“

5.2 Stemming

Proces stemming má za cíl najít u daných tokenů, které jsou slovního typu, a tudíž jsou tvořeny řetězci písmen, jejich kořen pomocí odstraňování přípon.

“Odstraňováním přípon automatickými prostředky je opera obzvláště užitečná především na poli získávání informací. V typickém informačním prostředí, je dána množina dokumentů, které jsou popsány slovně v titulu daného dokumentu. Popřípadě je jejich popis získán z abstraktu dokumentu. Pokud připustíme riziko mírné nepřesnosti vlivem původu slov, můžeme říci, že dokumenty jsou

reprezentovány vektory slov. Kořeny těchto vektorů budou mít většinou podobný smysl například:

CONNECT
CONNECTED
CONNECTING
CONNECTION
CONNECTIONS“ (14)

Metody stemming se používají za účelem uskupování slov stejného nebo podobného významu. Na základě stemmingu však je možné sdružovat slova pouze morfologicky podobná. Tokeny je potřeba dále zpracovávat z hlediska jejich významu. Pokud bychom se spoléhali pouze na stemming u slov jako „obchodní smlouva“ a „obchodník pracující“ nastal by problém, jelikož sice mají stejný kořen, ale význam je jiný, jedno je listina a druhé je zaměstnání.

5.3 Lematizace

Lematizace má za cíl stejně jako stemming najít kořen slova. Kde však stemming používá hrubou sílu odstraňování sufixů, tam lematizace používá doménový přístup, při němž z daného slova (tokenu) vytvoří lemu v závislosti na okolním textu.

„Lema je kolekce lexikálních LEMMA forem, které mají stejný kořen, stejný účel ve větě a stejný význam slova.“ (15)

„Lematizace je proces redukování skupiny slov do jejich lem, popřípadě do slovníkové formy. Tento proces bere v úvahu větný druh slov, význam daného slova, také význam daného slova ve větě, ve které se nachází a význam slova pro okolní věty. Tyto faktory jsou brány v potaz před zredukováním slova do jeho lemy. Pro ukázku lze uvést názorný příklad:

- **Krásný** a **krásně** budou lematizovány do **krásný** a **krásně**.
- **Dobrý**, **lepší**, **nejlepší** však budou všechny lematizovány na **dobrý**.“ (16)

Těmito teoretickými základy byly v této diplomové práci nastíněny algoritmy pro práci s textem.

Z těchto algoritmů vychází aplikace IBM Knowledge studio, která je zakomponována do tvořeného modelu na zpracování nestrukturovaného textu.

6 IBM Knowledge studio

IBM Knowlege studio je cloudově[4] zaměřená aplikace, která umožňuje vývojářům a doménovým expertům spolupracovat na tvorbě vlastních anotačních komponent, které mohou být využity k identifikování výskytu a vazeb v nestrukturovaném textu. (17)

Watson Knowledge Studio se používá pro tvorbu modelů strojového učení, které porozumí lingvistickým nuancím, významu a specifickým vazbám v dané doméně nebo k vytvoření plavidlového modelu pro nacházení entit v dokumentech. (18)

Entitní a vazbové pracoviště se používá pro vytvoření vlastního entitního systému a k vytrénování vlastního modelu, který rozeznává specifické entity v textu. S modelem pro strojové učení, můžete definovat specifické typy vazeb a vytrénovat model k jejich rozeznávání mezi dvěma entitami. (18)

IBM Knowledge Studio poskytuje jednoduché nástroje pro anotování nestrukturovaného textu ze specifické domény, a využití těchto anotací k vytvoření vlastního modlu strojového učení, který porozumí dané doméně. Přesnost modelu se zlepšuje skrz opakující se testování. Hlavní výsledkem je schopnost algoritmu učit se a rozeznat opakující se prvky ve velkém množství dokumentů. (18)

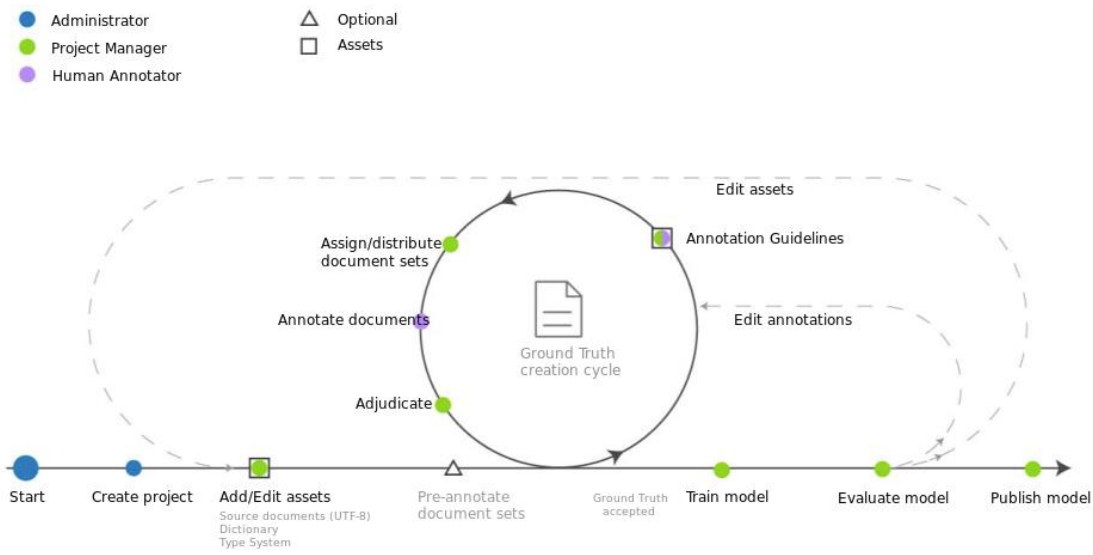
6.1 Postup tvorby modelu

Model se tvoří rámcově tímto způsobem (18):

- Na základě množiny dokumentů spojených s určitou doménou. Vývojáři vytvoří systém definující entitní typy a typy vazeb pro informace, které jsou zajímavé a podstatné pro danou doménu.

[4] Uživatel danou aplikaci nemá na svém počítači. Aplikace běží na vzdáleném serveru a uživatel má přístup pouze k rozhraní dané aplikace.

- Skupina anotátorů začne ručně anotovat část z množiny dokumentů. Přidělováním štítků k určitým slovům, aby určili jejich entitní druh. Následně tito anotátoři spárují slova, která mají mezi sebou vazby, významné pro danou doménu. V poslední řadě se provede určení koreference[5], spárování slov, která odkazují na stejnou věc, či osobu stejného entitního druhu. Následně se porovnají práce anotátorů na daných dokumentech, kdy více anotátorů anotovalo stejný dokument a určí se, které anotace budou uznány. Výsledkem je jeden anotovaný dokument. Ten je následně předán do základní pravdy.
- Pomocí základní pravdy se trénuje model.
- Vytvořený model se následně využívá k hledání entit, vazeb a koreference v zatím neanotovaných dokumentech.
- Jakmile je tým s modelem spokojen, předá se do servisní služby poskytující NLP [6] služby.



Obrázek 2 Anotační cyklus (18)

[5] Vyjádření jednoho jevu použitím různých slov, například synonym.
[6] Natural language processing

6.2 Dokumenty

K vytvoření modelu je potřeba nashromáždit dokumenty s obsahem reprezentujícím určitou doménu, pro kterou bude model určen.
Dokumenty k vytrénování daného modelu by měly obsahovat co možno nejvíce relevantních slov, která mohou být oanotována. Je dobré se držet těchto bodů (18):

- Je dobré nashromáždit dokumenty, které budou dohromady čítat okolo 300 000 slov. Pokud je doména složitějšího charakteru, je dobré mít tuto základnu dokumentů větší, a naopak je možné mít menší základnu při jednoduší doméně.
- Jeden dokument by měl být omezen na jednu či dvě stránky. S celkovým počtem dvou tisíc slov. Nejlépe by však dokumenty měly mít okolo jednoho tisíce slov. V počátcích vývoje modelu je dobré mít v dokumentu pouze pár paragrafů. Zpracovávání delších dokumentů je samozřejmě možné, avšak zkoušet označovat koreference skrz více stránkové dokumenty se nedoporučuje.
- Vývojáři by se měli ujistit, že dokumenty obsahují celou škálu všech možných entitních typů, sub-typů a vazeb mezi nimi. Cílem by mělo být mít více než padesát anotací od každého entitního typu a padesát výskytů od každého typu vazby v základní kolekci dokumentů.
- Kolekce pro vytvoření cvičebního prostředí pro model musí obsahovat aspoň deset anotovaných dokumentů.

Dokumenty jsou následně pospojovány do pracovních sad. Tyto sady se používají pro trénování modelu, kdy se sady rozdělí do tréninkových dat, testových dat a slepých dat. Oddělení datových sad je důležité pro hodnocení výkonu modelu. (18)

Dokumenty se do Knowledge Studia mohou nahrávat ve vícero formátech.

CSV

Formát CSV se dá použít pro nahrání vícero dokumentů v jednom CSV souboru. Formát CSV je sloupcový formát, kdy se pro oddělování sloupců používá znak čárky "," a samotný obsah sloupce se nachází v uvozovkách ". V prvním sloupci se má nacházet název daného dokumentu. V druhém sloupci pak obsah dokumentu. (18)

PDF

Získání dat z formátu PDF nemusí vždy proběhnout úspěšně. V závislosti na tom, jakým způsobem bylo dané PDF vytvořené. Příkladem takové nemožnosti získat text z PDF je, když je znaková sada PDF neslučitelná se znakovou sadou unicode[7]. (18)

Formátovaný text

Formátovaný text je možné použít pouze v případě, že po převedení z formátované podoby do neformátované, se nevyskytují nežádoucí defekty v textu. Příkladem takového defektu mohou být tabulky ve formátu DOCX, nebo spojník slov v PDF. V případě, že naše doména obsahuje takovéto dokumenty, je vhodné zařadit předzpracování takovýchto dokumentů do obyčejného textu. (18)

Dokumenty exportované z Knowledge Studia

Je možné stáhnout dokumenty z již existujícího pracovního prostoru jednoho Knowlege Studia a použít je v jiném. Soubory se z Knowlege Studia stahují zabalené ve formátu ZIP. Následně je možné celý balík nahrát do jiného pracovního prostoru. (18)

[7] Standardní kódování textu.

6.3 Typový systém

Typový systém definuje vše, co je nějakým způsobem zajímavé v obsahu dané domény a co je potřeba označit anotací. Typový systém kontroluje, jak může být obsah anotován. Definuje entitní typy, které mohou být přiřazeny jako štítek. Definuje označení vazeb mezi různými nebo stejnými entitami. O vytvoření typového sytému se typicky starají vývojáři aplikace, ve které bude používán a experti na danou doménu. (18)

Typový systém pro Knowledge Studio může být vytvořen od základu, nebo může být importován z jiného Knowledge Studia a následně upravován. (18)

Zmínka

Za zmínku se považuje text, který je relevantní pro data dané domény. Například v typovém sytému o motorových vozidlech jsou za zmínku považovány termíny jako *„bezpečnostní pás, volant“*. (18)

Entitní typy

Entitní typ je, jak jsou kategorizovány věci reálného světa. Entitní zmínky jsou příklady takových typů. Pro názornost může být použit následující příklad: (18)

- zmínka „*Prezident Zeman*“ může být anotována jako *Osoba,*
- zmínka „*IBM“* může být anotována jako *Organizace*.

Entity jsou často podstatná jména, ale mohou to být také slovesa. Pokud je sloveso důležité k zachycení účelu dané aplikace, je následně zařazeno do typového systému. Například:

- *událost_srážka* může být validní typ entity pro systém o automobilových vozidlech, stejně jako podstatné jméno n*áraz,*
- příkladem může být věta *„Auto narazilo do bariéry“.* (18)

Role

Je dobré kvalifikovat zmínky podle kontextu, ve kterém se vyskytují. Pro příklad, zmínka *„Francie“* ve větě *„Student navštívil Francii“*. V této větě je zmínka *„Francie“*, entitního typu *GPE*[8]. Avšak, „Francie“ je v tomto kontextu také myšlena jako destinace, kam studenti cestují, tak je entitní typ upraven přidáním vlastnosti *Role*. Vlastnost *Role* bude mít hodnotu *Location*. Například zmínka “právník” může být označena jako entitní typ *Lidé* a také *Role* zaměstnání. (18)

Entitní pod-Typy

Hlubší klasifikování daného entitního typu. Na příklad entitní typ *Organizace* má pod typ *Vládní, Soukromá*. (18)

Typ zmínky

Určuje, jakého slovesného typu je daná zmínka: (18)

- **NAM** jsou názvy osob, míst, organizací.
- **NOM** jsou obecná podstatná jména.
- **PRO** jsou zájmena on, ona atd.
- **NONE** jsou všechna zbylá slova, která nebylo možné zařadit do předchozích třech typů.

Třídy zmínek

Třídy kvalifikují zmínky podle jejich specifického zařazení ve větě: (18)

- **SCP** zmínka je specifická. Takové slovo je často název, popřípadě je z kontextu dáno, že se jedná o určitou věc. Například *moje auto je červené,* kdy je myšleno jedno konkrétní auto.
- **GEN** zmínka je obecná. Slovo ve větě nepředstavuje konkrétní objekt, ale obecnou skupinu. Na příklad *všechna auta jezdí na palivo*.

[8] Geo-political entity/ zeměpisně politický objekt.

- **NEG** zmínka je negace. Aby mohl být model vycvičen k rozeznávání záporných vět, jako *nemám rád, nechci atd.,* je nutné mít možnost takovou zmínku označit.

Typy rolí

Typy rolí představují binární, seřazenou vazbu mezi dvěma entitami. K tomu, aby mohla vazba existovat, musí být definován její název spolu s pořadím dvou entit, které budou spolu svázány. Entity, které jsou takto spolu svázány, musí být v jedné společné větě. Například ve větě *Jana pracuje pro IBM,* je vazba *zaměstnanáV.* Na pořadí entit záleží, jelikož vazba *Jana zaměstnanáV IBM,* dává smysl, ale I*BM zaměstnanáV Jana* již ne. Pokud je však vazba mezi dvěma stejnými entitami, tak na pořadí již nezáleží. Vazba *sourozenec,* bude platit, ať vychází z kteréhokoli sourozence. V případě takovýchto vazeb, kde nezáleží na pořadí entit, je doporučováno držet pořadí, které je v textu. Tudíž vazba bude vycházet z prvně zmíněné entity. (18)

6.4 Slovníky

Ve strojovém učení slovníky slučují dohromady slova a fráze, která mají podobný nebo stejný význam. Slovník je také vždy přiřazen některému z entitních typů. Záznam ve slovníku neznamená, že všechny výrazy daného slova znamenají to samé, ale že se model ke slovům v daném záznamu chová stejně.

Slovník je list slov nebo frází, která jsou stejná z pohledu získávání informací ve smyslu, že jsou zaměnitelná pro potřeby identifikování entit a významu vazeb.

V úvahu přichází příklad, kdy do záznamu ve slovníku vložíme sedm dní v týdnu. K označení takové zmínky v dokumentu, anotátor přidělí entitní typ *den_v_týdnu* zmínce v textu jako je P*ondělí* nebo P*átek.* Jelikož slovník zrovnoprávňuje všech sedm dní v týdnu, pomůže to k vytvoření modelu v jeho samostatném anotování zbylých dnů, jako jsou *Úterý, Sobota.* V další řadě díky slovníku se zajistí, že model bude správně anotovat i slova, na která nebyl specificky vycvičen. Jako jsou *Čtvrtek* a *Středa.* Tato synergie napomáhá v získávání informací

z textu, kdy se model naučí souvislosti na části a následně je dokáže aplikovat na celku. (18)

Využití slovníků

Slovníky se dají využít ve vícero případech, jejich využití je však čistě nepovinné. Slovníky jsou využívány strojovým učením při tvorbě modelu a poskytují mu slova nebo fráze se stejným významem pro vytěžování informací. Dále se také využívají při před-anotováním k usnadnění práce anotátorů. (18)

Užití slovníků při strojovém učení

Entitní typ přiřazený k danému slovníku není použit k definování pravidla v modelu. Při učení modelu se zmínky v dokumentech vyhodnocují nezávisle. Tudíž to, že se slovo dané zmínky nachází v některém slovníku, ještě neznamená, že bude automaticky přiřazen entitní typ daného slovníku. Informace, že se takové slovo v daném slovníku nachází, se však vezme v potaz, spolu s dalšími informacemi, které budou získány po lingvistické analýze. Dokonce, pokud se žádný ze záznamů v slovníku nevyskytne v obecné pravdě dokumentů, slovník nebude použit vůbec ve výsledném modelu. (18)

Užití slovníků v před-anotační fázi

Použití slovníků k přímému přiřazení entitních typů ke zmínkám, pokud se slovo nachází ve slovníku. Toto přiřazení funguje i na pravidlově-řízené třídy. Toto rychlé přiřazení má za účel pomoci anotátorům k rychlejšímu zpracovávání dokumentů. Když je část již známých slov a spojení anotována předběžně, anotátor má více času soustředit se na zbytek, zatím pro model neznámé zmínky. Anotátor má však vždy možnost entitní typ, který byl přidán na zmínku v rámci před-anotování nahradit takovým, jaký považuje za správný. (18)

6.5 Pravidlově-řízené třídy

Pravidlově-řízené třídy jsou používány k rozeznávání paternů v dokumentech. K vyhledávání paternů v dokumentech se používají pravidla.

Pravidlově-řízené typy jsou tvořeny pravidly, která se používají k reprezentování různých typů informací. Tato pravidla jsou podobná entitním typům. Avšak jsou od nich separována. Důvodem této separace je fakt, že některá pravidla jsou vytvořena pouze za účelem, aby byla částí složitějších pravidel. Tudíž jsou tato pravidla samostatně nepoužitelná. Používají se tedy pouze v kombinace s dalšími nesamostatnými a samostatnými pravidly a k definování více užitečných a samostatně již fungujících tříd. Tyto mezi třídy jsou nezbytné, avšak vystavovat samostatně nefungující části do typového systému by mohlo být nebezpečné, proto jsou takto separovány. Jakmile však je vytvořeno komplexní pravidlo, které je již schopné samostatně fungovat, je možné ho namapovat na některou z entitních tříd. Následně je možné takové pravidlo používat i v předanotační fázi.

Jako příklad se uvádí tvorba pravidla pro rozeznávání jmen lidí. K vytrénování modelu, který rozená lidská jména je možné dojít dvěma cestami.

- Anotátoři budou anotovat mnoho jmen, v různých formách, entitním typem *Jméno.* Na závěr by se modelu ukázaly všechny možné varianty a on by je rozpoznával.
- Druhou možností je vytvoření pravidla, kterému se naopak řekne, jak mohou jména vypadat. (18)

K vytvoření pravidlově-řízené třídy na rozpoznávání lidských jmen je nutné vytvořit nejprve pravidlo, které popisuje textové vzory toho, jak je možné napsat lidské jméno. Tudíž se nejprve vytvoří pravidlo *KřestníJméno* a *Příjmení.* Tyto třídy budou však pouze pomocné. Třída pro skutečné rozpoznávání jména bude *CeléJméno* a bude se skládat z dvou předchozích. Výhodou takto složeného pravidla *CeléJméno* také je, že je možné k němu přidat další nesamostatné pravidlo bez zásahu do ostatních. Takové pravidlo by mohlo být na tituly před jménem, nebo za jménem. Výsledné pravidlo *CeléJméno* by následně bylo namapováno na entitní třídu *Jméno*. (18)

Důležité je také namapovat pouze finální a komplexní pravidla. Například pravidla jako jsou *KřestníJméno a Příjmení* by se nikdy samostatně neměla dostat do typového systému. Když by však byla do typového systému přidána, docházelo by k zmatení modelu, jelikož by jednu zmínku bylo možné zařadit až k příliš mnoha pravidlům. Tato situace nastává i v případě, že by všechna pravidla, *CeléJméno, KřestníJméno, Příjmení,* byla mapována na entitní typ *Jméno.* (18)

- Při tvorbě pravidel je možné využívat Slovníky.
- Při tvorbě pravidel je možné využívat regulární výrazy, které odpovídají syntaxi java.util.regex.Pattern
- Do pravidel je možné vkládat konstantní text.

6.6 Základní pravda

Účelem anotování dokumentů je vytvoření základní pravdy, což je množina ověřených dat, která jsou použita k přizpůsobení Watsona k dané doméně. V Knowledge Studiu anotátoři, což by měli být experti v oblastech cílové domény, mají klíčovou roli při tvorbě základní pravdy.

Tvorba Základní pravdy obsahuje následující kroky: (18)

- Anotátoři pošlou zpracované dokumenty k revizi.
- Seniorní doménový expert v manažerské pozici, zkontroluje přesnost anotování dokumentů. Porovná důslednost anotací dokumentů, které se překrývají mezi pracovními kolekcemi.
- Pokud je mezi-anotační skóre příliš nízké, je sada dokumentů odmítnuta. Odmítnuté dokumenty jsou zaslány zpět anotátorům k úpravě. Pokud je odmítnut jeden dokument v sadě, je vrácena celá sada.
- Pokud seniorní doménový expert potvrdí správnost anotací na skupině dokumentů, jsou dokumenty, které se neprolínají napříč různými sadami přesunuty do základní pravdy. Dokumenty v základní pravdě mohou být použity při učení modelu.

- Pokud se dokument vyskytuje ve více sadách, seniorní doménový expert vyřeší tento konflikt v anotacích. V této rozhodovací fázi je dobré přizvat ostatní anotátory. Je totiž nutné objasnit, proč anotátoři odlišně anotovali daný dokument a určit, jaká anotace bude posunuta do základní pravdy.

Je nutné mít na paměti, že anotování vždy vyžaduje posouzení. Anotační manuály mohou velice pomoci při zajišťování správnosti a celistvosti anotací, avšak i nejlepší manuály jsou vždy vykonávané lidmi. K ujištění, že do základní pravdy se dostávají pouze kvalitní data, je nutné vzdělat anotátory v této problematice. Čím lépe budou anotátoři proškolení, tím lepší a kvalitnější mohou být jejich úsudky ohledně anotování obsahu dané domény. (18)

Mezi-anotační skóre

Následně poté, co anotátoři anotovali dokumenty, je nutné rozhodnout, které dokumenty budou předány do základní pravdy. Tento proces začíná zkontrolováním mezi-anotačního skóre. Dokumenty s nízkým skóre by měli být odmítnuty, znovu zrevidovány a předány nazpět anotátorům k přepracování.

Při počítání mezi-anotačního skóre, systém kontroluje všechny překrývající se dokumenty ve všech skupinách, které byly anotovány v rámci úkolu, nehledě na stav úkolu. Není možné přijmout nebo odmítnou skupinu dokumentů, dokud nejsou ve stavu Odeslané. Je žádoucí neprovádět výpočet mezi-anotačníchho skóre, dokud všechny skupiny dokumentů nejsou právě ve stavu „Odesláno“. Popřípadě je žádoucí omezit výpočet pouze na anotátory, kteří již dokončili své skupiny dokumentů.

Mezi-anotační skóre ukazuje, jak různí anotátoři anotovali zmínky, vazby, a koreference. Toto skóre je možné vidět a porovnávat v párech. Tudíž se vždy porovnává skóre anotátora A proti skóre anotátora B. Zároveň je možné zobrazit skóre anotátorů u specifického dokumentu.

Mezi-anotační skóre napomáhá k odhalování oblastí, kterým by se měla věnovat větší pozornost. Skóre, které klesne pod určitou hodnotu, se následně zobrazí červeně. Tuto kritickou hodnotu je možné nastavit. V prvních fázích anotování dané domény se předpokládá, že mezi-anotační skóre bude horší u vazeb

než u zmínek. Toto je často způsobené tím, že nejprve je nutné odladit anotování zmínek a až následně vazeb mezi nimi.

Mezi-anotační skóre se udává jako Fleissův kappa[9] koeficient. Toto skóre reprezentuje, jak byl jeden entitní typ používán k anotování zmínek napříč všemi dokumenty v daném úkolu. Hodnota skóre může být v rozmezí -1 až 1, toto rozmezí pomáhá identifikovat slabá místa v metodice anotování, či u jednotlivých anotátorů. (18)

Tabulka níže může sloužit jako výchozí bod pro hodnocení výkonů.

Score	Agreement level
< 0	Poor
.01 - .20	Slight
.21 - .40	Fair
.41 - .60	Moderate
.61 - .80	Substantial
.81 - 1.0	Perfect

Tabulka 2 Anotační skóre (18)

6.7 Anotování

Anotování je proces, při kterém anotátoři anotují, tudíž přiřazují zmínkám označení, jakého jsou typu v rámci typového systému. To, jaké dokumenty bude anotátor anotovat, je určeno jeho úkoly. (18)

Úkoly jsou vytvářené managerem projektu. Úkoly jsou při vytváření přiřazeny na jednotlivé anotátory. Úkoly se skládají z těchto částí: (18)

- Názvu daného úkolu.
- Stanoveného termínu do, kdy musí být úkol zpracován.
- Určení anotátora, kterému je úkol přiřazen.

[9] Koeficient shody.

- Kolekce dokumentů, které musí být v rámci úkolu zpracovány.

Vytvoření kolekce dokumentů je samostatná část. V té se určuje název dané kolekce a zároveň se určuje, které dokumenty budou do této kolekce zařazeny. Dokumenty však mohou být zároveň ve více kolekcích. Tím následně vzniká překrývání, při kterém je nutné rozhodnout, které anotace budou uznány a předány do základní pravdy. Proces rozhodování o překrývání je popsán výše, v kapitole o základní pravdě. (18)

Následně poté, co je úkol vytvořen, se automaticky zobrazí přiřazenému anotátorovi, který ho může začít zpracovávat. (18)

Anotátoři by se vždy měli snažit anotovat celý dokument a všechny dokumenty v daném úkolu. Při tvorbě modelu se strojové učení totiž učí i z negativních příkladů, tudíž též i z toho co není anotováno. Anotátoři musejí nalézt rozumnou míru mezi tím co anotovat a co už ne. Pokud by anotátoři zodpovědně anotovali pouze prvních pět z deseti dokumentů z kolekce a zbylým dokumentům by se již nevěnovali, tak by následně na základě takto nezachycených anotacích ve zbylých dokumentech vycvičili model k ignorování takovýchto entit nebo vazeb. Toto vynechání by nakonec mohlo zvrátit celý proces a důkladná práce v první části kolekce by neměla žádný smysl. (18)

Konzistentní anotování v dokumentech má stejnou prioritu jako správnost anotování. Mnoho rozhodnutí o způsobu anotování je plně libovolných na daném týmu anotátorů. Příkladem by mohlo být anotování názvu aut. Buď se může název modelu auta skládat pouze z modelu *Camry,* nebo může být přidán i ukazatel výbavy *Camry LX*. Tudíž nastávají dvě možnosti, buď bude *Camry LX* celé označeno jedním entitním typem, nebo *Camry* jako jeden entitní typ a *LX* jako druhý entitní typ. To, jaký způsob bude vybrán, je pro tvorbu modelu nedůležité, avšak vybraný způsob musí být následně striktně dodržován. (18)

Při označování zmínek by se měly striktně dodržovat hranice slov. Jelikož automatická detekce zmínek vyhledává slova na této úrovni. (18)

Označování zmínek entitními typy, by se mělo omezit na jedno nebo dvě spojená slova, vždy když je to možné. (18)

Anotátoři by se měli vyvarovat překrýváním entitních typů a jednotlivých zmínek. Překrývaní nastává, když anotátoři označí jednu zmínku v dokumentu dvěma různými entitními typy. Pro ilustraci by mohla být užita věta*:*
She donated her father's journals to the JFK Library. Překrytí by v této větě nastalo, kdyby anotátoři označili *JFK* jako typem *Osoba* a zároveň *JFK Library* jako *Místo* v téže větě. Anotátoři tedy musí vzít v potaz význam věty a ne jen pouze slepě anotovat samotná slova. Správně by tedy měli použít entitní typ *Místo* pro celek *JFK Library.* Takovéto více slovní názvy mají výjimku v předchozím pravidle, které hovoří o tom, že by se neměly anotovat více než dvou slovné fráze. Rozklíčování takovýchto víceslovných struktur však vyžaduje vícero cyklů strojového učení, než se je model naučí správně rozeznávat (18).

Anotační tým se musí také rozhodnout, jaké stanovisko zaujme při práci se skupinami a množnými čísly v textu. Typový systém může mít například dva odlišné entitní typy pro jednu veličinu, jakou mohou být typy *Osoba* a *Lidé. J*akmile se v textu objeví skupina *Pavel, Martin a Jana šli do obchodu,* anotátoři si mohou vybrat, jestli označí každého z této skupiny samostatným typem *Osoba* nebo celou skupinu jedním typem *Lidé*. To, jaký způsob si anotátoři na začátku zvolí, je pro model nepodstatné, avšak je nutné tento přístup držet po zbytek anotování ve všech dokumentech. (18)

Koreference se při anotování používají k označení jednoho a toho samého objektu reálného světa, vyskytujícího se na více částech dokumentu. Je důležité mít na paměti, že jakmile označíme zmínku jako koreferentní k jiné zmínce, už bychom tyto dvě zmínky neměli spojovat žádnou vazbou. (18)

6.8 Nasazení modelu a odkazování se na něj

Pokud je model připraven být nasazen a používán pro analyzování textů dané domény je možné vytvořit jeho verzi, kterou pak administrátoři daného projektu nasadí do servisní služby. (18)

Před nasazením modelu do servisní služby je však nutné službu založit na IBM Cloude a zároveň jí mít ve stejném regionu jako IBM Knowledge Studio.

Vytvoření modelu pro exportování se provádí vytvořením verze. Při vytváření verze se automaticky vygeneruje *Model ID,* toto id je unikátní identifikátor daného modelu. Vygenerovaná verze modelu může být nasazena na libovolný počet instancí servisních služeb, nejenom Natural Language Understanding, které se bude věnovat následující kapitola. Stejná verze modelu může být nasazena i na více typově stejných servisních službách. Na druhou stranu jedna servisní služba, například Natural Language Understanding může v sobě nést vícero modelů, které jsou právě rozlišeny svým *Model ID*. Jakmile se ve službě nachází daný model, vývojáři aplikace, která tento model chce využívat, se na model odkazují v HTTP požadavku pomocí elementu *model_id*. Do tohoto elementu se vkládá *ModelID*. (18)

6.9 Tvorba vlastního modelu

Původním záměrem praktické části této práce bylo vytvoření vlastního modelu na zpracování nestrukturovaného textu.

Tvorba modelu probíhala v aplikaci IBM Knowledge Studio. Zabudovaný systém správy rolí a přidělování práce však nemohl být využit, z důvodu nedostatečného počtu členů vývojového týmu. Vytvoření nového modelu pro účely praktické části se ukázalo jako nerealizovatelné. Hlavním důvodem bylo nenalezení společného vzoru sdíleného napříč knihami. Výsledný model byl tedy velice obecný, a přesto nedosahoval kvality obecného modelu IBM. Přidání slovníků do modelu bylo vyhodnoceno jako nepříliš přínosné, jelikož většina knih nesdílí stejné názvy, pokud se nejedná o knihy stejného autora, či knihy v jedné sérii.

Vzhledem k nereálnosti vytvoření funkčního vlastního modelu bylo tedy potřeba nalézt již existující model. Pro demonstrativní ukázku byl tedy použit již existující model od IBM.

7 Natural Language Understanding

Služba IBM Natural Language Understanding, v překladu znamená porozumění přirozenému jazyku. Tato služba je vyvíjená a poskytována společností IBM. Dále se v textu bude o Naturel Language Understanding hovořit jako o službě. Službu je možné používat prostřednictvím cloudového rozhraní. Vytvoření této služby a její spravování se provádí přes webový portál IBM[10]. Po vytvoření služby uživatel získá její adresu a specifický klíč k této službě. Díky adrese a klíči je možné zasílat na službu texty k analýze.

Relevantnost a množství vytěžených informací, které dokáže služba z textu vytěžit, se odvíjí od modelu používaného službou. Tvorba a nasazení modelu do služby je popsána v kapitole “Nasazení modelu a odkazování se na něj". Informace vracející službou se týkají určitých aspektů textu. Tyto aspekty budou následovně představeny.

V následujícím textu budou použity praktické ukázky a výsledky z knihy IBM (19). Výsledky z ukázek se mohou lišit, nemusejí být znovu reprodukovatelné, v závislosti na použitém modelu. Principiální fungování jednotlivých aspektů však není modelem ovlivněno.

7.1 Kategorie

Kategorizace vstupního textu, HTML nebo webového obsahu do hierarchické taxonomie pomocí pěti úrovňové klasifikační hierarchie. Služba vrací první tři kategorie. Seznam kategorií je uveden na stránkách IBM[11]. V době psaní této práce bylo možné text kategorizovat do 1038 kategorií. (19) (20)

[10] https://cloud.ibm.com/
[11]https://cloud.ibm.com/docs/natural-language-understanding?topic=natural-language-understanding-categories-hierarchy

Ukázka fungování:
Zdrojovým textem je věta „*Machine learning is the science of how computers make sense of data using algorithms and analytic models.*“
Z takovéhoto textu služba z analyzovala kategorie: (19) (20)

- /science/computer science/artificial intelligence
- /science/
- /science/mathematics/geometry

Informace získané z kategorizace textu mohou být využity v zařazení textu do určitých složek a podsložek. V kombinaci s Konceptuální analýzou je možné vytvořit velice organizovanou kolekci dokumentů se snadným přístupem k odpovídajícím informacím. (19) (20)

7.2 Koncept

Koncepční analýza identifikuje vysoko úrovňový koncept, který nemusí být přímo zmíněn ve vstupním textu. (19) (20)

Ukázka fungování:
Zdrojem pro koncepční analýzu je věta „*Machine learning is the science of how computers make sense of data using algorithms and analytic models.*“
Výsledkem analýzy je následně množina tagů: (19) (20)

- Computer
- Machine learning
- Artificial intelligence
- Computer science
- Alan Turing
- Scientific method
- Psychology
- Learning

Příkladem získání informace o textu, která v textu přímo není zmíněna, je tag „Alan Turing“. Alan Turing je jedním ze zakladatelů počítačových věd, proto bylo jeho jméno s tímto textem spojeno. Jako zdroj tagů pro koncepční analýzu používá IBM knihovnu DBpedia[12]. (19) (20)

Získávání vysoko úrovňového konceptu může pomoci k lepšímu rozdělování textů do skupin, podobně jako kategorizace. (19) (20)

7.3 Emoce

Analýza emocí je schopná nalézt vztek, znechucení, strach, potěšení a smutek zmíněný v textu. Analýza může být provedena nad vícero částmi textu. Specificky se dá určit pro analýzu část textu, kterou vyhodnotili jako důležitou jiné části služby. Například jednotlivé entity či klíčová slova. Dále je možné získat celkové emoční zabarvení textu. (19) (20)

Zdrojovým textem pro analýzu emocí je věta, „I love apples! I don't like oranges.“

Analýza byla nastavena specificky pro slova *apples* a *oranges*. Následující výsledky byly tyto:

Pro slovo *apples:*

smutek: 0.028574,
potěšení: 0.859042,
strach: 0.02752,
znechucení: 0.017519,
vztek: 0.012855

Pro slovo *oranges:*

smutek: 0.514253,
potěšení: 0.078317,
strach: 0.074223,

[12] Jedná se o projekt s cílem získávat strukturovaná data z encyklopedických zdrojů. (28)

znechucení: 0.058103,
vztek: 0.126859

Pro jednotlivé emoce je vždy přidáno i skóre představující úroveň výskytu dané emoce. Toto skóre se pohybuje od nuly do jedné, kdy jedna je nejvíce. Význam hodnoty je určen použitým modelem. V době psaní této práce nebylo možné službě dodat vlastní model pro analýzu emocí. (19) (20)

Emoční analýza může být použita například v zákaznických centrech. Při zjišťování spokojenosti zákazníků s jejich službami a případném zkvalitnění služeb. Tyto informace by měli dopomoci k lepšímu porozumění potřeb zákazníků. (19) (20)

7.4 Klíčová slova

Analýza klíčových slov identifikuje důležitá klíčová slova z kontextu udaného textu. Klíčová slova z kontextu jsou typicky využívána pro indexační data, tvoření tagů, nebo pro vyhledávání a identifikaci. Informační hodnota klíčových slov se tedy nejvíce uplatňuje při vytváření organizačních struktur, jako jsou glosáře, pro vyhledávání podle nejvíce podobných slovních termínů. Služba pro zpracování klíčových slov automaticky identifikuje použitý jazyk ve zdrojovém textu. Nalezená klíčová slova jsou ohodnocena a seřezaná podle relevance. Relevance je číselná hodnota, pohybující se od nuly do jedné, kde jedna je maximum. U klíčových slov je možné dále analyzovat jejich sentiment a emoce. Obě hodnoty, sentiment i emoce, budou popsány v samostatné kapitole. (19) (20)

7.5 Vazby

Analýza vazeb identifikuje předmět, akci a objekt, se kterým je v textu interagováno. Následně po provedení sémantické analýzy rozdělí věty na větné členy, jako jsou předměty a slovesa. Služba následně provede další analýzy na získání entit, klíčových slov a dalších informací z textu. Díky těmto všem

informacím je služba schopná rozeznat, že dvě entity jsou na sebe navázané a také jaký typ vazby je mezi nimi. Typy vazeb, které služba rozpoznává, jsou určeny modelem. Je možné službě předat vlastní model. Služba však již obsahuje základní model, v době psaní této práce základní model obsahoval 32 typů vazeb. (19) (20)

Jako zdrojová věta bude použita tato: „*Bob Dylan won the Nobel Prize in Literature in 2016. Bob Dylan was born in Duluth, Minnesota*.“ (19) (20)

Služba následně našla tyto vazby: (19) (20)

- "affectedBy" relation between “Bob Dylan” and “won”
- "timeOf" relation between “2016” and “won”
- "awardedTo" relation between "Nobel Prize" and "Bob Dylan"
- "bornAt" relation between "Bob Dylan" and "Duluth"
- "locatedAt" relation between "Duluth" and "Minnesota"

Analýza vazeb je použitelná k rozeznávání, že dvě entity jsou na sebe navázané v určitém typu. Tyto informace mohou představovat vazbu, že někdo něco potřebuje. Díky vazbám organizace mohou lépe porozumět potřebám svých zákazníků. (19) (20)

7.6 Sentiment

Analýza sentimentu identifikuje postoje, názory a pocity v kontextu textu předeného ke zpracování. Analýzou je možné hodnotu sentimentu získat jak ze specifické části textu, tak z textu jako celku. Tedy z dokumentu jako z celku. Je možné také získávat sentiment jednotlivých entit či klíčových slov. Analýza sentimentu je prováděna na základě modelu. V době psaní této práce, IBM Watson NLP možnost použít vlastní model pro analýzu sentimentu podporuje zatím pouze ve fázi testování. Služba NLP však obsahuje předpřipravený model pro analýzu. (19) (20)

Pro ukázku analýzy bude použita věta: (19) (20)
„I'm very upset about the quality of this product.“

Služba v zdrojové větě našla sentiment negativní hodnoty. Služba vrací slovní hodnocení sentimentu spolu s číselnou hodnotou od mínus jedné do jedné. Slovní hodnocení je rozděleno na negativní se zápornými hodnotami, neutrální s nulovou hodnotou a pozitivní s kladnými hodnotami.

- Negativní sentiment (skóre: -0.890748)

Analýzu sentimentu textu je možné použít pro vyhledávání stížností či pochval v hodnocení od zákazníků. Dalšími možností využití je získávání dat ze sociálních médií, kdy sběrem negativních příspěvků je možné zjistit určitá stanoviska lidí k tématům. (19) (20)

8 Aplikace Knihovna 2.0.

Aplikace demonstruje využití NLP v oblasti knihovnictví. V klasických knihovních vyhledávajících systémech je možné vyhledávat podle pevných atributů. Těmito atributy jsou například jména autorů, rok vydání, žánru či výskyt určitých slov.

Pomocí aplikace knihovna 2.0 je možné vyhledávat tituly podle jejich obsahu. Tudíž před hledáním nemusíme o daném titulu vědět vůbec nic. Budeme totiž hledat podle informací, které chceme v titulu najít. Toto hledání je podobné tomu, kdy se hledají tituly podle klíčových slov. Avšak hledání podle klíčových slov nebere v úvahu význam daného slova. Tudíž nám jsou vráceny i titulky, kde se daná klíčová slova objevují, ale význam je naprosto jiný.

8.1 Technologie

Aplikace Knihovna 2.0 je napsána v programovacím jazyce Java, verze 1.8. Programovací jazyk Java byl zvolen z důvodu jeho univerzálního nasazení napříč platformami. Jazyk Java dále umožňuje nejen vytvořit aplikaci zajišťující zpracování daného textu pomocí NLP, ale navíc i vytvoření ostatních komponent. Jako je například uživatelské rozhraní, práce s databází dané knihovny či napojení na systémy třetích stran.

V aplikaci Knihovna 2.0 je využit Framework Spring. Tento Framework usnadňuje vytvoření logické oddělení jednotlivých komponent aplikace, aby bylo zajištěno snadnější modifikování. Dále je přes Spring pracováno s databází. Databáze byla zvolena SQLite.

Analýza textu pomocí NLP se provádí pomocí IBM cloude service, ve které je vytvořená servisní služba Natural Language Understanding.

8.1.1 Spring

Aplikace Knihovna 2.0 používá knihovny Spring ve verzi 4.3.18.

Spring Framework poskytuje komplexní programovací a konfigurační model pro moderní podnikové aplikace založené na Javě – na jakékoli nasazované platformě.

Klíčovým prvkem Springu je infrastrukturní podpora na aplikační úrovni: Spring se zaměřuje na zapouzdření podnikových aplikací, aby se týmy mohly soustředit na logiku dané aplikace, než aby byly zbytečně svázány specifiky konkrétního prostředí, na kterém je aplikace nasazena. (21)

8.1.2 SQLite

SQLite je soustavně se rozvíjející knihovna, která je implementována soběstačně, tudíž bez serverově a ne konfiguračně. SQLite je transakční SQL databázová knihovna. Zdrojový kód pro SQLite je veřejně dostupný. Zároveň je zdarma k použití pro jakýkoli účel, ať již komerční nebo osobní. SQLite je v současnosti nejvíce nasazovanou databází na světě. (22)

SQLite je integrovaná databázová knihovna. Na rozdíl od ostatních SQL databází, SQLite nemá separované serverové procesy. SQLite čte a zapisuje přímo do obyčejného souboru na disku. Celá SQL databáze s několika tabulkami, indexy, spouštěči a pohledy, je obsažená v jednom souboru na disku. Tento databázový soubor je více platformní – můžete databázi volně kopírovat mezi 32-bitovým a 64-bitovým operačním systémem, nebo mezi architekturami big-endian a litte-endian[13]. Tyto funkce dělají z SQLite populární volbu jako aplikační souborový formát. Databázové soubory SQLite jsou doporučeným úložným formátem US Library of Congress. (22)

8.1.3 IBM Watson

Aplikace Knihovna 2.0 používá knihovny Watson ve verzi 7.3.1. Tyto knihovny slouží pro komunikaci s cloudovými službami IBM Watson.

[13] Způsob uložení dat v operační paměti.

8.2 Vnitřní fungování

V této kapitole budou představeny samotné procesy aplikace Knihovna 2.0. Jak bylo zmíněno v kapitole Technologie, aplikace používá pro své fungování vícero separátních komponent, které spolu propojuje. Hlavním účelem aplikace je zprostředkovávání již zpracovaných dat uživateli. Proto v úvodu kapitoly bude představen databázový model. Práce s těmito daty, jejich tvorba a předávání uživateli bude vysvětlena v podkapitole Services.

Aplikace Knihovna 2.0 je postavena na architektuře modelu MVC.

MVC, v plném názvu Model View Controller (Model Pohled Řadič), je architektura vytvořená pro vývoj webově zaměřených aplikací. Jak název naznačuje, aplikace se rozděluje do tří hlavních částí. Tradiční softwarová architektura pracuje na principu vstupních dat, jejich zpracování a výstupních dat, zatímco architektura MVC pracuje na principu řadiče, modelu a pohledu. U přístupu MVC architektury se při vývoji aplikace její aspekty rozdělují podle svých specifikací. V MVC jsou tyto aspekty UI logika, vstupní logika, procesní logika. Mezi těmito aspekty popřípadě elementy existuje tenká vazba. Podle této architektury, každý element musí existovat v aplikaci, ale nemají být pevně spojeny nebo propojeny. UI logika se stará o pohledy (část, kterou vidí uživatel) aplikace. Vstupní logika je obstarávána řadiči, představuje data přijímána od uživatele nebo jinak. Nakonec, procesní logika je zakomponována v modelu aplikace. Tyto volně vázané aspekty pomáhají vývojářům zvládnout komplikace při vyvíjení jakékoli webové aplikace. Zároveň umožňuje vývojářům, soustředit se implementaci jednotlivých částí separátně. Díky tomu se mohou vývojáři soustředit na implantování procesní logiky bez nutnosti mít již hotovou vstupní logiku a naopak. (23)

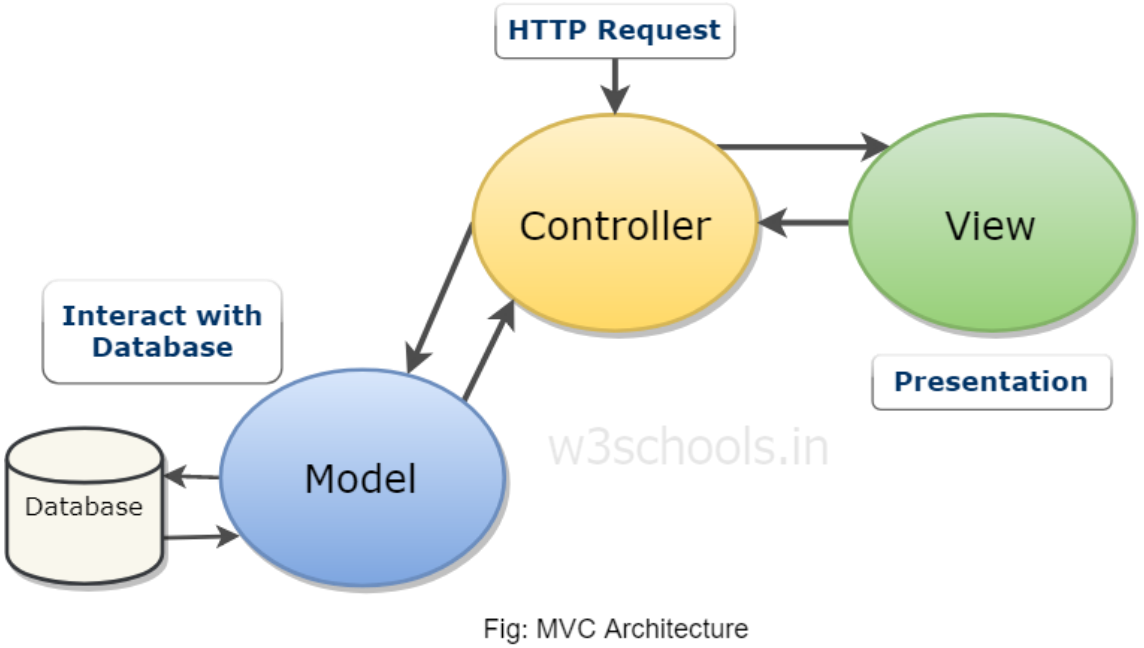


Fig: MVC Architecture

Obrázek 3: MVC architektura (23)

8.2.1 View

Jedná se o část aplikace, kterou vidí uživatel. Tato část je obecně složena z webových stránek. Tedy z html, css a skriptů na klientské části. Přes tuto část aplikace zobrazuje uživateli data. Zároveň jsou v této části nástroje, díky nimž uživatel zasílá své požadavky do aplikace. Tato komunikace se provádí přes protokol http. Uživatel do této prezenční vrstvy vyplní informace, které daná aplikace potřebuje k vykonání uživatelského požadavku. Následně jsou tyto informace přidány do http requestu a poslány do vrstvy řadiče (controler).

Pohledová vrstva následně obdrží od řadiče odpověď v podobě http response, kterou uživateli zobrazí.

Interface

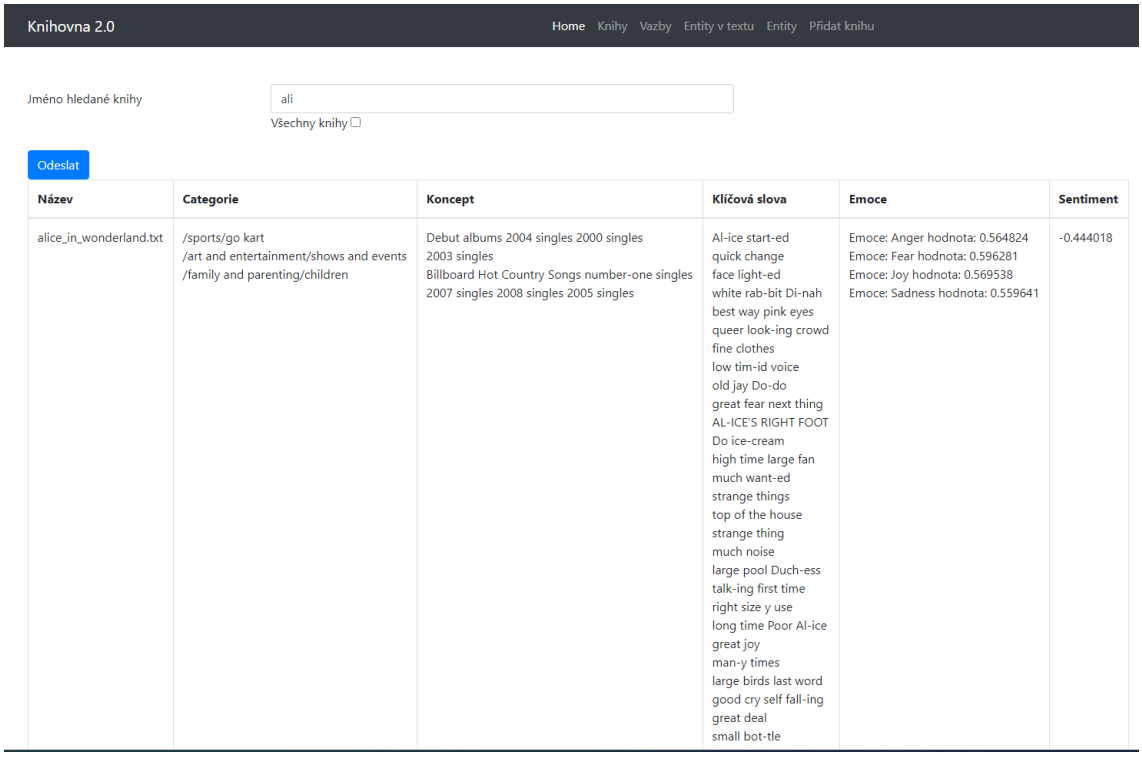
Aplikace komunikuje s uživatelem pomocí webového rozhraní. Po načtení webových stránek aplikace Knihovna 2.0, může uživatel v pravém horním rohu

nalézt menu stránek. V textu níže budou postupně představeny jednotlivé stránky a jejich funkcionalita.

Knihy

Stránka **knihy** zprostředkovává uživateli náhled na tituly, které jsou obsažené v aplikaci. Vyhledávání se provádí pomocí vstupního pole, do kterého uživatel zadává celý název, nebo aspoň část názvu hledaného titulu. Po zadání uživatel klikne na tlačítko „Odeslat“. Pokud by uživatel chtěl zobrazit všechny tituly obsažené v aplikaci, může zaškrtnout políčko „Všechny knihy“.

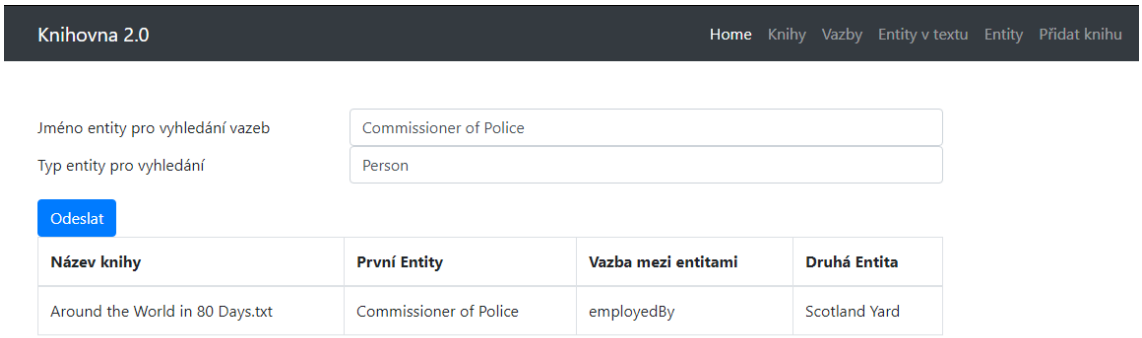
Aplikace následně vyhledá zadaný název ve své vnitřní databázi, a pokud nalezne shodu, zobrazí titul spolu s nashromážděnými informacemi. Aplikace zobrazuje název titulu (jedná se o název souboru, ze kterého byl text získán). Dále jsou zobrazovány nalezené kategorie, koncept, klíčová slova, emoce, sentiment. Bližší vysvětlení jednotlivých hodnot se nachází v kapitole 7 Natural Language Understanding a Zpracování souboru.



Obrázek 4 Interface: Knihy

Vazby

Vyhledávání vazeb v aplikaci Knihovna 2.0 se nachází na stránce **vazby**. Přístup na stránku vazby je stejný jako u stránky knihy, přes horní pravé menu. Pro vyhledání vazeb uživateli stačí zadat název hledané entity a její typ. Typ entit je nutný kvůli jednoznačné identifikaci entit. Jelikož jedním jménem je možné pojmenovat člověka i zvíře. Pokud uživatel neví, nebo si není jistý typem entity, může si její typ ověřit na stránce entity v textu. Fungování stránky Entity v textu bude vysvětleno v diplomové práci později. Jakmile uživatel zadá potřebné údaje pro vyhledání, může kliknout na tlačítko odeslat. Aplikace se pokusí zadanou entitu vyhledat. Pokud entitu nalezne, zobrazí jí v tabulce níže. Stránka zobrazuje, v jakém díle se vazba nachází, jméno první entity, typ vazby a jméno druhé entity. Bližší informace o vazbách jsou v kapitole 7 Natural Language Understanding. Výsledky analýzy vazeb budou představeny v práci později.

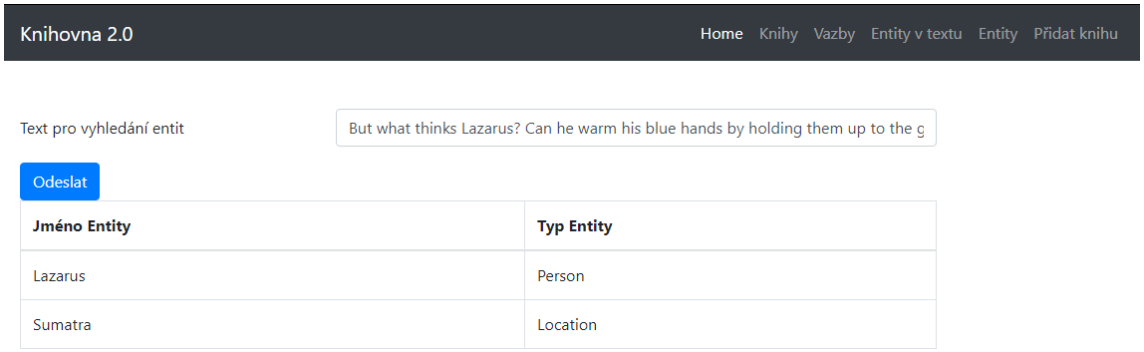


Obrázek 5 Interface: Vazby

Entity v textu

Další stránkou v menu je **entity v textu**. Tato stránka slouží uživatelům k vyhledávání entit v úryvcích textu. Délka textu pro analýzu entit sice není omezena počtem znaků, ale výsledky nejsou nijak ukládány, jsou pouze zobrazovány. Pokud uživatel přenačte stránku veškeré výsledky ztratí. Při zadání příliš dlouhého textu navíc uživatel bude muset prohledávat zbytečně mnoho výsledků. Po kliknutí na tlačítko „Odeslat“ aplikace zpracuje zdrojový text a zobrazí uživateli tabulku. V prvním sloupci uživatelé mohou vidět název entity. V druhém sloupci pak mohou vidět, jakého typu entita v daném textu je. Obrázek 6 Vyhledání entit v textu, ukazuje

vyhledání entit v textu *„But what thinks Lazarus? Can he warm his blue hands by holding them up to the grand northern lights? Would not Lazarus rather be in Sumatra than here?“*[14]. Tento text byl vybrán z knihy Moby Dick.

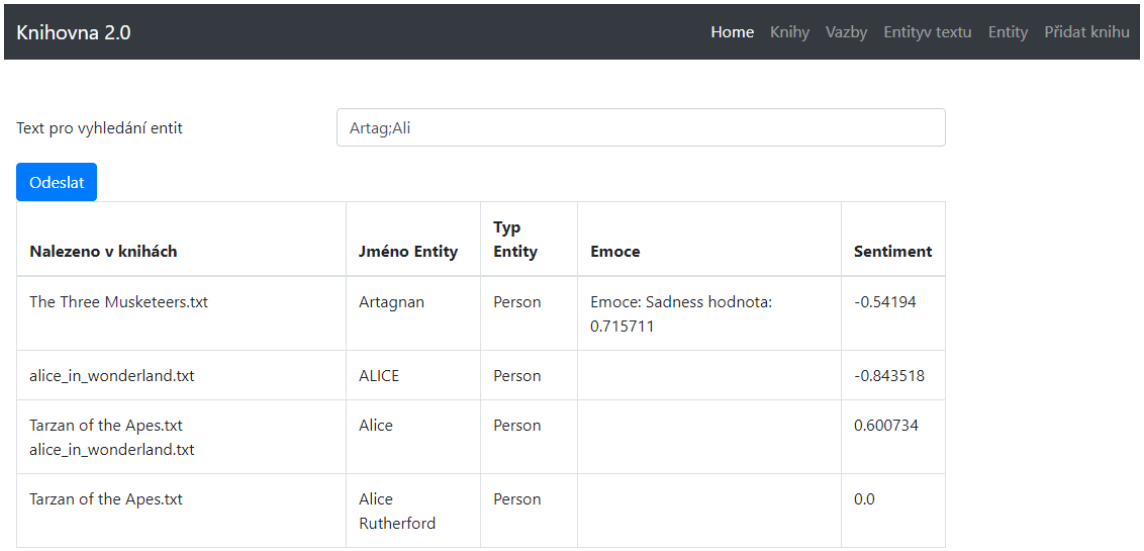


Obrázek 6 Vyhledání entit v textu

Entity

Stránka **entity** slouží k vyhledávání entit, které byly již aplikací nalezeny, z analyzovaných titulů. Uživatel zadá název hledané entity. Pokud uživatel nezná celé jméno entity, může napsat jen začátek. Aplikace vyhledává entity začínající vloženým textem. Uživatel může najednou vyhledávat více entit najednou. Pokud uživatel chce vyhledat více entit najednou, stačí jednotlivé entity oddělovat středníkem. Entity se i při hromadném zadání zpracovávají samostatně tudíž je i při hromadném zadání psát pouze začátky hledaných entit. Pokud hledanému výrazu odpovídá jedna nebo více entit, aplikace uživateli zobrazí tabulku, kde každý řádek odpovídá hledané entitě. Uživatel vidí, v jakém díle se entita nachází a jak se celá entita jmenuje. Pokud k entitě byla nalezena významná emoce, je zobrazena i ta. Na konci je zobrazen sentiment, ve kterém se entita v textu vyskytla. Všechny výše zmíněné hodnoty jsou více vysvětleny v kapitole 7 Natural Language Understanding. Bližší popis výsledků práce s těmito hodnotami je popsán v další části práce s názvem 8.4 Zpracování souboru.

[14] Herman Melville, Moby-Dick, on-line, https://www.gutenberg.org/files/2701/2701-0.txt



Obrázek 7 Vyhledání entit v aplikaci

Přidat knihu

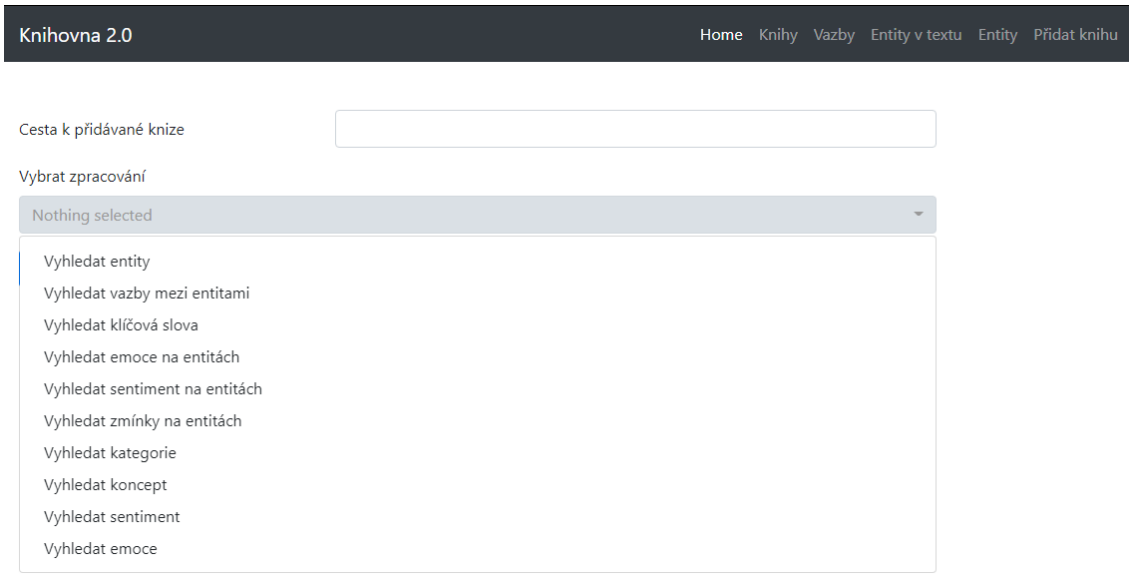
Poslední položkou menu stránek aplikace Knihovna 2.0 je **přidat knihu**. Tato stránka slouží uživatelům k přidávání nových titulů do aplikace. Uživatel aplikaci předává cestu, na které aplikace může nalézt uvedený titul. V době psaní této práce musí umístění titulu být na stejném souborovém systému, jako spuštěná aplikace. Uživatel také může zadat, jaké analýzy chce nad titulem provést. Níže budou popsány jednotlivé možnosti, uživatel může vybrat více možností zpracování najednou.

Druhy zpracování:

- Vyhledat entity – Vyhledání entit v titulu. Entity je následně možné vyhledávat na stránce Entity.
- Vyhledat emoce na entitách – Pokud budou vyhledávány entity, bude analyzováno, jaké emoce se na entitu váží.
- Vyhledat sentiment na entitách – Pokud budou vyhledávány entity, zanalyzuje se i sentimentální zabarvení.
- Vyhledat zmínky na entitách – V době psaní práce nebylo plně implementováno. Pokud by bylo vybráno, u entit by byla uchovávána informace o jejich přesném umístění v textu.

- Vyhledat vazby mezi entitami – Vyhledá vazby mezi entitami. Nalezené vazby je možné zobrazit na stránce vazby.
- Vyhledat kategorie – Budou vyhledány kategorie, do kterých je možné titul zařadit.
- Vyhledat koncept – Pro zpracovávaný titul budou vyhledány koncepty.
- Vyhledat sentiment – Bude zanalyzován sentiment celého titulu.
- Vyhledat emoce – Budou zpracovány celkové emoční hodnoty daného titulu.

Bližší vysvětlení jednotlivých druhů zpracování je možné nalézt v kapitole 7 Natural Language Understanding.



Obrázek 8 Přidání knihy

8.2.2 Model

Modelová vrstva aplikace Knihovna 2.0 v sobě zapouzdřuje data, se kterými aplikace pracuje. Zároveň do této vrstvy patří logika pro práci s úložištěm dat, v případě Knihovny 2.0 se jedná o databázi SQLite.

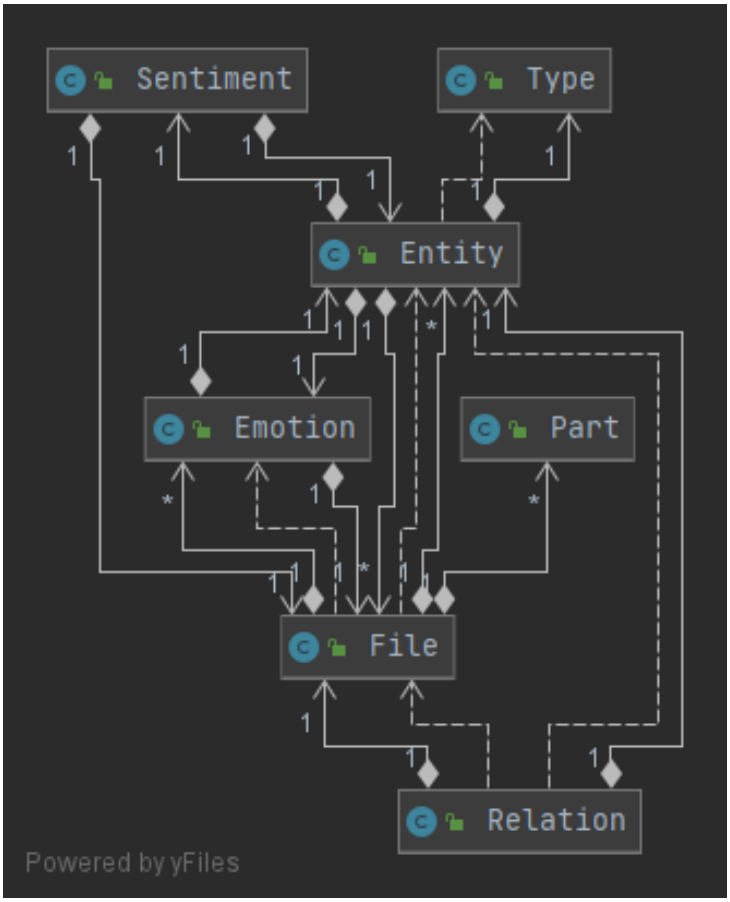
Aplikace Knihovna 2.0 využívá pro práci s databází funkcionalitu frameworku Spring, projekt Spring Data JPA. Díky tomuto projektu není nutné v aplikaci vyvíjet vlastní základní databázové operace. Jednotlivé objekty jsou

pomocí anotací namapovány na entity[15]. Aplikace Knihovna 2.0 dodržuje jmenné konvence názvů metod u tříd obstarávajících práci s databází. Toto jmenné schéma umožňuje Spring Data JPA porozumět, co daná metoda má z databáze získat a následně ORM[16] Hibernate pomocí vlastní funkcionality vytvoří SQL dotazy. Synergie tohoto všeho, usnadňuje práci při vytváření a úpravách datové vrstvy aplikace, bez nutnosti psaní vlastních SQL dotazů. Pokud by pro naplnění potřeb uživatelů nebylo možné tímto způsobem získat některá data z databáze, nebo by byla aplikace rozšířena o jiné úložiště dat, je možné přidat vlastní SQL dotazy, popřípadě přetížit [17]již stávající logiku.

8.2.2.1 Class model

Class model představuje objekty, tak jak je vidí aplikace Knihovna 2.0. Tudíž z pohledu objektově orientovaného. Tento pohled je nastaven tak, aby co nevíce odpovídal potřebám aplikace pro práci s daty. Nemusí však odpovídat tomu, jak jsou data uložena v databázi.

[15] Třída anotována javax.persistence.Entity, Toto anotování je nutné pro použití ORM.
[16] Objektově relační mapování, technika automatického převodu dat z tabulek do objektů.
[17] Technika objektového programování, kdy potomek třídy může vložit vlastní zpracování metody svého předka.



Obrázek 9: Entitní diagram

Z přiloženého obrázku Obrázek 9: Entitní diagram je patrné, že středobodem je třída File. Jedná se o třídu představující konkrétní titul a všechny ostatní informace jsou na ní navázaná. Bližší seznáme se třídou je níže v textu.

Entity

Třída entity představuje slova, která jsou NLP vyhodnocena jako entity. Tudíž slova významná. Entity jsou klíčovou částí modelu aplikace. Většina ostatních tříd modelu je navázána právě na třídu entity.

Entity se skládá z vlastností:

- **Id** unikátní identifikátor dané entity,
- **Name** text dané entity, v jejím kořenovém tvaru,
- **Type** odkaz na jiný objekt, představuje jakého typu entita je,

- **Files** odkaz na Set jiných objektů, představuje, v jakých souborech se entita vyskytuje.

File

Třída představuje soubory, ze kterých aplikace získává text. Obsah těchto souborů je následně zpracován NLP.

- **Id** unikátní identifikátor daného souboru.
- **Path** cesta k danému souboru na daném souborovém systému.
- **ProcessComplete** Číselná hodnota 0 nebo 1, který představuje, jestli byl daný soubor již celý zpracován NLP.
- **Entities** odkaz na Set[18] jiných objektů, který představuje, jaké entity se v souboru vyskytují.
- **KeyWords** odkaz na Set jiných objektů, který představuje, jaká klíčová slova se v souboru vyskytují.

Relation

Třída relation představuje vazby mezi entitami. Tato vazba je vždy mezi dvěma entitami, kdy vazba má určitý směr, a vždy jde od jedné entity k druhé. Například: Thomas Watson founder of IBM.

- **Id** unikátní identifikátor daného souboru.
- **Type** text, vyjadřující jaký je mezi entitami vztah.
- **EntityFrom** odkaz na objekt, představující entitu, z které vazba vychází.
- **EntityTo** odkaz na objekt, představující entitu, ke které vazba směřuje.

Type

Třída type, představuje, jakého typu entita je.

- **Id** unikátní identifikátor daněného typu.

[18] Množina jedinečných objektů.

- **Name** text, se slovním vyjádřeným daného typu.

Keyword

Keyword reprezentuje třídu klíčových slov, objevujících se v textu daného souboru.

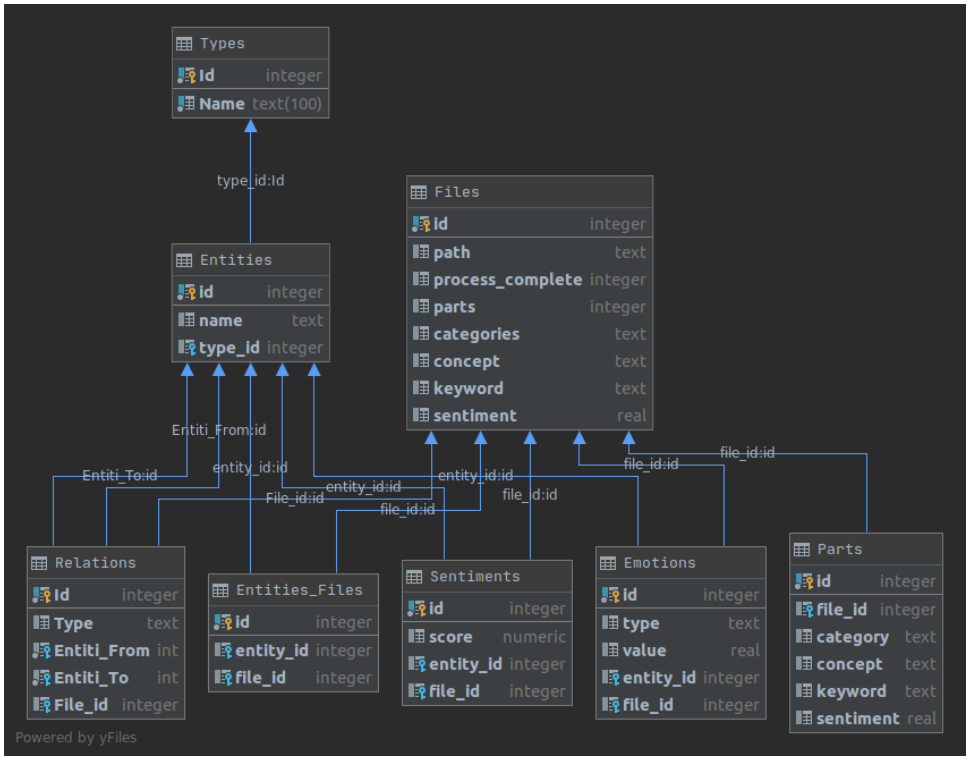
- **Id** unikátní identifikátor daněného keyword.
- **Text** je hodnota daného keywordu.
- **Files** odkaz na Set jiných objektů, představuje, v jakých souborech se keyword vyskytuje.

PostContent

Tato třída je využívána při komunikaci mezi vrstvami view a controler. Jakmile uživatel zadá požadavek na zobrazení stránky, http request typu GET, aplikace mu kromě html stránky vrátí také instanci této třídy. Následně, když uživatel zašle svůj požadavek zpět do aplikace, http request typu POST, instance této třídy je poslána zpět s již obsaženými vstupy. Tento přístup usnadňuje následné získávání dat z http requestů.

8.2.2.2 Databázový model

Aplikace Knihovna 2.0 uchovává svá data do databáze. Obrázek 10: Databázový diagram, znázorňuje databázové schéma pro ukládání těchto dat.



Obrázek 10: Databázový diagram

Entities

Tabulka obsahující všechny entity známé aplikaci.

- **Id** číslo, unikátní identifikátor daného entity.
- **Name** text, textová podoba dané entity.
- **Type_id** číslo, cizí klíč do tabulky types.

Files

Tabulka obsahující všechny soubory, nahrané do aplikace.

- **Id** číslo, unikátní identifikátor daného souboru.
- **Path** text, cesta k danému souboru.
- **ProcessComplete** číslo, hodnota jestli je soubor již zpracován.

Entities_Files

Tabulka představující vazbu n: n mezi tabulkami Entitis a Files. Vazba je přes cizí klíče, které drží právě tabulka Entities_Files. Díky tomu nejsou samotné tabulky Entities a Files zatíženy touto informací. Dále tento přístup zajišťuje,

aby se jedna entita objevila ve více souborech a naopak, aby jeden soubor mohl obsahovat více entit.

- **Id** číslo, unikátní identifikátor dané vazby.
- **Entity_Id** číslo, identifikátor dané entity.
- **File_Id** číslo, identifikátor daného souboru.

Types

Tabulka obsahující všechny typy entit.

- **Id** číslo, unikátní identifikátor daného typu.
- **Name** text, textová podoba daného typu.

Relations

Tabulka uchovávající vazby mezi entitami.

- **Id** číslo, unikátní identifikátor dané vazby.
- **Type** text, jméno dané vazby.
- **Entity_From** číslo, identifikátor dané entity.
- **Entity_To** číslo, identifikátor dané entity.

Sqlite_master

Řídící tabulka databáze SQLite. Aplikace Knihovna 2.0 s touto tabulkou nepracuje.

Sqlite_sequence

Tabulka databáze SQLite pro uchovávání automaticky generovaných hodnot, aplikace s touto tabulkou přímo nepracuje.

8.2.3 Controller

Tato část aplikace je utvářena procesní logikou požadavků na aplikaci. Kdy je převzat požadavek z view, v podobě http requestu. Z něj se následní zjistí,

jaký typ požadavku chce uživatel vykonat a jaké k tomu poskytl vstupy. Výsledky zjištění se následně ještě zpracovávají. Postup tohoto zpracování je popsán níže.

8.2.3.1 Servlety

Servlety[19] jsou první z části Controler, které zpracovávají požadavek od uživatele. Jedná se o třídu, jejíž metody jsou namapovány, aby zpracovávaly požadavky zaslané na určité adresy. Například http://localhost:8080/ser/entities bude zpracována jinou metodou než http://localhost:8080/ser/rel. Daná metoda pomocí services, které budou popsány v následujícím textu, vykoná požadavek od uživatele. Potom co servlet tento požadavek vykoná, předá výstupní data zpět vrstvě view. Vrstva view následně odpověď zobrazí uživateli ve formě html stránky.

8.2.3.2 Services

Services jsou třídy zajišťující určitou specifickou funkcionalitu. Tyto třídy jsou vytvářeny a předávány pomocí dependenci injection[20] do jednotlivých servletů, popřípadě jiných services podle potřeby. Jejich správu zajištuje framework Spring.

WatsonService

Třída, jejímž účelem je komunikace se službami zajištující NLP. Tudíž se servery IBM cloud. Tato třída pomocí Watson API sestavuje http requesty, které posílá službě Nature Language Processing. Potom co služba Nature Language Processing vrátí odpověď v podobě http response, tato třída odpověď přečte a výsledky zpracuje. Výsledky se vrací ve formátu JSON[21]. Watson API tento JSON interně transformuje do Java tříd pro snadnější práci. Zpracováním se nejčastěji myslí uložení vrácených dat do databáze. Na toto uložení WatsonService používá třídy představené v předešlé kapitole Model. Třída krom samotného ukládání

[19] Java třída implementující rozhraní javax.servlet. Servlet, pro zpracování requestů a responsů
[20] Technika správy vytváření tříd v kontextu aplikace.
[21] Standard pro formátování textu.

filtruje, co se má uložit, aby nevznikaly duplicity. Při nalezení nové entity či vztahu, zajišťuje vytvoření vazeb se stávajícími entitami a soubory.

WatsonService pro komunikaci se službami IBM potřebuje mít určitou konfiguraci. Tato konfigurace se skládá z:

- **apiKey** jedná se o hash[22] z hesla k instanci služby zajišťující NLP na straně IBM cloudu,
- **serviceUrl** URL adresa, na které poslouchá instance služby NLP,
- **modelId** identifikátor modelu, kterým se řídí služba NLP.

Aplikace by v budoucnu mohla mít více konfigurovatelných parametrů modelId. Díky specifičnosti každého modelu, by bylo možné hledat jiné informace v textu, v závislosti na použitém modelu.

LoadDataFrom

Metoda na zpracování souboru s textem skrz NLP. Metoda má na vstupu adresu, kde leží soubor s textem. Dále Set enumů[23], které představují, co se v textu má vyhledávat. Tento soubor musí obsahovat text v kódování UTF-8[24]. Obsah souboru je následně načten do paměti. Metoda vytvoří záznam o tomto souboru v DB.

Metoda vytvoří pomocné třídy z Watson API, obsahující konfiguraci, co se s textem má stát. Tudíž jestli se má text zpracovat pouze na vyhledání entit, či se na něj mají aplikovat další algoritmy.

CutFileToParts

Velikost zdrojového textu pro analyzování může překračovat únosnou mez. Též existují knihy, které se skládají ze samostatných příběhových částí. Tyto části však na sebe nemusejí nijak navazovat a jejich společná analýza by byla naopak pro výsledek škodlivá. Tyto všechny faktory se mohou dynamicky měnit. Pro možnosti

[22] Technika převedení libovolného množství dat na jedinečný řetězec o předem dané délce.
[23] Java třída představující konstanty.
[24] Způsob kódování znaků v souborech.

upravování, jak velké části textu budou najednou odesílány k analyzování, byla vytvořena metoda cutFileToParts. Tato metoda má za úkol rozdělit zdrojový text do několika částí, které budou zpracovávány samostatně. Základní implementace metody pracuje na principu rozdělení textu do částí podle počtu slov. Počet slov v jedné části je definován parametrem v souboru project.properties pod klíčem sizeOfPart. Tato metoda může být však přetížena podle aktuálních potřeb a text může být rozdělován jiným způsobem. Například by mohl být text rozdělován podle klíčových slov *kapitola*, či jiným způsobem odpovídajícím dané situaci.

ProcessTextOnEntities

Tato metoda zpracovává segment entities. Metoda prochází jednotlivé nalezené entity. U každé entity je nejdříve zjištěn její typ. Pokud takový typ v databázi aplikace ještě není, je následně založen. Následně je porovnáno, jestli takováto entita již existuje. Pokud ano, je načtena již existující entita. V případě, že ne, aplikace entitu vytvoří. Vytvořená či získaná entita je následně spojena se souborem, z kterého byl text získán. Pokud uživatel zvolil hlubší analýzu entit, budou provedeny metody processEntityOnEmotion a processEntityOnSentiment.

ProcessEntityOnEmotion / ProcessEntityOnSentiment

Tyto dvě metody fungují na stejném principu. Zjistí z odpovědi od service NLP, jestli se na dané entitě vyskytuje jimi hledaná vlastnost. Pokud ano, je tato vlastnost v hodnotách, které jsou pro aplikaci zajímavé. V případě, že jsou všechny podmínky splněny, bude vytvořena vazba mezi entitou a souborem, ve kterém se entita vyskytla. Tato vazba představuje buď emoční zabarvení, nebo sentiment.

ProcessTextOnRelations

Metoda zpracovává část odpovědi relations. Metoda prochází jednotlivé vazby, které jsou metodě předány přes objekt z Watson API. Metoda nejdříve načte entitu, ze které vazba vychází. Pokud taková entita není v databázi, agent jí vytvoří. Toto agent udělá i s druhou entitou, do které vazba vede. Metoda následně z objektu Watson API získá typ vazby. Toto vše, spolu s odkazem na soubor, v kterém se vazba nachází, uloží do databáze.

SaveFileToDb

Jedná se o metodu na uložení informací o souboru do databáze. Metoda do databáze uloží umístění souboru na disku, spolu s informací, jestli je již soubor zpracován.

8.2.3.3 Server-Side script

Aplikace Knihovna 2.0 využívá pro dynamické vytváření obsahu, pro uživatele, knihovnu Thymeleaf.

Thymeleaf je knihovna založená na Javě, pro šablonování webového prostředí na straně serveru. Hlavním účelem Thymeleaf je přinést elegantní řešení pro tvorbu HTML[25] stránek. Thymeleaf se hodí pro tvorbu webů, které slouží jako statické prototypy, pro lepší spolupráci ve vývojovém týmu. (24)

Knihovna Thymeleaf se nachází mezi vrstvou controler a vrstvou view. Jejím účelem je vsadit data předávaná controlerem do předpřipravených HTML stránek. Nejčastějším příkladem využití Thymeleaf je převedené seznamu Java objektů na HTML elementy.

8.3 Příprava dat

Data jsou do aplikace Knihovna 2.0 předávána pomocí odkazů s umístěním souboru na disku. Tento soubor však musí splňovat určitá kritéria, aby mohl být aplikací zpracován:

- text v souborech musí být ve formátu UTF-8,
- v souborech se musí nacházet pouze text. Nesmí se v nich nacházet obrázky, ani žádné speciální znaky pro formátování.

[25] Hypertext Markup Language, značkovací jazyk pro webové stránky.

8.4 Zpracování souboru

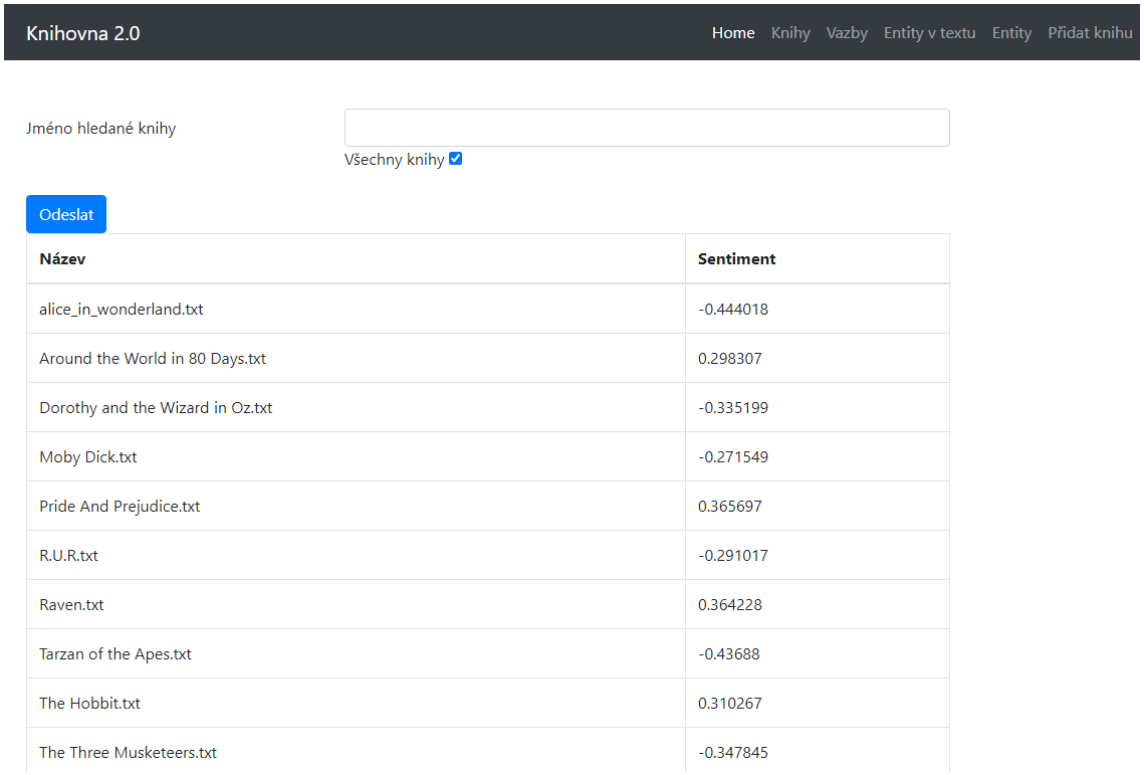
V následující části bude představeno zpracování knih pomocí aplikace Knihovna 2.0.

Pro ukázku byly vybrány tyto knihy:

- Alice in Wonderland
- Around the World in 80 Days
- Dorothy and the Wizard in Oz
- Moby Dick
- Pride And Prejudice
- R.U.R.
- Tarzan of the Apes
- Raven
- The Hobbit
- The Three Musketeers

Výsledky analýzy sentimentu

Sentiment dané knihy je zobrazován v tabulce s ostatními informacemi, které se váží k danému titulu. V tabulce níže jsou sepsány hodnoty analyzované napříč knihami. Z tabulky byly odstraněny ostatní zobrazované hodnoty pro lepší přehlednost.

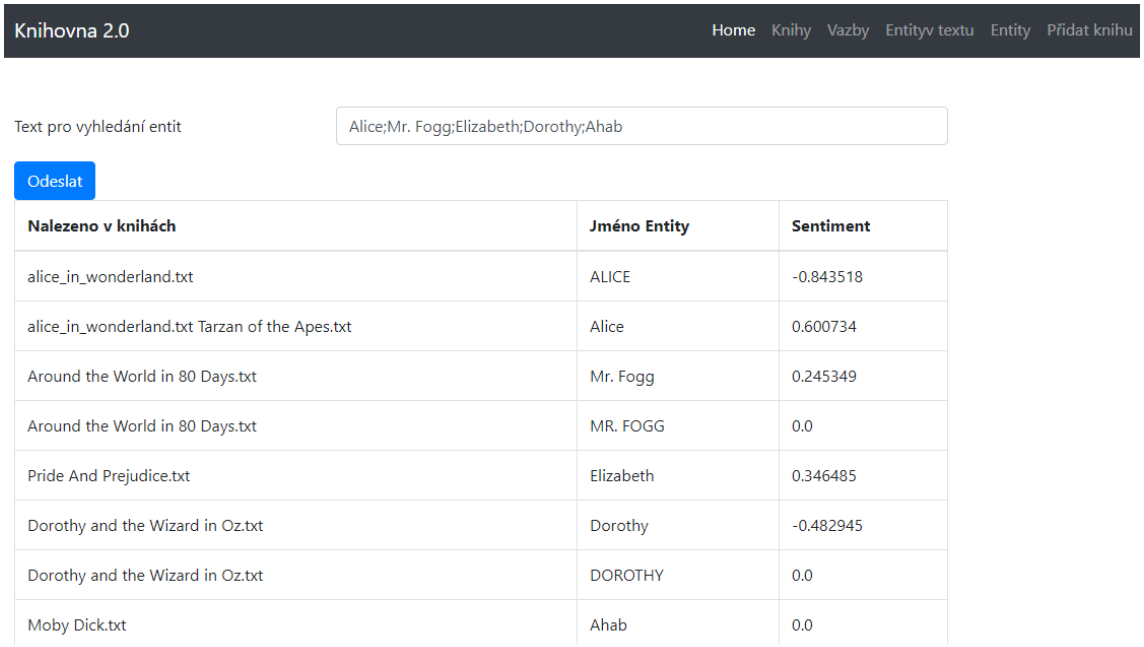


Obrázek 11 Sentiment na knihách

Díky těmto hodnotám je možné snadno zjistit sentimentální úroveň daných titulů. Hodnota sentimentu daného díla se zobrazuje uživateli v části **knihy**. Význam hodnot je popsán v kapitole Sentiment.

Sentiment se určuje také ještě pro dané části, avšak současná implementace nerozděluje texty do částí. Tudíž hodnoty pro dané části jsou stejné, jako pro titul samotný. Aplikace je však připravena se sentimentem v částech pracovat. Pokud by byl titul rozdělen do více částí, sentimentem celého titulu by byl průměr ze sentimentů částí.

V tabulce níže bude představen sentiment na jednotlivých entitách. Hodnoty sentimentu se zobrazují při vyhledání entit podle názvu na stránce **entity**. Z tabulky byly odstraněny ostatní zobrazované hodnoty pro lepší přehlednost.

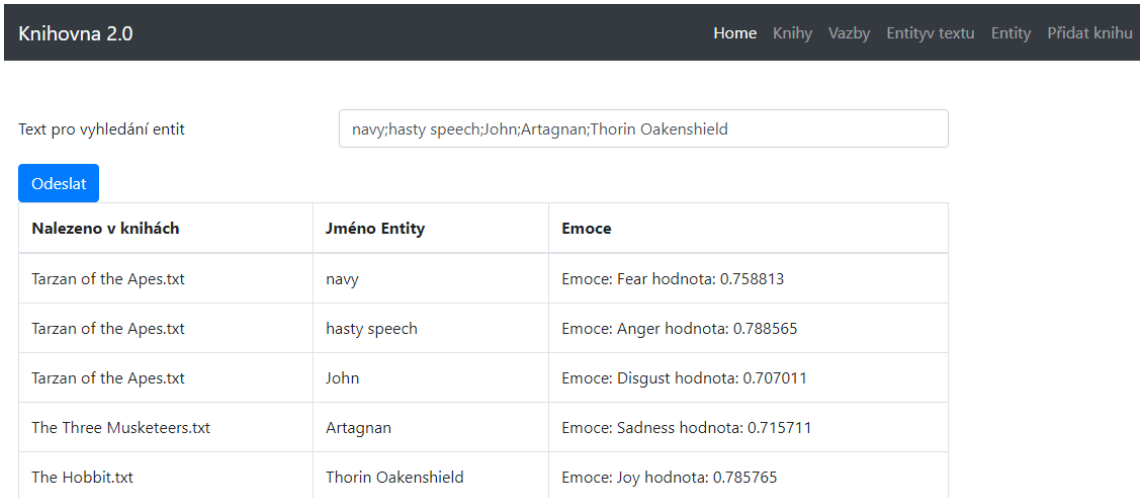


Obrázek 12 Analýza sentimentu na entitách

Tato tabulka představuje sentiment jednotlivých entit v daných textech. V tabulce je pouze výňatek ze všech nalezených entit pro ukázku. Díky tomuto je možné hledat danou entitu v textech, kde vystupuje v určitém sentimentálním zabarvení. Entita se v textu může vyskytnout vícekrát s různými hodnotami sentimentu. Bližší seznámení s významem sentimentálního scóre je v kapitole Sentiment.

Výsledky analýzy emocí

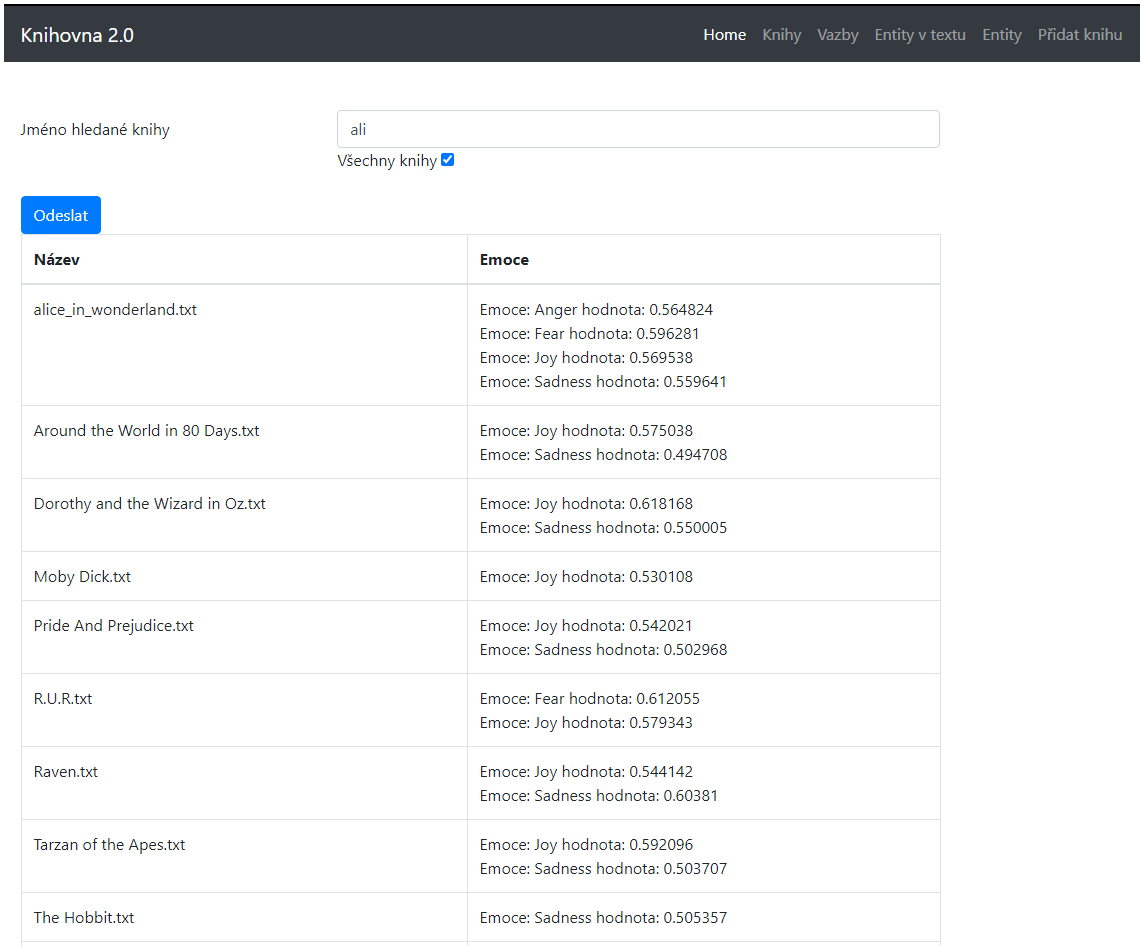
Při analýze emocí bylo do aplikace nastaveno kritérium, kdy se zaznamenávají pouze emoční zabarvení přesahující určitou hodnotu. Pro Entity byla tato hodnota nastavena na 0,7. Toto kritérium je možné nastavit také pro tituly jako celek. Toto kritérium bylo takto nastaveno z důvodu zaznamenávání pouze významných emočních zabarvení. Jakákoli nastavení týkající se tohoto zabarvení, jsou však měnitelná v konfiguračním souboru.



Obrázek 13 Analýza emocí na entitách

Tabulka výše ukazuje výsledky analýzy emocí spojených s určitou entitou v daném textu. Díky těmto datům je možné vyčíst, že v titulku Tarzan of the Apes se vyskytuje silný strach z námořnictva. Díky těmto vazbám se uživatelům otvírá možnost vyhledávat tituly, které mají určitý názor na dané téma. Model by dokázal nalézt pro dané entity i jiné emoce. Avšak bylo záměrně určeno vyšší kritérium pro zaznamenávání entit, aby byly zaznamenávány pouze extrémy. V reálném nasazení by bylo toto kritérium nejspíše nižší a uživateli by byla dána možnost určit, jaké emoční skóre je pro něj zajímavé. Tyto hodnoty uživatelé mohou nalézt na stránce **Entity.** Zbylé hodnoty zobrazované na entitách byly odstraněny pro lepší čitelnost.

Výsledky analýzy emocí se též připisují ke knihám samotným. Proto je možné vidět na stránce pro vyhledávání knih i emoce, které byly nalezeny v dané knize jako celku. Pro názornost je uvedena ukázka v obrázku níže.



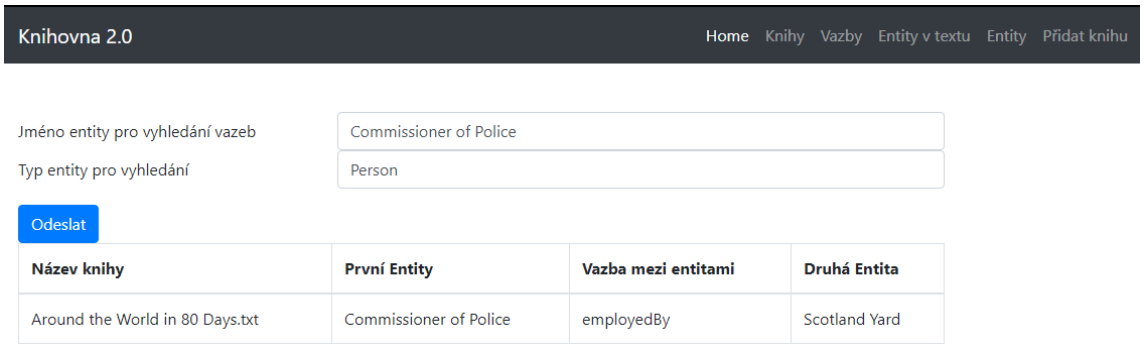
Obrázek 14 Analýza emocí pro knihy

Stejně jako v analýze emocí u entit, tak i u knih, je možné určit minimální hodnotu pro dané emoce v konfiguračním souboru. Vyhledávání emocí nad celkovým textem se však ukázalo jako velice zavádějící. Emoce **Joy** v rozmezí 5 až 6 model nalezl v devíti z deseti knih. Tím, že byla tato emoce nalezena v devadesáti procentech analyzovaných textů, navíc s takřka stejnou hodnotou, znamená, že pro účely vyhledávání a filtrování je takřka nepoužitelná. Ostatní emoce se již vyskytují méně a přináší tedy onu přidanou hodnotu pro filtrování.

Výsledky analýzy vazeb.

Zobrazování vazeb mezi entitami je významnou částí analýzy textu. Tato část uživateli poskytuje možnost vidět, s jakými entitami jeho hledaná entita

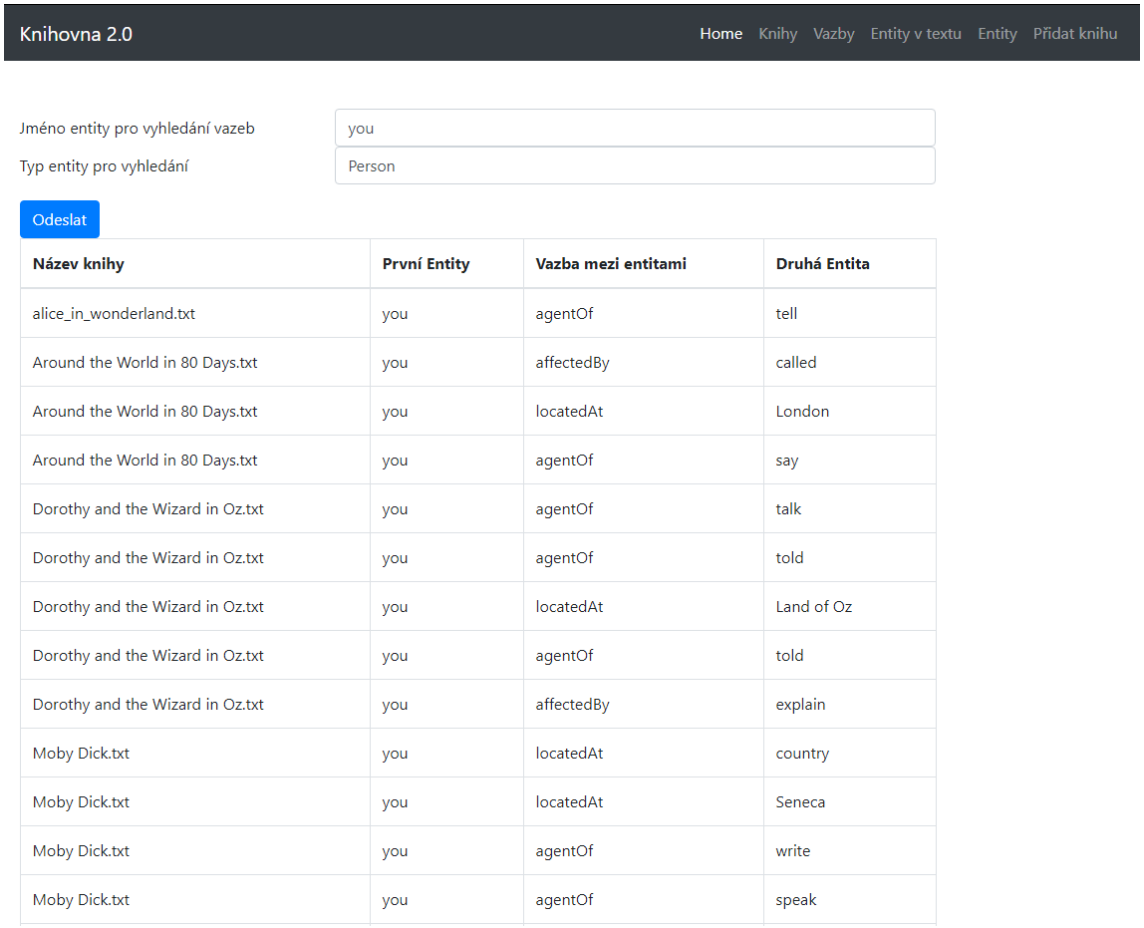
interaguje. Tyto informace se uživateli zobrazují v tabulce spolu s knihou, ve které se daná vazba nachází.



Obrázek 15 Analýza vazeb

V tabulce výše je nastíněna ukázka možnosti vyhledávat určité, pro uživatele zajímavé, situace v daných titulech. Množství druhů vazeb, které aplikace může v textech nalézt, je omezen modelem, podle kterého je text analyzován.

Čím je však model specifičtější a přesnější, tím však roste i míra vazeb, které jsou v podstatě bezvýznamné a nic neříkající. Například:



Obrázek 16 Analýza vazeb potřeba koreference

Jak je vidět, tyto vazby zobrazené na obrázku výše, bohužel samy o sobě žádné informace nepřináší. Jejich informační hodnota je pro nás velmi nízká. Vazba nám říká pouze to, že někdo něco dělá, například mluví, píše, nebo že někdo něco řekl, ale to je vše. Více informací, které by nám mohly pomoci více rozklíčovat text a díky tomu i cíleně v textu vyhledávat, touto vazbou nezískáme.

Toto však není chyba modelu, jen jeho současné omezení. Řešením v budoucnu bude takzvaná koreference, tedy že model bude schopný určit, že text referuje o jedné entitě pomocí vícero různých slovních zmínek napříč větami či celým dílem. V době psaní této diplomové práce byla koreference ve Watson NLP pouze v experimentální fázi, proto jí model neobsahoval.

9 Shrnutí výsledků

Použití mechanik pro zpracování nestrukturovaného textu se ukázalo jako možné, Aplikace IBM Watson umožňuje důkladně analyzovat a vytěžovat nestrukturované texty, což přináší nové možnosti při zpracování jakéhokoliv textu. V této práci byly tyto možnosti ukázány na problematice vyhledávání knih dle jejich obsahu.

Při psaní této práce se vyskytlo mnoho problémů, které by si zasloužily vlastní zpracování či dořešení. Mezi největší problémy, na které jsem během psaní této práce narazil, jsou problematika modelu a koreference entit.

Problém modelu

Původním záměrem této práce bylo vytvořit a použít vlastní aplikaci Knihovna 2.0, která měla dle původního záměru využívat vlastní doménový model na analýzu textu. Aplikace byla úspěšně vytvořena, avšak vytvořit vlastní model pro zpracování textu se nepodařilo.

Jak již bylo v této práci zmíněno – složitost modelu se při tvorbě ukázala jako nepřekonatelná překážka. Hlavním důvodem bylo nenalezení společného vzoru sdíleného napříč knihami. Vzhledem ke zmíněné složitosti bylo rozhodnuto, vlastní model pro potřeby této práce nevyužít a bylo využito obecného modelu IBM.

Z analýz uskutečněných v této práci vyplývá, že využití obecného modelu IBM pro analýzu knih je možné, avšak model by bylo dobré obohatit o další vazby, z kterých by bylo možné více vyčíst děj v knihách. Vytvoření takového modelu by však znamenalo vytvoření velice detailního a vše účelného modelu pro zpracování lidské řeči. Takový model by však mohl být náchylný k zaznamenávání pro danou knihu nedůležitých vazeb. Řešením by mohlo být nasazení vícero modelů a více průchodná analýza. Tento proces by se mohl zakládat z využití obecného modelu pro analýzu základních prvků textu. Následně by byl proveden další průchod s již jiným modelem zaměřeným na více specifickou doménu. Tento proces by se mohl opakovat vždy s více doménově specifickým modelem pro dosažení maximální výtěžnosti textu.

Koreference

Velkým problémem ve shromažďování dat z titulů není to, že by již stávající obecný model od IBM nebyl dostatečně přesný, nebo že by nedokázal vytěžit dostatek informací z textu. Problém je v přiřazování nových informací, k již zjištěným. Služba IBM Natural Language Understanding poskytuje hlubokou analýzu textu, jak bylo ukázáno pomocí aplikace Knihovna 2.0. Avšak zatím nedostatečně řeší koreferenci entit napříč zadaným textem, nijak než slovníkovým označením v modelu. Slovníková koreference totiž může ošetřit pouze již předem známé pojmy. Avšak při přidávání nových, nebo zatím neznámých titulů, není vhodným řešením upravovat vždy slovník a model.

Dalším problémem z této kategorie, za který IBM Watson nemůže, je koreference napříč texty a tituly. Vícedílné knihy nelze zadávat najednou, už z principu postupného tvoření oněch děl. Vytvoření postupu pro navazování entit napříč texty se ukázal jako nemalá výzva v oblasti NLP a zasloužil by si důkladnější analýzu.

10 Závěr

Cílem této diplomové práce bylo představení možného využití Aplikace IBM Watson při zpracování nestrukturovaného textu a následného využití takto získaných informací pro vyhledávání v prostředí knihovních archívů. Aplikace IBM Watson umožňuje důkladně analyzovat a vytěžovat nestrukturované texty. Přínosem této práce jsou konkrétní výstupy jednotlivých analýz, na základě, kterých bylo prokázáno, že i obecný, veřejně dostupný model od IBM, dokáže dostatečně zpracovat i rozsáhlejší texty a vyčíst z nich potřebné informace.

Zároveň se však pro zpracování rozsáhlých nestrukturovaných textů prokázala nutnost mít na dané texty specifiky doménový model, protože model použitý pro potřeby této práce, je určený spíše na zpracovávání dokumentů menšího rozsahu.

V případě, že bychom pro dané analýzy použili textu více doménově odpovídající modely, bylo by dosaženo přesnějších výsledků.

Již dnes se využití strojového zpracování strukturovaných textů uplatňuje v různých odvětvích, od bankovnictví a pojišťovnictví počínaje, až po stavebnictví, zdravotnictví, vzdělávání a mnoho dalších odvětví konče. Díky tomu došlo v posledních letech k ohromným úsporám lidské práce, která je považovaná většinou firem za nejdražší a nejnákladnější položku. Pokud by se podařilo zopakovat tento úspěch i u nestrukturovaných dat, přínosy pro firmy a státní správu v podobě další úspory lidské práce, by byly několikanásobně větší než v případě zpracování strukturovaného textu.

Jak bylo již zmíněno v úvodních kapitolách této práce, Phillip Russom ve své publikaci uvádí, že poměr mezi strukturovaným a nestrukturovaným textem je cca 1:3 (5).

Zpracování nestrukturovaného textu je finálním a jednoznačně nejdůležitějším krokem ke kompletnímu dokončení strojového zpracování textu. Považuji ho za budoucnost a nutnost, která pomůže se zefektivněním práce v mnoha oblastech.

11 Zdroje

1. **Apache UIMA Documentation Team.** Apache UIMA - Apache UIMA. *Apache UIMA - Apache UIMA.* [Online] [Citace: 21. Prosinec 2018.] https://uima.apache.org/index.html.

2. **Gartner.** www.gartner.com. *What Is Big Data? - Gartner IT Glossary - Big Data.* [Online] [Citace: 26. Prosinec 2018.] https://www.gartner.com/it-glossary/big-data/.

3. **Oracle.** What is Big Data? | Oracle. *Oracle.* [Online] [Citace: 26. Prosinec 2018.] https://www.oracle.com/big-data/guide/what-is-big-data.html.

4. **holubová, Irena, a další.** *Big data a NoSql databáze.* Praha : Grada, 2015. ISBN 978-80-247-5466-6.

5. **Phillip, Russom.** BI Search and Text Analytics. [Online] 2007. [Citace: 7. 11 2020.] http://download.101com.com/pub/tdwi/Files/TDWI_RRQ207_lo.pdf.

6. **SAS.** What Is Big Data? | SAS US. *sas.com.* [Online] [Citace: 26. Prosinec 2018.] https://www.sas.com/en_us/insights/big-data/what-is-big-data.html.

7. **Stančík, Martin.** Big Data bez správné analýzy nejsou k ničemu. *computerworld.* [Online] 27. 8 2013. [Citace: 28. 2 2019.] http://computerworld.cz/technologie/big-data-bez-spravne-analyzy-nejsou-k-nicemu-50285.

8. **Ministerstvo vnitra České Republiky.** https://www.mvcr.cz. *Ministerstvo vnitra České Republiky.* [Online] [Citace: 3. 1 2019.] https://www.mvcr.cz/clanek/informace-podle-zakona-106.aspx.

9. **Canuma, Prince.** The brief history of NLP. *medium.* [Online] 29. 8 2019. [Citace: 18. 10 2020.] https://medium.com/datadriveninvestor/the-brief-history-of-nlp-c90f331b6ad7.

10. **Foote, Keith D.** A Brief History of Natural Language Processing. *dataversity.* [Online] 22. 5 2019. [Citace: 18. 10 2020.] https://www.dataversity.net/a-brief-history-of-natural-language-processing-nlp/#.

11. **Cvrček, Václav.** Regulární výrazy. *wiki.korpus.cz.* [Online] 21. Leden 2015. [Citace: 7. Červenes 219.] https://wiki.korpus.cz/doku.php/pojmy:regularni_vyrazy.

12. **Kleene, Stephen C.** www.rand.org. *www.rand.org.* [Online] 15. December 1951. [Citace: 7. Červenec 2019.] https://www.rand.org/content/dam/rand/pubs/research_memoranda/2008/RM704.pdf.

13. **Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman.** *Compilers, principles, techniques, and tools.* místo neznámé : Addison-Wesley Publishing Company, 1987. ISBN 0-201-10088-6..

14. **M.F.Porter.** An algorithm for suffix stripping. *Program.* 14, 1980, 3.

15. **Daniel Jurafsky, James H. Martin.** *Speech and Language Processing.* New Jersey 07632 : Prentice Hall, Englewood Cliff, 1999. ISBN 0-13-504196-1..

16. **analyticsvidhya.** The Essential NLP Guide for data scientists. *analyticsvidhya.* [Online] září. 26 2017. [Citace: 5. Srpen 2019.] https://www.analyticsvidhya.com/blog/2017/10/essential-nlp-guide-data-scientists-top-10-nlp-tasks/.

17. **Boyd, Tanmay Sinha and Andrew.** IBM Watson Knowledge Studio – Teach Watson about your domain. *www.ibm.com.* [Online] IBM, 27. leden 2016. [Citace: 9. Únor 2020.] https://www.ibm.com/blogs/watson/2016/06/alchemy-knowledge-studio/.

18. **IBM.** Watson Knowledge Studio. *cloud.ibm.com.* [Online] 26. 11 2019. [Citace: 15. 3 2020.] https://cloud.ibm.com/docs/services/watson-knowledge-studio.

19. —. *Building Cognitive Applications with IBM Watson Services: Volume 7 Natural Language Understanding.* místo neznámé : An IBM Redbooks publication, 2017. ISBN-10: 0738442623.

20. —. natural-language-understanding. *cloud.ibm.com.* [Online] IBM, 3. 9 2020. [Citace: 29. 10 2020.] https://cloud.ibm.com/docs/natural-language-understanding?topic=natural-language-understanding-about.

21. **Spring.** spring.io. *spring-framework.* [Online] Privotal, 2019. [Citace: 2. 12 2019.] https://spring.io/projects/spring-framework.

22. **The SQLite Consortium.** About SQLite. *www.sqlite.org.* [Online] The SQLite Consortium. [Citace: 7. 12 2019.]

23. **W3cschool.in .** What is MVC Architecture? *www.w3schools.in.* [Online] W3cschool.in . [Citace: 2019. 12 29.] https://www.w3schools.in/mvc-architecture/.

24. **The Thymeleaf Team.** Thymeleaf. *Thymeleaf.* [Online] Thymeleaf. [Citace: 16. Leden 2020.] https://www.thymeleaf.org/.

25. **Manning, Christopher D., Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky.** The Stanford CoreNLP Natural Language Processing Toolkit. *nlp.stanford.edu.* [Online] 2014. [Citace: 3. 8 2019.] https://nlp.stanford.edu/pubs/StanfordCoreNlp2014.pdf.

26. **Stanford coreNLP.** Stanford CoreNLP – Natural language software | Stanford CoreNLP. *stanfordnlp.github.io.* [Online] [Citace: 4. Srpen 2019.] https://stanfordnlp.github.io/CoreNLP/index.html.

27. **Microsoft.** file-path-formats. *docs.microsoft.com.* [Online] Microsoft, 6. 6 2019. [Citace: 2019. 12 27.] https://docs.microsoft.com/cs-cz/dotnet/standard/io/file-path-formats.

28. **dbpedia.** about. *wiki.dbpedia.org.* [Online] dbpedia. [Citace: 29. 10 2020.] https://wiki.dbpedia.org/about.

12 Seznam obrázků

13 Seznam tabulek