MASARYKOVA
UNIVERZITA

FAKULTA INFORMATIKY

Kdo, co, o čem – TextVis aneb vizualizace obsahu textu

Bakalářská práce

RADEK MEJZLÍK

Brno, jaro 2022

MASARYKOVA UNIVERZITA

FAKULTA INFORMATIKY

Kdo, co, o čem – TextVis aneb vizualizace obsahu textu

Bakalářská práce

RADEK MEJZLÍK

Vedoucí práce: doc. RNDr. Aleš Horák, Ph.D.

Katedra strojového učení a zpracování dat

Brno, jaro 2022

FACULTAS ARTIS INFORMATICÆ • UNIVERSITAS MASARYKIANA •

Prohlášení

Prohlašuji, že tato bakalářská práce je mým původním autorským dílem, které jsem vypracoval samostatně. Všechny zdroje, prameny a literaturu, které jsem při vypracování používal nebo z nich čerpal, v práci řádně cituji s uvedením úplného odkazu na příslušný zdroj.

Radek Mejzlík

Vedoucí práce: doc. RNDr. Aleš Horák, Ph.D.

Poděkování

Chtěl bych poděkovat panu doc. RNDr. Aleši Horákovi, Ph.D. za všechny odborné konzultace a poskytnuté rady, obzvláště si cením jeho nekonečné trpělivosti. Dále děkuji rodině za podporu při studiu a rovněž všem sestřičkám a lékařům z Interní hematologické a onkologické kliniky Fakultní nemocnice Brno, bez nichž by tato práce nemohla vzniknout, za jejich špičkovou péči. Obrovský dík patří především MUDr. Lukáši Semerádovi za záchranu mého života v listopadu roku 2017.

Stejně tak děkuji všem fyzioterapeutkám a ergoterapeutkám, které mne od té doby měly v péči. Zejména pak Tereze Kašparové, Petře Tvrdoňové, Janě Urbánkové, Elišce Veselkové, Michaele Schramlové, Ivaně Baťkové a Petře Sobolové, které mají na zlepšení mého stavu největší zásluhu, případně byly ochotné obětovat svůj volný čas, aby mi pomohly.

Shrnutí

Práce se zabývá problematikou automatické extrakce informací z nestrukturovaných textů v přirozeném jazyce jako jsou příspěvky na sociálních sítích či novinové články. Praktickou část práce tvoří analýza, návrh a implementace systému pro extrakci informací s názvem TextVis. Součástí systému je konzolová aplikace, jejímž účelem je identifikace entit a vztahů v česky psaných textech, analýza polarity daných textů a následná vizualizace získaných informací v podobě tabulky či síťového grafu. Tento graf navíc systém uživateli dovoluje upravit dle jeho libosti. Dostupná je také webová aplikace pro snadnější použití systému. Teoretická část práce pak nabízí kromě popisu samotného systému také shrnutí přístupů a technik používaných v oblasti extrakce informací, stejně tak jako přehled podobných projektů vytvořených ve světě. V závěru práce je čtenář seznámen s vyhodnocením úspěšnosti vyvinutého systému na skutečných textech jak z hlediska přesnosti systémem extrahovaných informací, tak z pohledu časové náročnosti samotné analýzy.

Klíčová slova

analýza a zpracování textu, dolování z textu, vizualizace, zpracování přirozeného jazyka, extrakce informací, rozpoznávání pojmenovaných entit, extrakce vztahů, analýza sentimentu, webová aplikace, síťový graf, D3.js

Obsah

1 Teorie **2**

1.1 Extrakce informací 2
1.2 Přístupy 3
1.2.1 Syntaktická analýza 3
1.2.2 Rozpoznávání pojmenovaných entit 4
1.2.3 Extrakce vztahů 4
1.2.4 Analýza sentimentu 5
1.3 Používané techniky 5
1.3.1 Definice pravidel 6
1.3.2 Strojové učení 6
1.3.3 Vyhledávání ve slovníku 8
1.3.4 Hybridní metody 8
1.4 Podobné projekty ve světě 8
1.4.1 Geneea Interpretor 8
1.4.2 SpaCy 10
1.4.3 Orange 11
1.4.4 Textnets 13
1.4.5 InfraNodus 15
1.4.6 GATE 16
1.4.7 Shrnutí 18

2 Analýza **20**

2.1 Požadavky 20
2.1.1 Funkční požadavky 20
2.1.2 Nefunkční požadavky 21
2.2 Použité nástroje 22
2.2.1 Unitok 22
2.2.2 Majka 22
2.2.3 SET 23
2.2.4 EFA 23
2.2.5 jusText 24
2.2.6 Aara 25
2.2.7 Derivancze 25
2.2.8 Propaganda Sentiment Analysis 26

2.2.9 NER a multiNER 26
2.2.10 VerbaLex 27
2.2.11 Declension 28
2.2.12 fastText 28
2.2.13 Data Driven Documents 29
2.3 Systém TextVis 30

3 Implementace **32**
3.1 Zpracování vstupu 32
3.2 Agregace informací 33
3.3 Výstup tabulkou 34
3.4 Síťový graf 35
3.4.1 Navigace v grafu 36
3.4.2 Obsah grafu 38
3.5 Konzolová aplikace 41
3.6 Webová aplikace 42

4 Vyhodnocení úspěšnosti systému **43**
4.1 Ladící sada 43
4.2 Úspěšnost se skutečnými texty 47
4.2.1 Zpravodajské články 49
4.2.2 Příspěvky na sociálních sítích 50
4.2.3 Blogy – amatérské články 51
4.3 Časová náročnost analýzy 53

Bibliografie **56**

Seznam tabulek

1.1 Korpus nadpisů novinových článků k události přistání člověka na Měsíci. Tyto nadpisy byly použity k vygenerování sítě na snímku 1.7. 13
1.2 Přehled vlastností uvedených systémů pro extrakci informací. 18
4.1 Údaje získané při testování textů ze zpravodajských článků. 50
4.2 Nasbíraná data v kategorii sociálních sítí. 51
4.3 Pozorovaná chybovost systému při zpracování textů z blogů. 52
4.4 Rychlost zpracování vybraných textů v různých kategoriích. Všechny uvedené číselné údaje reprezentují průměrnou délku běhu analýzy v dané kategorii v sekundách zaokrouhlenou na jedno desetinné místo. 54

Seznam obrázků

1.1 Strom syntaktických závislostí pro anglickou větu „Where do I buy fresh almonds?“ [14] 3

1.2 Schéma činnosti systému k rozpoznání pojmenovaných entit v textu při využití strojového učení s učitelem. Červenou barvou jsou v trénovací sadě zvýrazněny učitelem anotované výrazy představující jména osob, zelenou pak lokace a žlutou data. 7

1.3 Schéma extrakce informací z textového dokumentu systémem, jenž využívá hluboké učení. 8

1.4 Výstup systému Geneea Interpretor pro část článku ze serveru iDNES, včetně návrhů fotografií a štítků. 9

1.5 Výstup modulu DisplaCy pro anglicky psaný článek ze serveru History. 11

1.6 Síťový graf vytvořený systémem Orange, sestavený ze slov, která se objevují alespoň dvacetkrát ve 25 vybraných článcích na Wikipedii pro hledaný termín *automobil*. Hranami jsou propojena ta slova, která se v těchto článcích objevují vedle sebe alespoň pětkrát. Velikost uzlů odpovídá četnosti slov. 12

1.7 Bipartitní graf sestrojený systémem Textnets pro vstup sestávající z nadpisů článků v tabulce 1.1. Uzly jsou rozděleny do skupin algoritmem Leiden. Tyto skupiny jsou na obrázku zvýrazněny zelenou barvou. 14

1.8 Graf vygenerovaný systémem InfraNodus pro známý projev Martina Luthera Kinga *I Have a Dream*. 16

1.9 Výstup systému ANNIE pro část anglického článku ze serveru CNN. Zelenou barvou jsou zvýrazněny jména osob, červenou názvy organizací. Pod růžovou barvou nalezneme procentuální číselné hodnoty, zatímco fialová barva značí místopisná označení. Oranžová barva pak náleží časovým údajům. Nakonec jsou tyrkysovou barvou označeny ty termíny, které byly rozpoznány jako pojmenovaná entita, ovšem nebylo možné určit jejich kategorii. 17

2.1 Schéma procesu zpracování uživatelova vstupu k vytvoření vizualizace systémem TextVis. 31
3.1 Tabulka vygenerovaná webovou aplikací TextVis pro text „Rodiče vzali na narozeniny dceru do Paříže. Eva je šťastná, to si přála.“ při povoleném zpracování anafor 34
3.2 Část síťového grafu sestrojeného systémem TextVis pro článek o autonehodě ze serveru Brněnská Drbna 36
3.3 Porovnání barevného zvýraznění vět v grafu pro text „Výtržník zemřel na zámku v Lánech. Také Tomáš Garrigue Masaryk zemřel 14. 9. 1937 v Lánech.“ 37
3.4 Informace, které se zobrazí při umístění kurzoru na uzel *Václav Klaus* v grafu, který vznikl zadáním dvou souborů, přičemž soubor *klaus1.txt* obsahoval větu „Václav Klaus byl prezidentem do 7. března 2013.“ a soubor *klaus2.txt* obsahoval text „Klaus se netají láskou k lyžování. Jako prezident mimo jiné podepisoval zákony.“ 39
3.5 Část síťového grafu vygenerovaného pro text, který obsahuje věty „Bitva na Bílé hoře vypukla 8. listopadu 1620.“ a „Ferdinand II zrušil 23. 11. 1620 platnost Rudolfova majestátu.“. V grafu je patrné propojení uzlů *23. 11. 1620* a *8. listopad 1620* pomocí speciální hrany. 40
3.6 Tabulkový výstup systému TextVis pro text „Dnes na Primě dávali osvětový pořad z oblasti adiktologie, který dobře ilustroval, že pokud je pacient donucen ke zdrženlivosti až abstinenci, začne se mít po relativně krátké době docela k světu.“ Sloupec tabulky s určením způsobu, obsahující v druhém řádku slovo *dobře* a v posledním *docela* je na obrázku vynechán. 41
3.7 Úvodní stránka webové prezentace systému TextVis, vyzývající uživatele k zadání vstupu. 42

Úvod

S nástupem informačního věku, rozšířením informačních technologií a především s větší dostupností internetu se více a více lidské komunikace přesouvá právě do digitální podoby. Informace, které si lidé skrze internet předávají, mají nejčastěji podobu textu. Místo klasické pošty jsme dnes zvyklí posílat si dopisy e-mailem, novinové články dnes čteme namísto jejich klasické papírové podoby raději v mobilu. Žebříčky nejčastěji navštěvovaných webů ovládají sociální sítě, jako je Facebook, Twitter a podobné, kde se lidé obvykle dorozumívají textovými zprávami. Dalo by se říci, že lidé jsou v dnešní době obklopeni textem v elektronické podobě.

Tato bakalářská práce si klade za cíl vyvinout systém schopný extrahovat informace z libovolného nestrukturovaného textu v českém jazyce, jako je například článek ze zpravodajského webu nebo příspěvek na sociální síti, a tyto následně sumarizovat v přehledné podobě tabulkou a síťovým grafem. Tento systém dostal název TextVis.

Za tímto účelem jsou užity metody a nástroje vyvinuté v Centru zpracování přirozeného jazyka Fakulty informatiky Masarykovy univerzity, které jsou využity mimo jiné pro rozpoznání pojmenovaných entit v textu a následnou identifikaci vztahů mezi těmito entitami. Ohodnocena je taktéž polarita textu, a to pro každou větu zvlášť.

V rámci předkládané bakalářské práce jsem navrhl a implementoval systém TextVis v podobě konzolové aplikace. Součástí implementace je taktéž webová aplikace, která uživateli poskytuje možnost systém používat v přívětivější podobě odkudkoliv.

V kapitole 1 se věnuji problematice samotné extrakce informací a jednotlivým úkolům, které tento termín zastřešuje, stejně jako výčtu metod, se kterými se v této oblasti lze setkat. Ve stejné kapitole nabízím přehled podobných projektů, které již byly ve světě realizovány. Kapitola 2 se dále zabývá požadavky systému TextVis a představením všech použitých nástrojů a podpůrných programů. Kapitola 3 potom navazuje popisem vlastní implementace systému. V kapitole 4 práci zakončuji prezentací výsledků testování vyvinutého systému na skutečných datech. Na stejném místě je také zhodnocena časová náročnost samotné analýzy v závislosti na různých parametrech, jako je například délka vstupního textu.

1 Teorie

1.1 Extrakce informací

Extrakcí informací (angl. *Information extraction – IE*) rozumíme úlohu zpracování přirozeného jazyka, jejímž cílem je automatický převod vstupních dat z dokumentu v přirozeném jazyce do podoby vhodné pro uložení v databázi nebo další strojové zpracování. Cílem extrakce informací je tedy získat důležité informace[1] ze vstupních dat (dokumentu) a tyto převést do předem definovaného strukturovaného formátu [1, 2].

Zpravidla je vstupem pro extrakci informací částečně strukturovaný text (seznamy, tabulky a podobně) či text zcela nestrukturovaný. Vstup ale může nabývat i jiných podob, jako jsou obrazová data, ze kterých je nejprve nutné text získat pomocí optického rozpoznávání znaků (OCR) [3] nebo zvukové stopy obsahující mluvené slovo [4]. V této práci se však zaměřuji pouze na zpracování a sumarizaci nestrukturovaných textových dokumentů.

Zde popisovaný pojem extrakce informací je nutné nezaměňovat s pojmem *získávání informací* (angl. information retrieval – IR). V případě získávání informací se totiž jedná pouze o nalezení relevantních dokumentů na základě parametrů vyhledávání. Informace v těchto dokumentech obsažené musí proto uživatel zpracovat dál sám, nepodléhají další automatické analýze. Typickým systémem pro získávání informací je webový vyhledávač [5].

Extrakce informací má celou řadu praktických využití, typickými oblastmi jsou:

- Automatická sumarizace textů [6]
- Dolování dat o událostech, osobách, produktech či jiných subjektech zájmu z webových stránek nebo zpravodajských článků [1, 7, 8]
- Sledování trendů ve společnosti monitoringem sociálních sítí [8, 9]
- Automatické vytváření metadat k dokumentům na základě jejich obsahu [7]

1. Co považujeme za důležitou informaci záleží na úkolu, který řešíme. Může to být například časový údaj, jméno, peněžní částka a podobně.

- Biologický a chemický výzkum, výzkum léků [1, 8, 10]
- Analýza zdravotní dokumentace [8, 9, 11, 12]

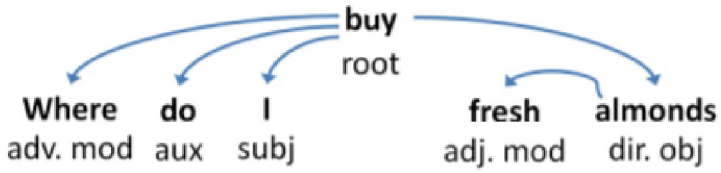
Obecně se dá říci, že všechny systémy pro extrakci informací mají podobnou strukturu, kdy každý provádí tyto základní kroky: [6]

1. Tokenizace, kdy dochází k rozčlenění textu na věty a jednotlivých vět na slova
2. Morfologická analýza, kdy se pro jednotlivá slova zjišťují jejich gramatické kategorie
3. Syntaktická analýza, dále popsána v sekci 1.2.1
4. Samotná identifikace a zpracování informací

Úspěšnost jednotlivých systémů pro extrakci informací potom vyhodnocujeme pomocí tří metrik – přesnosti (precision), pokrytí (recall) a jejich harmonického průměru, takzvané F-míry (F-measure nebo také F-score) [13].

1.2 Přístupy

1.2.1 Syntaktická analýza



Obrázek 1.1: Strom syntaktických závislostí pro anglickou větu „Where do I buy fresh almonds?“ [14]

Automatická syntaktická analýza je softwarem prováděný rozbor vnitřní struktury věty, který spočívá v identifikaci větných konstituentů,[2] odhalování jejich vzájemných závislostí a určení syntaktické role jednotlivých částí [15]. Součástí syntaktické analýzy je také eliminace možných nejednoznačností významu věty. Obzvlášť v případě češtiny je tento krok důležitý vzhledem k bohatosti jazyka a poměrně volnému pořadí slov ve větách, které k nejednoznačnosti přispívá.

2. Slovo nebo část věty, která se skládá z více slov, ale chová se jako jeden celek, například několikanásobný podmět.

Syntaktickým analyzátorem potom nazýváme takový program, který tuto úlohu řeší.

Výsledkem syntaktické analýzy je zpravidla strom syntaktických závislostí, který lze dále využít při některém z úkolů spadajících do kategorie extrakce informací. Na obrázku 1.1 se nachází příklad takového syntaktického stromu.

1.2.2 Rozpoznávání pojmenovaných entit

Rozpoznávání pojmenovaných entit[3] (Named Entity Recognition – NER) si klade za cíl v zadaném nestrukturovaném textu rozpoznat pojmenované subjekty a tyto rozdělit do předem definovaných kategorií, které povětšinou tvoří jména osob, názvy společností a organizací, vyjádření data či času, lokace[4] nebo adresy [10, 16]. Úlohu je možné pojmout i obecněji a zahrnout taktéž rozpoznání peněžních hodnot, procentuálních číselných údajů, e-mailových adres a tak dále.

Podúkolem při NER je normalizace pojmenovaných entit, tedy schopnost systému rozeznat, že výrazy *Zeman*, *Miloš Zeman* a *český prezident* představují jednu jedinou entitu, a nevytvářet tak při extrakci duplikáty.

1.2.3 Extrakce vztahů

Po tom, co jsou identifikovány jednotlivé entity vystupující v textu, se nabízí otázka, zda nemohou mít tyto entity mezi sebou nějaký vzájemný sémantický vztah. Například z věty „Pepa pracuje pro ČEZ už od roku 2007.“ je možné vydedukovat, že vztah mezi osobou Pepa a společností ČEZ je zaměstnanec–zaměstnavatel a že tento vztah už několik let trvá. Obdobný vztah se může objevit mezi libovolnou dvojicí pojmenovaných entit [17].

Stejným způsobem lze z textu v přirozeném jazyce odvozovat různé vztahy, ne jen ty „mezilidské“. Kupříkladu věta „Toskánsko se nachází na severozápadě Itálie.“ nám dává informaci, že Toskánsko leží v Itálii. Výsledkem extrakce vztahů z takové věty by byla trojice (Toskánsko – nacházet se – Itálie), kde vztahem (relací) je *nacházet se*. Dalším příkladem může být méně triviální odvození vztahu (Jennifer

3. Někdy se též může objevit pod názvem *extrakce entit*.
4. Patří sem názvy států, měst, řek, památek apod.

Aniston – být ex-manželkou – Brad Pitt) z tvrzení „Brad Pitt se rozvedl s Jennifer Aniston v říjnu 2005.“

Ve skutečných textech ovšem nejsou informace podávány v takto jednoduché podobě a mohou vyplývat třeba až při čtení napříč větami. Úkolem při **extrakci vztahů**[5] (anglicky *Relation extraction* – RE) je potom takové vztahy v rámci textu automaticky identifikovat a správně pojmenovat [18].

Při zkoumání odborných textů se tato úloha využívá třeba pro nalézání vztahů mezi léky a nemocemi nebo nemocí a jejími projevy ve vědeckých pracích [12].

1.2.4 Analýza sentimentu

Při **analýze sentimentu** jsou zkoumány emoce, názory a postoje autora daného textu vzhledem k nějakému subjektu (veřejně známé osobnosti, události, produktu, ...) [19]. Závěr takové analýzy může být vyjádřen jak polaritou (má pozitivní × neutrální × negativní postoj), tak i detailněji, například přímo zjištěnou emocí (strach, radost, smutek, hněv, ...) [20].

Své využití má tento přístup zejména v případě komerčních firem. Ty si mohou nechat softwarem zanalyzovat spokojenost zákazníků se svými produkty a službami na základě uživatelských recenzí či názorů, které uživatelé prezentují na sociálních sítích. Tyto znalosti poté využijí při marketingu a zkvalitňování svých služeb [21].

Dalším využitím může být sledování trendů ve společnosti, jako jsou politické názory obyvatel nebo jejich pohled na události veřejného života.

1.3 Používané techniky

Pro popsané úkoly jsou neustále vyvíjeny nové a pokročilejší přístupy. Cílem při návrhu těchto metod je dosáhnout co možná nejvyšší rychlosti algoritmů, maximalizovat schopnost detekce hledaných informací a naopak vyhnout se falešně pozitivním výsledkům a tím dosáhnout co možná nejlepší F–míry. V této kapitole popisuji metody, které se uplatňují při extrakci informací v letech 2020 až 2022.

5. Někdy označována jako *detekce vztahů* (Relation Detection – RD).

Třebaže každý systém je vždy uzpůsoben na míru doméně[6] a danému úkolu, principy fungování systémů pro extrakci informací je možno obecně rozdělit do následujících základních kategorií.

1.3.1 Definice pravidel

Dříve se metody založené na definici pravidel těšily velké popularitě a přestože již byly převážně nahrazeny metodami využívajícími strojové učení, stále se v některých případech používají. Při budování systému s použitím těchto metod jsou člověkem ručně vytvářeny vzory – pravidla [12]. Tato pravidla bývají nejčastěji stavěna na výsledcích morfologické a syntaktické analýzy, hledání konkrétních podřetězců, popřípadě na regulárních výrazech [10].

Nevýhodou těchto metod je potřeba rozsáhlých znalostí lingvistiky (nejvhodnější je spolupráci přímo s odborníky – lingvisty), časová náročnost samotného vytváření pravidel, stejně jako obtížná rozšiřitelnost takového systému [22].

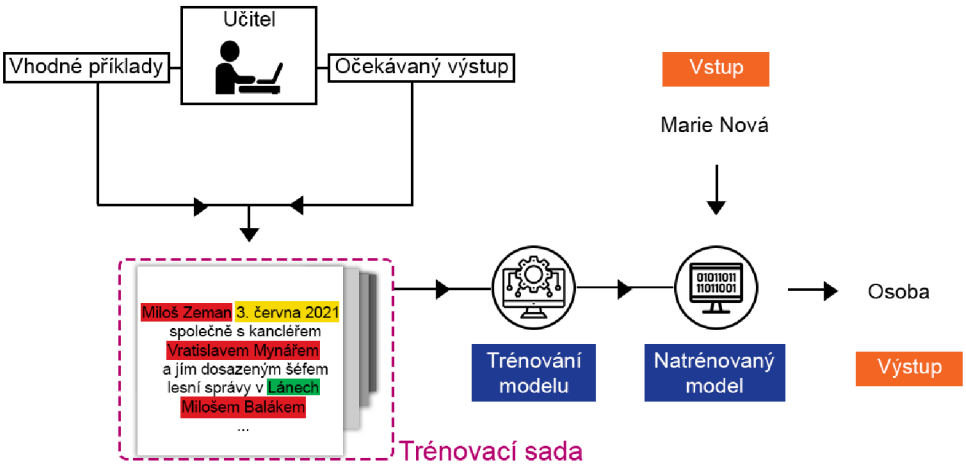
Za výhodu tohoto přístupu lze považovat, že pravidla mohou být „vyšperkována“ dle znalosti specifické domény, ve které se systém bude využívat a díky tomu můžeme při zvolení tohoto přístupu dosahovat přesnějších výsledků. Tyto metody proto pořád nachází uplatnění, například při analýze právních textů [23].

1.3.2 Strojové učení

Obzvlášť úspěšná se v posledních letech ukázala řešení využívající strojové učení, a to nejen bez učitele nebo s učitelem, nýbrž i takzvané hluboké učení.

Tyto metody v mnohých případech staví na tom, že se extrakce informací pojme jako čistě klasifikační problém, kdy je systému předložena rozsáhlá databáze vzorů – příkladů, s dodatečnou informací o tom, jakým způsobem by tyto vzory měly být klasifikovány. Klasifikace může probíhat pouze na dvoustupňové či třístupňové úrovni, což je vhodné kupříkladu při analýze sentimentu na úrovni pozitivní/negativní, případně i neutrální [24] nebo jako začlenění do určité kategorie, což je vhodné zejména při kategorizaci pojmenovaných entit. Ilustrace práce takového systému je k dispozici na obrázku 1.2.

6. Množina očekávaných vstupů, například vědecké práce z oblasti chemie.

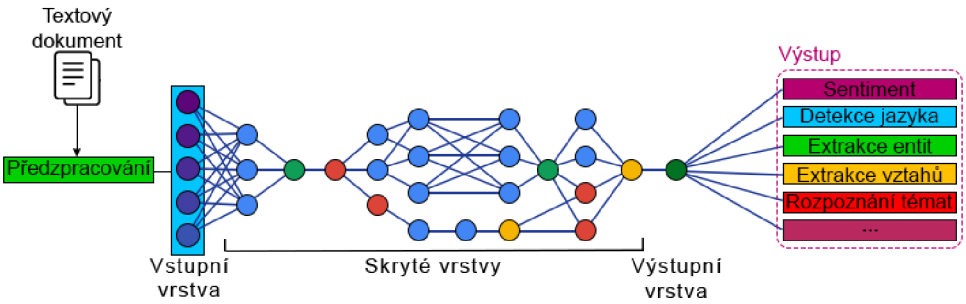


Obrázek 1.2: Schéma činnosti systému k rozpoznání pojmenovaných entit v textu při využití strojového učení s učitelem. Červenou barvou jsou v trénovací sadě zvýrazněny učitelem anotované výrazy představující jména osob, zelenou pak lokace a žlutou data.

Jiným užitím strojového učení může být automatické vygenerování jazykového modelu, tedy specifické interní reprezentace slov, z korpusu vstupních textů. V takovém modelu potom může být slovo navíc zakódováno spolu se svým kontextem, tedy bezprostředním okolím slova. Vzhledem k tomu, že model je vytvářen počítačem, nejen že může vytvořit velmi přesnou reprezentaci, kterou by ručně bylo možné vytvořit jen stěží, ale zvládne zpracovat i korpusy obsahující velmi velké množství textů. Vygenerovaný jazykový model nalezne své uplatnění právě při extrakci informací, například extrakci entit ze zdravotnické dokumentace.

Nevýhodou jazykového modelu je značná výpočetní náročnost při jeho vytváření. Přesně proto jsou dostupné již natrénované jazykové modely, které je potom možné upravit v závislosti na typu extrahovaných informací a doméně textů. Mezi osvědčené modely patří BERT [25], RoBERTa [26] a GPT-2 [27].

Čím dál více se z důvodu jejich vysokého výkonu objevují v oblasti extrakce informací systémy implementující takzvané hluboké učení. Schéma práce takového systému nalezneme na obrázku 1.3.



Obrázek 1.3: Schéma extrakce informací z textového dokumentu systémem, jenž využívá hluboké učení.

1.3.3 Vyhledávání ve slovníku

Mezi úspěšné metody se řadí i ty založené na vyhledávání ve slovníku. Při využití takové metody je nejprve zkonstruována rozsáhlá databáze termínů (slovník) a následně se výrazy v textu testují na přítomnost ve slovníku. Testování může tolerovat jak striktně přesnou shodu, tak i výrazy alespoň podobné.

Tyto metody nachází své využití zejména při rozpoznávání pojmenovaných entit. Systémy využívající takovou metodu však mají tendenci k nižší odezvě, vzhledem k tomu, že nemusí zachytit všechna možná synonyma či různá hláskování hledaného termínu [10, 12, 22].

1.3.4 Hybridní metody

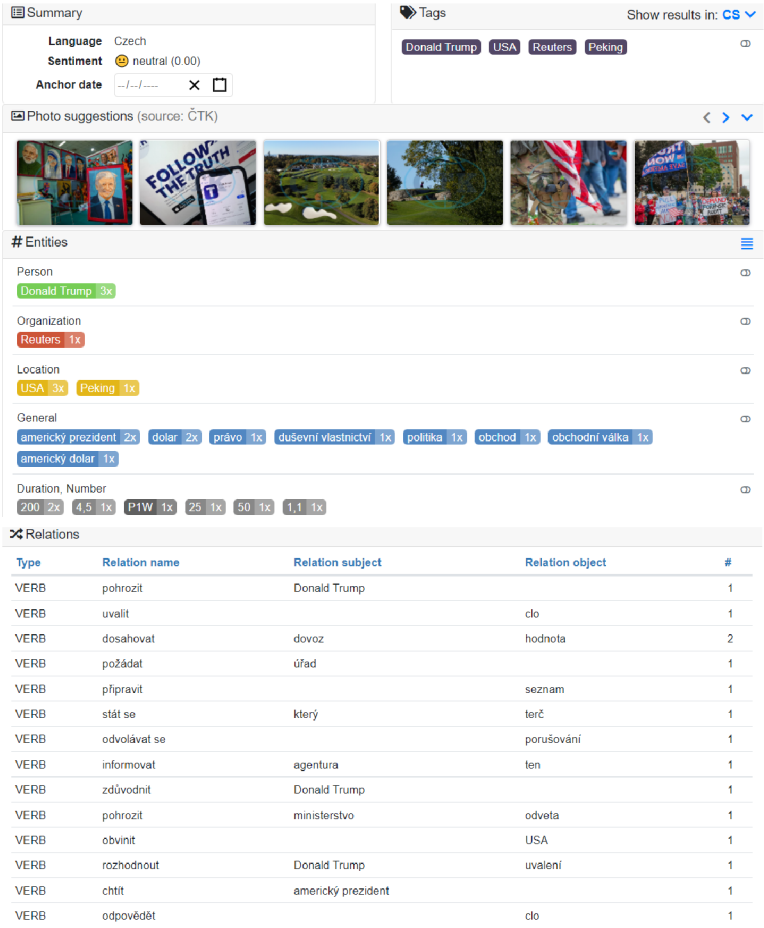
Existují taktéž systémy využívající různé kombinace výše uvedených technik. Tím mohou využít výhod jednoho z přístupů a zároveň eliminovat jeho nedostatky. Poměrně běžná je kombinace strojového učení s manuální definicí pravidel či vyhledáváním ve slovníku [10].

1.4 Podobné projekty ve světě

1.4.1 Geneea Interpretor

Česká firmy Geneea Analytics s.r.o. nabízí kromě produktů pro generování zpravodajských článků nebo automatické přiřazování štítků (tzv. tagů) k článkům napsaným člověkem také službu **Interpretor** [28] pro rozpoznávání pojmenovaných entit v textu jako jsou organizace,

osoby, lokace, časové a datové údaje, zpětné časové odkazy[7] a relace, podobně jako TextVis. Nabízí také rozpoznání emailových adres, URL odkazů a oproti TextVis navíc dokonce i hashtagů,[8] či takzvaných mentions[9] v příspěvcích ze sociálních sítí. Rozpoznané entity dále rozdělí do příslušných kategorií. Stejně jako v případě systému TextVis je text ohodnocen na základě své polarity. Neděje se tak však na úrovni jednotlivých vět, nýbrž textu jako celku.



Type	Relation name	Relation subject	Relation object	#
VERB	pohrozit	Donald Trump		1
VERB	uvalit		clo	1
VERB	dosahovat	dovoz	hodnota	2
VERB	požádat	úřad		1
VERB	připravit		seznam	1
VERB	stát se	který	terč	1
VERB	odvolávat se		porušování	1
VERB	informovat	agentura	ten	1
VERB	zdůvodnit	Donald Trump		1
VERB	pohrozit	ministerstvo	odveta	1
VERB	obvinit		USA	1
VERB	rozhodnout	Donald Trump	uvalení	1
VERB	chtít	americký prezident		1
VERB	odpovědět		clo	1

Obrázek 1.4: Výstup systému Geneea Interpretor pro část článku ze serveru iDNES, včetně návrhů fotografií a štítků.

7. Například *před rokem* nebo *o den dříve*.
8. Klíčové slovo uvedené znakem mřížky (#).
9. Jedná se o označení profilu jiného uživatele nebo organizace.

Narozdíl od TextVis nabízí Geneea pouze vizualizaci formou tabulky, zato má několik dalších funkcí. Oproti TextVis je možné tuto službu použít nejen pro česky psané texty. Podporována je také slovenština, angličtina, nizozemština, francouzština a v době psaní této práce má firma v plánu přidat podporu také pro němčinu, polštinu a španělštinu, které jsou prozatím dostupné v omezeném rozsahu.

Jazyk textu je navíc automaticky rozpoznán, stejně jako hlavní téma zadaného textu. Cílovou skupinou služby jsou novináři, na základě obsahu textu proto dokáže také navrhnout štítky k článku a fotografie související s daným tématem.

Služba je pro komerční použití zpoplatněna. Cena základního balíčku začíná v době psaní této práce na 39 eurech měsíčně.[10] Webové demo je k vyzkoušení volně dostupné.[11] Příklad výstupu je potom k vidění na obrázku 1.4.

1.4.2 SpaCy

Jedním z nástrojů pro extrakci informací z textu je knihovna SpaCy [29] dostupná pro programovací jazyk Python. SpaCy nabízí celou řadu funkcí s využitím při zpracování přirozeného jazyka – jedná se například o moduly pro segmentaci vět a slov, morfologickou analýzu, syntaktickou analýzu, lemmatizaci a další.

Jedním z modulů v portfóliu SpaCy je také takzvaný *EntityRecognizer*, který je určený k detekci pojmenovaných entit a jejich kategorií. Zatímco SpaCy poskytuje podporu pro více než 60 jazyků včetně češtiny a slovenštiny [30], EntityRecognizer podporuje pouze 13 z nich. V těch bohužel není zastoupena čeština ani slovenština. Pro vizualizaci entit objevených tímto modulem je ve SpaCy obsažen i modul DisplaCy. DisplaCy v textu objevené entity zvýrazní dle jejich kategorie a výsledek zobrazí buď ve formátu HTML, SVG nebo jako buňku v poznámkových blocích Jupyter[12] [31]. Ukázka výstupu modulu DisplaCy je dostupná na obrázku 1.5.

Velkou výhodou SpaCy je fakt, že se jedná o volně dostupnou knihovnu s otevřeným kódem. Natrénované modely je tudíž možné libovolně doplňovat a upravovat dle potřeb uživatele. S využitím této

10. Viz `https://geneea.com/interpretor/pricing.html`
11. `https://demo.geneea.com/`
12. Webová aplikace pro vizualizaci dat

knihovny proto vznikly takové projekty jako je síť vztahů mezi entitami v románech o Harry Potterovi [32] či podobná síť pro postavy ze série filmů Avengers [33].

Obrázek 1.5: Výstup modulu DisplaCy pro anglicky psaný článek ze serveru History.

1.4.3 Orange

Orange [34] je název nástroje pro dolování dat a jejich vizualizaci, vyvíjený již od roku 1996 na univerzitě v Lublani. Nástroj nabízí uživateli takzvané vizuální programování, kdy se algoritmus tvoří namísto psaním kódu pouze přetahováním takzvaných widgetů[13] z nabídky. Vstup a výstup těchto widgetů lze na sebe napojovat tak, aby uživatel obdržel kýžený výsledek, podobně jako je tomu u metod či funkcí v klasických programovacích jazycích. Díky vizuálnímu programování je nástroj vhodný jak pro experty, tak i pro naprosté nováčky. Pokud si však uživatel přeje programovat kódem, jsou funkce nástroje Orange dostupné i skrze knihovnu pro jazyk Python.

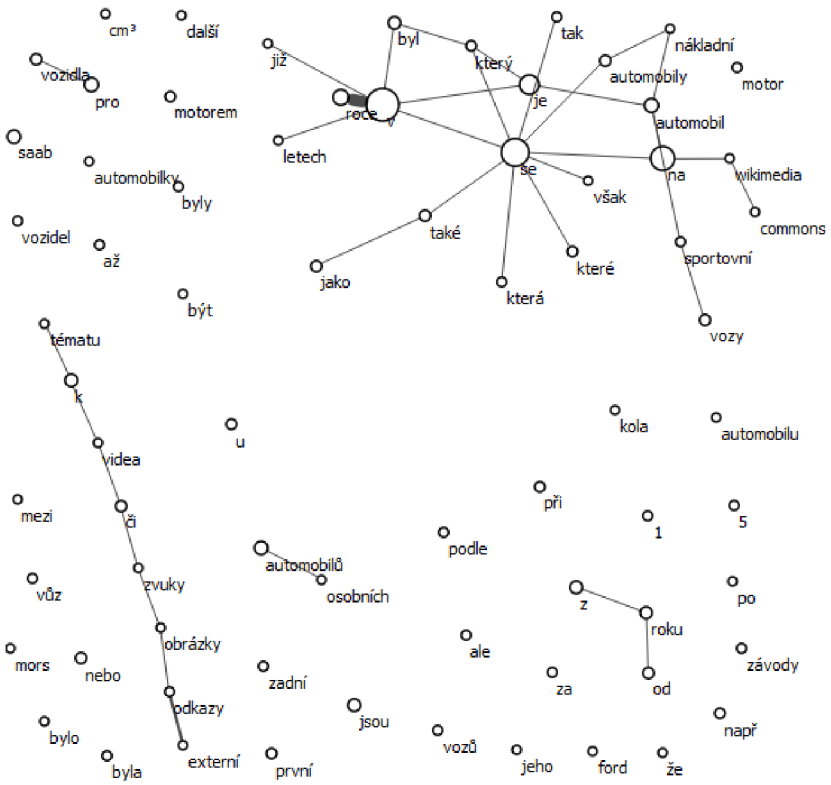
Pro zobrazení výsledků výpočtu si uživatel tohoto nástroje může zvolit z celé škály vizualizací, od klasických sloupcových či liniových grafů, přes Vennovy diagramy, až po teplotní mapu. Jedním z dostupných widgetů je také převod korpusu[14] do síťového grafu. Uživatel si může vybrat, zda si přeje, aby uzly grafu představovaly dokumenty nebo přímo slova v nich obsažená. Pokud si vybere první variantu, mezi dokumenty nalezne hranu v případě, že se v obou dokumentech objevuje alespoň takový počet slov, které si uživatel nastaví v parametru. Ve druhém případě mezi slovy vznikne hrana, pokud se tato

13. Jednotlivé kroky algoritmu, například načtení dat ze souboru.
14. Soubor dokumentů.

slova objevují v dostatečné blízkosti nastavené parametrem *window size* alespoň tolikrát, jako bylo nastaveno parametrem *threshold* [35].

Co se týče widgetů pro extrakci informací z textu, Orange poskytuje takové možnosti jako je rozpoznání polarity textu, rozpoznání geografických názvů, identifikaci nejčastějších slov v dokumentu a další [36]. Pokud uživateli nestačí základní nabídka widgetů, do Orange je možné doinstalovat další v podobě add-onů vytvořených komunitou.

Orange umí kromě souborů s nestrukturovaným textem zpracovávat také příspěvky ze sociální sítě Twitter a v těchto identifikovat hashtagy i mentions. Ukázkový výstup jsem připravil na obrázku 1.6.



Obrázek 1.6: Síťový graf vytvořený systémem Orange, sestavený ze slov, která se objevují alespoň dvacetkrát ve 25 vybraných článcích na Wikipedii pro hledaný termín *automobil*. Hranami jsou propojena ta slova, která se v těchto článcích objevují vedle sebe alespoň pětkrát. Velikost uzlů odpovídá četnosti slov.

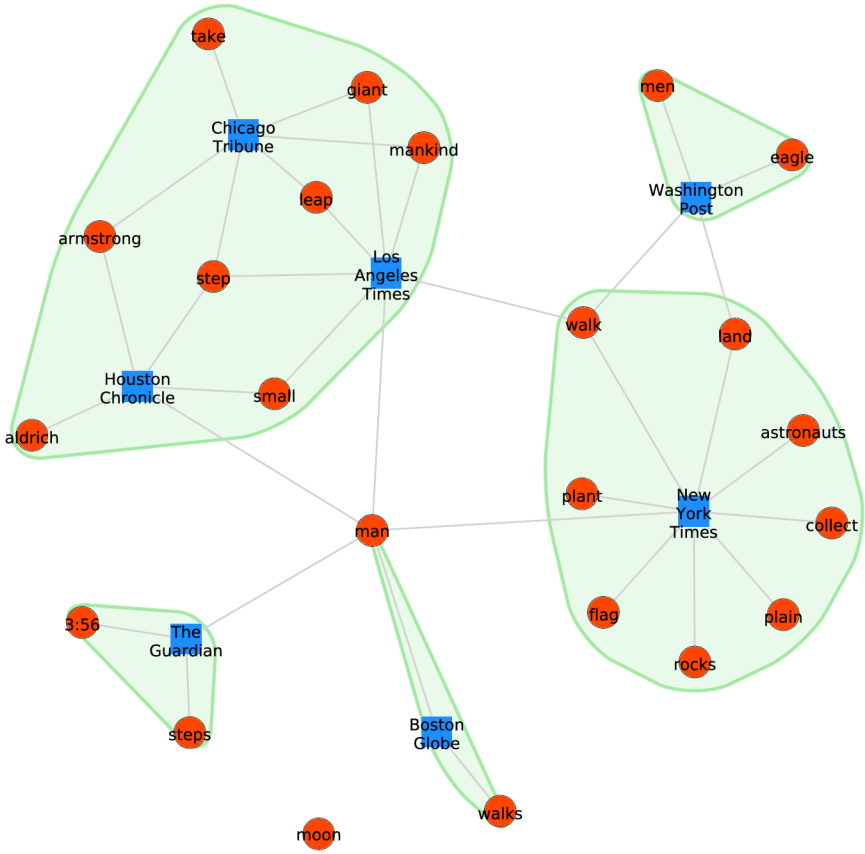
1.4.4 Textnets

Původně byla pod názvem Textnets vyvíjena knihovna pro analýzu textu v jazyce R [37]. V roce 2020 však John D. Boy vytvořil téměř identický port [38] této knihovny pro jazyk Python, který staví na již zmíněné knihovně SpaCy. Hlavní funkcí je možnost z množiny vstupních nestrukturovaných textových dokumentů sestrojit síťový graf.

Zvláštností výstupního grafu je, že je bipartitní – jednu množinu vrcholů tvoří vstupní dokumenty a druhou množinu slova v těchto dokumentech se vyskytující. Každý uzel představující dokument je spojený s uzly symbolizujícími slova v tomto dokumentu obsažená. Uzly grafu mohou volitelně být rozděleny do skupin, kdy každá skupina sdílí stejné téma (v případě slov), či používá podobné fráze nebo slova (v případě dokumentů). Tyto skupiny jsou detekovány pomocí algoritmu Leiden [39]. Uživatel si může také vygenerovat projekci grafu pro pouze jednu množinu vrcholů.

Tabulka 1.1: Korpus nadpisů novinových článků k události přistání člověka na Měsíci. Tyto nadpisy byly použity k vygenerování sítě na snímku 1.7.

Deník	Název článku
The Guardian	3:56 am: Man Steps On to the Moon
New York Times	Men Walk on Moon – Astronauts Land on Plain, Collect Rocks, Plant Flag
Boston Globe	Man Walks on Moon
Houston Chronicle	Armstrong and Aldrich „Take One Small Step for Man“ on the Moon
Washington Post	The Eagle Has Landed – Two Men Walk on the Moon
Chicago Tribune	Giant Leap for Mankind – Armstrong Takes 1st Step on Moon
Los Angeles Times	Walk on Moon – That's One Small Step for Man, One Giant Leap for Mankind



Obrázek 1.7: Bipartitní graf sestrojený systémem Textnets pro vstup sestávající z nadpisů článků v tabulce 1.1. Uzly jsou rozděleny do skupin algoritmem Leiden. Tyto skupiny jsou na obrázku zvýrazněny zelenou barvou.

Celkově podporuje modul Textnets 18 světových jazyků, mezi nimiž čeština chybí [40]. Výsledný graf je možné exportovat ve vektorových formátech EPS a SVG, v rastrovém formátu PNG nebo ve formátu Graph Modeling Language (GML) pro otevření v jiných programech [41]. Celý modul je dostupný ve veřejném repozitáři na

GitHub.[15] Ukázku grafu sestrojeného modulem Textnets je možné si prohlédnout na obrázku 1.7.

1.4.5 InfraNodus

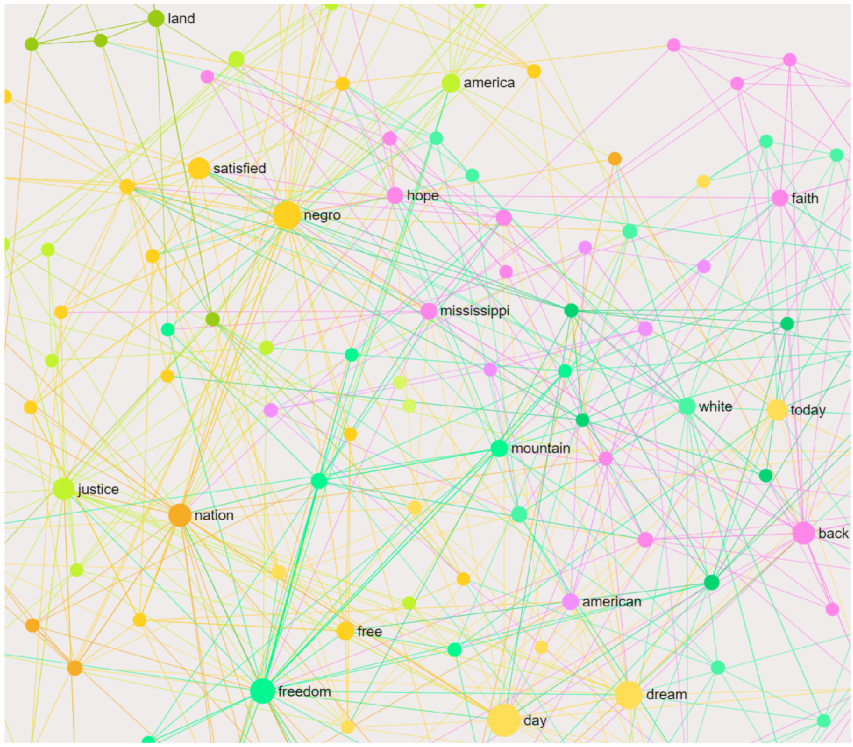
InfraNodus [42] je nástrojem, který dokáže na základě analýzy libovolného textu vytvořit síťový graf. Uzly tohoto grafu tvoří slova. Hranami jsou propojena ta slova, jenž se v textu objevují společně. Velikost jednotlivých uzlů závisí od toho, jak často se dané slovo v textu objevuje. Stejnou barvou jsou pak odlišeny ty uzly, které souvisí se stejným tématem.

InfraNodus umí zpracovávat jak nestrukturovaný text (a to včetně PDF dokumentů), tak i částečně strukturovaná data, například ve formátu CSV. Další možností, jak zadat vstup pro zpracování, je nahrát mluvené slovo. To je před zpracováním převedeno na text pomocí rozpoznávání řeči a s tímto textem je dále nakládáno, jako by byl zadán právě tento text. Tato možnost je však dostupná pouze u některých jazyků.

Vygenerovaný graf může být uživatelem dále filtrován i jinak upravován. Je možné zadat další text ke zpracování, který se zakomponuje do stávajícího grafu, přesouvat nebo odebírat uzly a podobně.

InfraNodus je nástupcem obdobného systému Textexture[16] od stejných tvůrců a kromě angličtiny podporuje také němčinu, francouzštinu, ruštinu, španělštinu, portugalštinu, švédštinu, italštinu a nizozemštinu. Pro použití tohoto nástroje je třeba účet, který vyžaduje měsíční poplatek. Cena účtu pro osobní použití je 9 euro za měsíc, pro komerční použití potom 29 euro měsíčně [43]. Ukázku výstupu vidíme na obrázku 1.8.

15. `https://github.com/jboynyc/textnets`
16. `https://textexture.com/`



Obrázek 1.8: Graf vygenerovaný systémem InfraNodus pro známý projev Martina Luthera Kinga *I Have a Dream*.

1.4.6 GATE

GATE [44], celým názvem General Architecture for Text Engineering, je jedním z nejstarších systémů pro extrakci informací z nestrukturovaných textových dokumentů vůbec. Od roku 1995 jej jako volný software vyvíjí Univerzita v Sheffieldu. Tento systém, naprogramovaný v jazyce Java, nabízí formou pluginů funkce pro zpracování nestrukturovaného textu v přirozeném jazyce.

Stěžejní součástí systému GATE je předinstalovaný plugin ANNIE [45] (A Nearly-New Information Extraction System). Jedná se o soubor modulů pro jednotlivé úkony potřebné k extrakci informací, jako je tokenizace, morfologická analýza, rozpoznávání pojmenovaných entit nebo zpracování anafor.[17]

17. Zpětný odkaz k již řečenému.

GATE podporuje 12 jazyků, v základní výbavě angličtinu, formou pluginů dále němčinu, čínštinu, francouzštinu, ruštinu a další [46].

Jako vstup systém přijímá jak textové a HTML soubory, tak i dokumenty ve formátu PDF a DOC. Od roku 2013 je pak možné formou pluginu doinstalovat funkcionalitu pro zpracování příspěvků ze sociální sítě Twitter [47].

Ukázka rozpoznávání pojmenovaných entit pluginem ANNIE je k vidění na snímku 1.9.

trade association for inbound tourism to Europe.
Kurt Janson, director of the UK's Tourism Alliance , knows who he thinks is to blame. "Some of the problems are government-inflicted as well as Covid-inflicted," he says.

So what exactly is going on in Blighty?
'Plague island'
London transport is the one place in England with a mask mandate, but it is routinely ignored.

First, of course, there's the pandemic.

While the UK was the first country in Europe to kick off its vaccination program, in recent months it has lagged behind its peers. Just under 69% of Britons have been vaccinated, according to Johns Hopkins University. Portugal, on the other hand, is at an 87% vaccination rate.

And while many other tourist-heavy countries have continued mask mandates -- in France, Spain and Italy, masks must be worn at all times indoors -- the UK has trumpeted the sloughing off of restrictions. In November, Prime Minister Boris Johnson faced criticism for not wearing a mask during a tour of a hospital.
But while the relaxation of the rules was intended to open the UK up, those outside are seeing it in a different way.
"Safety is the number one concern for Chinese visitors -- we're more sensitive and more careful than other countries," says Marcus Lee, CEO of China Travel Online .
Related content
Travel to the UK during Covid-19: What you need to know before you go
The UK's case numbers spiraled far beyond other European countries during September and October. And although numbers are on the rise across the continent, with countries such as Germany now seeing massive spikes in infection rates and the Netherlands re-entering lockdown, the UK still registered the second highest number of cases in the world over the past four weeks, according to Johns Hopkins University -- second only to the US, whose population nudging 330 million dwarfs the UK's 67 million.
Harsh quarantine rules mean outbound travel from China is effectively blocked, so Chinese visitors aren't going anywhere right now. But when they do -- Lee reckons in the first half of 2022, when almost 100% of the country has been vaccinated -- he says they won't be making a beeline for the UK.
"Eventually, I hope the UK cases will go down, and I think you will see more Chinese when that happens."
Europeans, too, are proving reluctant to hop over to what, late last year, the New York Times called "plague island."
Fabio Bergonzini, from Bologna, Italy, used to be a regular visitor to the UK. A fluent English speaker and lover of all things British, he visited England three times in 2019 -- city breaks in London and Manchester plus a country road trip around the northern region of Yorkshire -- but has not been back since the pandemic started.

Obrázek 1.9: Výstup systému ANNIE pro část anglického článku ze serveru CNN. Zelenou barvou jsou zvýrazněny jména osob, červenou názvy organizací. Pod růžovou barvou nalezneme procentuální číselné hodnoty, zatímco fialová barva značí místopisná označení. Oranžová barva pak náleží časovým údajům. Nakonec jsou tyrkysovou barvou označeny ty termíny, které byly rozpoznány jako pojmenovaná entita, ovšem nebylo možné určit jejich kategorii.

1.4.7 Shrnutí

Tabulka 1.2 přináší porovnání způsobů vizualizace, typů extrahovaných informací, cen, podporovaných jazyků a vstupních formátů uvedených systémů pro extrakci informací.

Tabulka 1.2: Přehled vlastností uvedených systémů pro extrakci informací.

Systém	Vizualizace	Informace	Jazyky[i]	Cena[ii]	Vstup[iii]
Geneea Interpretor	Zvýraznění v textu, tabulka	Sentiment, téma a jazyk textu, návrh fotografií a značek, pojmenované entity, relace	cs, sk, en, nl, fr, pl[iv], es[iv], de[iv]	39 €	holý text, HTML, PDF, DOC
SpaCy	Zvýraznění v textu, strom syntaktických závislostí	Pojmenované entity	cs, sk a dalších 64	0 €	holý text
Orange	Síťový graf, tabulka, sloupcový graf, liniový graf, …[v]	Souvýskyt slov, téma, sentiment, klíčová slova, geografické názvy	en, es, de, ar, ru a dalších 45	0 €	holý text, CSV, SQL, Twitter, Wikipedia, PubMed, NY Times, The Guardian

Textnets	Bipartitní síťový graf	Souvýskyt slov, téma textu	ca, da, de, en, es, fr, it, ja, lv, mk, no, nl, pl, pt, ro, ru, zh	0 €	holý text, CSV, SQL
InfraNodus	Síťový graf	Souvýskyt slov, téma textu	en, de, fr, ru, es, pt, sv, it, nl	9 €	holý text, PDF, CSV, mluvené slovo
GATE	Zvýraznění v textu	Pojmenované entity	en, de, zh, fr, ru, bg, ar, da, ceb, hi, cy, ro	0 €	holý text, HTML, PDF, DOC, Twitter

[i] Jazyky jsou uvedeny svým kódem dle normy ISO 639-1, vyjma cebuánštiny, která je uvedena kódem *ceb* dle normy ISO 639-2. [ii] Nejnižší částka, kterou musí zákazník zaplatit, aby získal k systému přístup, třebaže s omezenou funkcionalitou. Ceny jsou uvedené jako měsíční varianta předplatného. [iii] U služeb Twitter a dalších jsou myšlena data přístupná přes příslušný API klíč – příspěvky, články, komentáře atd. [iv] Jazyk je dostupný v době psaní práce dostupný v omezeném rozsahu. [v] Kompletní seznam vizualizací je dostupný na [36]

2 Analýza

Cílem této bakalářské práce bylo vyvinout systém, jenž na vstupu obdrží nestrukturovaný text v jazyce českém původem z internetu a za pomocí agregace dat získaných z nástrojů vytvořených v Centru zpracování přirozeného jazyka Masarykovy univerzity pro extrakci faktů, identifikaci pojmenovaných entit a analýzu sentimentu vytvoří pro tento text vizualizaci v podobě tabulky, případně síťového grafu. Tato kapitola přináší přehled požadavků a užitých nástrojů, rovněž jako základní návrh systému.

2.1 Požadavky

Vyjma toho, že systém bude umět zpracovat kromě holého textu v podobě textového souboru také soubory HTML nebo URL odkazující například na webový článek, byly na systém kladeny požadavky vyjmenované níže.

2.1.1 Funkční požadavky

Mezi základní funkce systému patří následující:

- Využije extrakce faktů k nalezení relace, modality, podmětu, předmětu, příjemce, příslovečného určení způsobu, místa a času v každé větě zadaného textu, rovněž pak obsažené datové a časové údaje
- Poskytne elementární polaritu jednotlivých vět užitím analýzy sentimentu
- Umožní uživateli omezit výstup pouze na věty obsahující hledaná klíčová slova
- V případě pojmenovaných entit určí jejich kategorii
- Dokáže alespoň částečně zpracovat anafory (zpětné odkazy)
- Do vět doplní nevyjádřený podmět
- Na základě informací z předchozí věty dopočítá časové údaje
- Uživatel bude mít při práci s TextVis na výběr ze dvou podob výstupu – tabulkou a síťovým grafem

Výstup formou síťového grafu s sebou přináší tyto další funkční požadavky na systém:

- Dokáže propojit uzly představující datum v případě stejného roku, případně měsíce
- Dokáže slučovat uzly synonymních výrazů do jediného uzlu
- Uzly příbuzných výrazů propojí zvláštní hranou
- Dokáže slučovat uzly výrazů významově blízkých na základě vyhodnocení nejbližších sousedů dle natrénovaného modelu programu fastText. Uživatel bude moci zároveň ovlivnit míru tohoto slučování
- Umožní uživateli graf exportovat formou obrázku ve formátu SVG
- Umožní uživateli uzly grafu před exportem dle libosti přemisťovat po plátně

2.1.2 Nefunkční požadavky

Na systém byly kladeny také některé nefunkční požadavky, jmenovitě:

- Bude naprogramovaný v jazyce Python verze 3
- Bude použitelný jak v konzolové, tak i ve webové aplikaci a tyto budou nabízet shodnou funkcionalitu
- Využije nástroje vyvinuté v rámci Centra zpracování přirozeného jazyka FI MU
- Případné nekompatibilní nástroje budou upraveny tak, aby byly funkční s jazykem Python verze 3
- Webová aplikace využije pro zobrazení síťového grafu knihovnu `D3.js`
- Kód implementovaného systému bude pečlivě zdokumentovaný pomocí komentářů
- Systém bude generovat záznamy (logy) o své činnosti
- V případě libovolného validního vstupu musí být výstupem alespoň nějaká vizualizace, třebaže nepřesná[1]
- V případě nevalidního vstupu bude uživatel upozorněn

1. Nepřesnou vizualizací se myslí chybně detekované entity, vztahy mezi nimi, případně nevhodně určená polarita textu a podobně.

2.2 Použité nástroje

TextVis při extrakci dílčích druhů informací využívá jak nástrojů, které byly z velké části vyvinuty v Centru zpracování přirozeného jazyka při Fakultě informatiky Masarykovy univerzity, tak i některé externí programy a knihovny. Získaná data poté agreguje a dále zpracovává, aby je bylo možné použít při tvorbě požadované vizualizace. Tato sekce každý nástroj využívaný systémem TextVis krátce představuje.

2.2.1 Unitok

Ač by se proces tokenizace mohl na první pohled zdát primitivním, není tomu tak vždy – existuje spousta výjimek a jevů specifických pro ten který jazyk. Pro jazyk český problém tokenizace spolehlivě řeší nástroj **Unitok** [48]. Ten je dostupný pod licencí GPL a zadaný text rozdělí na tokeny, tedy souvislé řetězce znaků až po následující bílý znak.

Implementace je realizována v jazyce Python. Text je nejprve transformován do kódování Unicode a normalizován, tedy dochází k nahrazení některých řídících znaků mezerami a odstranění jiných. Následuje samotná tokenizace, při níž jsou hojně využity regulární výrazy.

Výstupem tohoto nástroje je takzvaný vertikál, tedy textový soubor v původním kódování, obsahující všechny nalezené tokeny s tím, že každý je umístěn na samostatném řádku a skupiny tokenů jsou odděleny XML značkami podle příslušnosti k větám a frázím.

2.2.2 Majka

Majka [49] je nástupcem nástroje *Ajka* vyvinutého na Fakultě informatiky Masaykovy univerzity. *Ajka* je morfologickým analyzátorem pro češtinu, který své využití našel nejen v Informačním systému MU, ale i u dalších českých klientů, například u českého internetového giganta Seznam.cz.

Stejně jako v případě jejího předchůdce je posláním Majky morfologická (tvarotvorná) analýza slov, při které jsou pro každé slovo poskytnuta možná lemmata,[2] stejně jako morfologické značky, popisující mluvnické kategorie daného slova.

2. Základní tvar slova.

Tohoto je dosaženo použitím rozsáhlé databáze párů vstupů a odpovídajících výstupů – lemmat a značek. Tímto přístupem tak Majka navazuje na úspěchy svého předka, s dodatečnou výhodou mnohonásobně zvýšené rychlosti analýzy.

Jelikož může pro jeden výraz existovat více možných lemmat a morfologických značek, následuje morfologická desambiguace pomocí Desamb [50], kdy je dle kontextu zvolen nejvhodnější pár, tvořen jedním základním tvarem a jednou značkou.

2.2.3 SET

Syntaktický analyzátor SET [51] vznikl v rámci diplomové práce Vojtěcha Kováře. Věty v jazyce českém zpracovává pomocí metody takzvané postupné segmentace věty podle nalezených klíčových bodů, kdy samotnou syntaktickou analýzu zajišťují manuálně vytvořená pravidla, která jsou uchována v textovém souboru a je tudíž možno do systému jednoduše doplňovat pravidla další.

Tato pravidla jsem pro účely systému TextVis rozšířil tak, aby detekovala časové a datové údaje v textu. Mimo to jsem zdrojový kód analyzátoru upravil, aby vyhovoval standardům jazyka Python verze 3.6.

Vzhledem k tomu, že některá pravidla mohou záviset na morfologických vlastnostech jednotlivých slov, využívá SET interně výše uvedený analyzátor Majka.

Výstup programu SET tvoří nalezené slovesné, jmenné a další fráze, včetně jejich vzájemných závislostí. Generování výstupu je možné v režimu grafovém nebo textovém, přičemž druhý jmenovaný výstup následně využijeme v modulu EFA pro rozhodnutí, jakou roli tyto fráze ve větě zastávají, viz část 2.2.4. Tentýž výstup zároveň poslouží pro zpracování anafor nástrojem Aara, jehož popis nalezneme v sekci 2.2.6.

2.2.4 EFA

Pod pojmem **EFA** [52], čili zkratkovým slovem odvozeným od názvu **E**xtraktor **FA**ktů, se skrývá modul, který byl vytvořen jako součást analyzátoru SET. Jeho úkolem je ve vstupním textu v českém jazyce pro každou větu určit, jaký větný člen představují fráze v ní vystu-

pující. Uživatel se tak dozvídá, která fráze představuje podmět, která přísudek, předmět a tak dále.

K tomuto účelu EFA využívá pravidla, která s pomocí značky poskytnuté morfologickou analýzou provedenou nástrojem Majka a syntaktické analýzy textu programem SET rozhodnou o výsledku. V případě předložkových frází je pak také využita česká lexikální databáze Český WordNet [53].

Kromě úpravy pro Python verze 3.6 jsem v tomto modulu pro účely použití v TextVis opravil neduh, kdy si EFA mezivýsledky ukládala vždy do stejného dočasného souboru. To mohlo mít za následek, že při použití dvěma uživateli naráz si tito lidé soubor navzájem přepsali a jeden z nich proto na výstupu dostal nesprávný výsledek. Po opravě je při každém běhu programu vygenerován nový soubor s vlastním unikátním identifikátorem, aby tento problém nastat nemohl.

Funkcionalitu tohoto modulu jsem navíc rozšířil o detekci časových údajů a dat, stejně jako příslovečného určení času a místa, které není součástí žádné předložkové fráze (např. *včera*).[3]

2.2.5 jusText

jusText [54] označuje modul, za jehož zrodem stojí disertační práce Jana Pomikálka z CZPJ FI MU. Tento software s otevřeným kódem se zaměřuje na odstranění duplikátů, značkování, reklam, navigačních menu a dalších částí zadaného HTML dokumentu, které nepřináší žádné nové informace.

Tohoto je dosaženo použitím heuristické metody, kdy jusText pro každý stavební blok stránky shromažďuje jeho délku ve znacích, počet odkazů, popřípadě počet určitých slov. Na základě naměřených hodnot pak každý blok klasifikuje jako jednoznačně nepotřebný – *bad*, informačně přínosný – *good*, příliš krátký pro poskytnutí jednoznačného úsudku – *short*, popřípadě pravděpodobně informačně přínosný, ale krátký – *near-good*.

Uživatel se tak může sám rozhodnout, které z těchto bloků si přeje v dokumentu ponechat a které nikoliv. Navíc je možné si i libovolně přenastavit hraniční hodnoty pro jednotlivé výše uvedené kategorie a tím klasifikační algoritmus přizpůsobit svému zamýšlenému účelu.

3. V původní implementaci EFA tyto sice detekuje, ale nepřesně jako příslovečné určení způsobu.

Další možností přizpůsobení je volba, zda v dokumentu ponechat nadpisy či nikoliv.

Vzhledem k tomu, že se jedná o otevřený software, kód programu jusText je komukoliv přístupný ve veřejné digitální knihovně LINDAT [55].

2.2.6 Aara

Modul **Aara** [56] je jedním ze skupiny modulů, které jsou součástí projektu Watsonson, který vznikl v rámci disertační práce Zuzany Nevěřilové na Fakultě informatiky MU. Úlohou tohoto modulu je nahrazení zpětných odkazů (anafor) vyjádřených pomocí zájmen za odkazovanou entitu a doplnění nevyjádřených podmětů v českých textech.

Modul k tomuto účelu využívá výsledků syntaktické i morfologické analýzy, kdy jsou anafory nahrazovány podle toho, s kterou entitou byla rozpoznána shoda v rodě i čísle. Ukázka fungování tohoto modulu je pak dostupná také jako online demo.[4]

Jako některé ostatní, také modul Aara jsem upravil tak, aby fungoval s jazykem Python verze 3.6.

Systém TextVis nabízí možnost spuštění v režimu, kdy modulu Aara využívá k nahrazení anafor odkazujících na předchozí větu. Uživatel tedy této funkcionality využít může, ale zároveň nemusí.

2.2.7 Derivancze

Derivancze [57] je modulem, který podává informace o derivačním vztahu mezi dvěma slovy (například *informace – informovaný*). Dokáže tedy poskytnout k zadanému slovu v českém jazyce všechna slova odvozená či naopak slovo, od kterého bylo zadané slovo odvozeno. Kromě směru odvození podává taktéž informaci o jeho druhu. Tedy kupříkladu, že v případě dvojice *otcův–otec* se jedná o přivlastňovací zájmeno odvezené od podstatného jména.

Podobně jako v případě morfologického analyzátoru Majka, i zde implementace staví na rozsáhlé databázi dvojic vstup–výstup.

V systému TextVis znalost derivačních vztahů mezi slovy získanou tímto nástrojem používám jak k propojování příbuzných výrazů, tak

4. `https://nlp.fi.muni.cz/projekty/watsonson/aara`

ke slučování slov se stejným významem, která se liší pouze způsobem zapsání, jako je tomu dejme tomu v případě dvojice slov *komunizmus–komunismus*. Tento proces pak blíže popisuji v sekci 3.4.

2.2.8 Propaganda Sentiment Analysis

Propaganda [58] je projekt, který vznikl za spolupráce programátorů z Centra zpracování přirozeného jazyka Fakulty informatiky MU a odborníků z Fakulty sociálních studií MU, jehož původním záměrem je detekce propagandy a manipulativních technik v českých textech.

Součástí projektu je ovšem řada dalších dílčích funkcí, mezi které patří například rozhodnutí, zda je v textu přítomna citace zdrojů, zda se text zaměřuje na domácí či zahraniční dění či detekce některých společenských témat jako je migrační krize a podobně.

Jednou z těchto funkcí je také jednoduchá analýza polarity textu na úrovni pozitivní, neutrální, negativní [59]. Tato funkce je k vyzkoušení veřejně dostupná[5] a je to právě ona, kterou TextVis využívá k základní analýze sentimentu.

2.2.9 NER a multiNER

Vyvinutý systém TextVis používá pro detekci pojmenovaných entit v textu **multiNER**, který k tomuto účelu využívá strojového učení a je schopen v českém textu rozpoznat entity, jejichž typ odpovídá osobám, lokacím, názvům produktů a několika dalším.

Pod pojmem multiNER se ve skutečnosti skrývá systém s otevřeným kódem Stanford NER [60] vyvinutý na univerzitě v americkém Stanfordu a implementovaný v jazyce Java. Natrénovaním Stanford NER na české datové sadě CNEC potom vznikl právě extraktor multiNER. K jeho spuštění z prostředí jazyka Python slouží skript *nerclient*.

Fakulta informatiky Masarykovy univerzity má obdobný, ovšem propracovanější vlastní modul jazyka Python pro rozpoznávání pojmenovaných entit. V nynější podobě se jedná již o jeho druhou generaci, jež je příhodně nazvána **NER v2** [61, 62]. Ta je schopna detekovat některé typy entit shodné s typy multiNER uvedené výše. Konkrétně se jedná například o názvy knih a organizací a některé další.

5. `https://nlp.fi.muni.cz/projekty/propaganda/sentiment/`

Kromě toho však identifikuje také e-mailové adresy, IP adresy, telefonní čísla, zkratky,[6] souřadnice, procentuální číselné údaje a řadu dalších. Každý typ má na starosti některý z osmi takzvaných *detektorů*, přičemž jeden detektor obstarává detekci jednoho nebo několika typů pojmenovaných entit. Uživatel má přitom možnost zvolit, které detektory se při extrakci použijí. Může tak ovlivnit, které typy pojmenovaných entit budou rozpoznávány a které nikoliv.

Systém TextVis pro detekování pojmenovaných entit používá oba zmíněné systémy v tandemu tak, že extrahované informace sjednocuje, jako by se jednalo o dvě množiny. Ve výstupu jsou tedy zahrnuty entity nalezené oběma programy, přičemž ty z nich, které byly nalezeny oběma dvěma systémy, jsou obsaženy pouze jednou. Pro NER v2 jsem současně zvolil konfiguraci všech detektorů vyjma detektoru zodpovědného za rozpoznání zkratek. Snažím se tak předejít zahlcení výstupu entitami s nízkou informační hodnotou.

2.2.10 VerbaLex

VerbaLex [63] je název lexikonu slovesných valencí, který vybudovali v roce 2005 odborníci z Fakulty informatiky MU, inspirovaní lexikonem VALLEX [64] z Univerzity Karlovy. Dali tak vzniknout slovníku, který je dostupný ve formátu XML a obsahuje ke každému slovesu kromě definice, anglických ekvivalentů a příkladů jeho použití také množinu jeho významů podle dat z Českého WordNetu a ke každému z těchto významů také synonymní slovesa a jejich varianty.

Právě významově seskupená synonymní slovesa jsou tím, čím je VerbaLex pro systém TextVis přínosem. Právě na tyto skupiny je totiž pohlíženo při slučování přísudkových uzlů. Proces slučování uzlů je blíže popsán v sekci 3.4.

Celkově obsahuje VerbaLex přes 20 000 sloves a jejich významů. Webové demo slovníku VerbaLex je ve zmenšené podobě veřejně dostupné online.[7]

6. Například Bc.
7. `http://nlp.fi.muni.cz/verbalex/htmlDEMO`

2.2.11 Declension

Nástroj **Declension** [65] je dalším z dílny Zuzany Nevěřilové z Fakulty informatiky MU, který dovoluje uživateli zadat české podstatné jméno, popřípadě jmennou frázi a tu transformuje do vyžádaného pádu a čísla.

Za tímto účelem používá Declension morfologický analyzátor Majka a pro zajištění co nejpřesnějšího výsledku také řadu ručně vytvořených pravidel.

Declension, podobně jako další projekty vytvořené v Centru zpracování přirozeného jazyka, nabízí svou funkcionalitu jak v podobě Python modulu, ale též formou webové aplikace.[8]

TextVis tento nástroj užívá k tomu, aby všechny fráze převedl do základního tvaru a odstranil nadbytečné předložky tam, kde je to žádoucí, například u určení času. Z předložkové fráze „*v 8:00*“ se tak ve výsledné vizualizaci stane pouze „*8:00*“.

2.2.12 fastText

fastText [66] je software vyvinutý společností Meta[9] za účelem efektivního zpracování přirozeného textu napříč jazyky.

Modelování slov ve fastText je inspirováno modelem skipgram [67]. Stejně jako on bere model programu fastText pro každé slovo v úvahu jeho kontext, tedy k slov v jeho bezprostředním okolí. Konstanta k je přitom určena parametrem *window size*, který implicitně odpovídá číslu 5.

Vnitřní reprezentace slov ve fastTextu je k tomu navíc upravena tak, že každé ze slov je představeno součtem vektorů jeho n-gramů. Uvažované n-gramy navíc odlišují prefixy a suffixy[10] od n-gramů uprostřed slova pomocí znaků $<$ pro označení začátku slova a $>$ pro označení konce slova. Například pro slovo *leden* a $n = 3$ by jeho n-gramy tvořily množinu s prvky $<le, led, ede, den$ a $en>$.

V tomto případě by se k těmto n-gramům navíc přidala speciální sekvence $<leden>$. V potaz se přitom pro každé slovo berou n-gramy velikosti $3 \leq n \leq 6$, přičemž hraniční hodnoty mohou být posunuty

8. `https://nlp.fi.muni.cz/projekty/declension/index.py`
9. Tato společnost byla původně známa pod názvem Facebook.
10. Předpony a přípony slova.

pomocí parametrů programu. Díky označení začátku a konce slova máme zajištěno, že 3-gram *den* pro slovo leden bude odlišný od sekvence <*den*> pro samotné slovo *den*. Tento přístup na jedné straně umožňuje natrénování nového modelu z neanotovaných dat v krátkém čase a zároveň dobře funguje pro jazyky s rozmanitou slovní zásobou a bohatým skloňováním.

Využitím vektorových reprezentací slov pak může být kupříkladu hledání analogií. Pokud programu fastText na vstup zadáme dvojici *Berlín – Německo* a požádáme jej o výstup pro slovo *Francie*, měli bychom na výstupu dostat *Paříž*.

Jinou, neméně užitečnou aplikací je nalezení nejbližších sousedů slova. Nejbližšími sousedy jsou ta slova, která se používají ve stejném kontextu. Pokud bychom tedy chtěli vyhledat nejbližší sousedy pro slovo *dívka*, odpovědí fastTextu by byla slova *žena*, *dívenka*, *školačka*, *holčička* a tak dále.

Pod licencí Creative Commons jsou pro fastText ke stažení volně dostupné embeddings[11] slov pro 157 jazyků, které byly natrénované na datech z webových stránek, která jsou dostupná v databázi Common Crawl[12] a na článcích z jednotlivých jazykových mutací Wikipedie[13] [68]. Mezi dostupnými modely nalezneme i ten pro češtinu, jenž je zužitkovaný při použití programu fastText systémem TextVis.

2.2.13 Data Driven Documents

Knihovna **Data Driven Documents** [69] pro jazyk JavaScript, známá také pod zkratkou `D3.js`, představuje nástroj pro snadnou tvorbu vizualizací z libovolných dat tím, že poskytuje rozhraní pro manipulaci těchto dat a jejich navázání na HTML dokumenty, obrázky ve formátu SVG či kaskádové styly. Uživatel tak může svá data prezentovat pomocí tabulek, koláčových grafů, liniových grafů, kvadrantových stromů či pomocí jakéhokoliv jiného vyjádření, kterého je možné dosáhnout za pomocí zmíněných technologií.

Jediné, co je k sestavení vizualizace zapotřebí, je, aby uživatel měl data, která chce zobrazit, dostupná ve strukturovaném formátu CSV, JSON nebo XML.

11. Vektorové reprezentace.
12. `https://commoncrawl.org/`
13. `https://www.wikipedia.org/`

Funkce knihovny jsou dále rozděleny do modulů dle jejich zaměření na d3-array pro práci s poli, d3-color pro práci s barvami a tak dále.

V systému TextVis je knihovna `D3.js` využita při generování výstupu síťovým grafem ve formátu SVG, stejně jako pro tvorbu výstupní HTML tabulky ve webové aplikaci, jež je popsána v sekci 3.6. Pro předání extrahovaných informací byl zvolen formát JSON. Kromě základního d3-selection a d3-force pro tvorbu síťového grafu byl uplatněn také modul d3-drag, který uživateli umožňuje přesouvání uzlů vygenerovaného grafu.

Autorem knihovny, která je i v roce 2022 stále aktivně vyvíjena a rozšiřována, je Mike Bostock. Nejnovější dostupnou verzí v době psaní práce je D3 v7.4.4, vydaná 11. dubna 2022, proto je v systému TextVis použita právě tato.

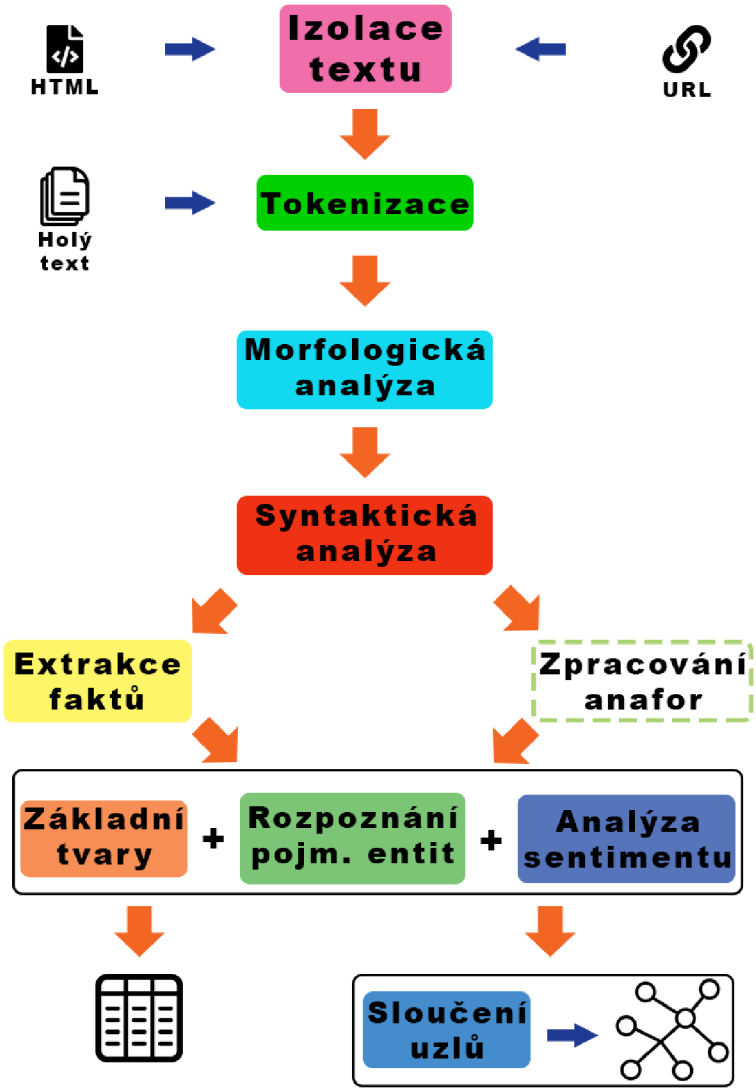
2.3 Systém TextVis

Systém TextVis, vzniknuvší v rámci této práce, v libovolném nestrukturovaném textu, případně vícero zadaných textech napsaných v českém jazyce identifikuje entity pojmenované i nepojmenované,[14] relace mezi těmito entitami, stejně jako polaritu jednotlivých vět. U pojmenovaných entit navíc určí jejich kategorii (osoba, lokace a tak dále). Všechny tyto získané informace následně převede do jediné vizualizace, která může nabývat podobu přehledné tabulky nebo síťového grafu na základě volby uživatele.

Aby bylo možné výše uvedené splnit, dochází nejprve k izolaci textu ve všech vstupních souborech. Získané texty je zapotřebí před jakýmkoliv dalším zpracováním také tokenizovat. Tokenizované texty poté podléhají morfologické a rovněž syntaktické analýze. Informace z obou analýz jsou dostačující k provedení extrakce faktů, to znamená k identifikaci podmětu, přísudku, příslovečného určení a tak podobně. Tytéž informace potom poslouží také ke zpracování anafor, o které uživatel může volitelně požádat. Následuje pro každý výraz jeho převedení do základního tvaru. Spolu s tím proběhne ověření, zda se jedná o pojmenovanou entitu či nikoliv. V případě, že ano, je zjištěna její kategorie. Na základě použitých výrazů je také ohodno-

14. Nepojmenovanou entitou je například *auto*.

cena polarita každé věty a to na ternární stupnici hodnotami pozitivní, neutrální, negativní. V případě zvolení výstupu síťovým grafem jsou synonymní uzly sloučeny, aby nedocházelo k přílišnému roztříštění grafu. Celý proces vytvoření vizualizace systémem TextVis krok po kroku zachycuje schéma 2.1. Konkrétními postupy a realizací každého kroku se následně podrobněji zabývám v kapitole 3.



Obrázek 2.1: Schéma procesu zpracování uživatelova vstupu k vytvoření vizualizace systémem TextVis.

3 Implementace

3.1 Zpracování vstupu

Na vstupu očekává systém TextVis holý text, textový soubor, popřípadě webovou stránku v podobě URL odkazu nebo HTML souboru. Pokud je vstupem webová stránka, je z ní textový obsah nejprve izolován za použití nástroje jusText s argumentem *no_headings* pro dodatečné odstranění nadpisů.

TextVis navíc podporuje zadání několika textových vstupů naráz. Ve výsledné vizualizaci vystupují všechny vstupní texty bez rozdílu.[1] Smyslem možnosti zadání vícero vstupů je umožnění propojení souvisejících entit v různých textech.

Každý ze vstupních textů je nejprve tokenizován nástrojem Unitok. Morfologickou analýzu obstarává analyzátor Majka, na tu potom navazuje syntaktická analýza, kterou má na starost SET. Pokud si uživatel přál, aby byly zpracovány také anafory, je tak učiněno užitím nástroje Aara. V takovém případě jsou navíc všechna individua očíslována (I1, I2, . . .), aby bylo jasné, ve kterém případě se jedná o již dříve zmíněnou entitu a kdy se v textu vyskytuje entita nová, avšak se stejným jménem. Některé anafory jsou zpracovány nehledě na toto nastavení. Je tomu tak ovšem pouze v případě vět, ve kterých bylo nalezeno osamocené adjektivum. V takovém případě se jej TextVis pokusí doplnit o poslední substantivum předchozí věty, které se shoduje v rodě a čísle. Příkladem takové situace je dvojice vět „Roman si před lety koupil značkový mobil. Teď už bude nejspíš bezcenný.“ Ve druhé větě se přídavné jméno *bezcenný* doplní tak, že celou entitou bude *bezcenný značkový mobil*.

Uživatel má možnost požádat o omezení výstupu pouze na některé věty a to tak, že definuje vyhledávané výrazy. Pokud tak učiní, ze vstupních textů jsou vypuštěny všechny věty, které žádný z těchto výrazů neobsahují.

Výsledky všech provedených analýz jsou následně předány dalším nástrojům, které provedou samotnou extrakci informací. Tento proces popisuji v následující sekci.

---

1. Kromě barvy v síťovém grafu, viz sekce 3.4.

3.2 Agregace informací

Výsledky syntaktické a morfologické analýzy jsou předány modulu EFA, aby rozhodl, které větné celky hrají v textu jakou roli. Je to právě zde, kde se rozhoduje o tom, která část každé věty je její přísudek, popřípadě příslovečné určení způsobu, eventuálně zda je předmětná část věty příjemcem (tj. můžeme se na něj zeptat otázkou *komu/čemu?*) či nikoliv. Jasno také dostáváme v tom, zda je určitý konstituent příslovečným určením místa nebo spíše času.

Každý výraz je pomocí nástroje Declension převeden do svého základního tvaru. Tyto základní tvary jsou v případě výstupu grafem užity pro slučování uzlů, dále viz sekce 3.4.

TextVis tyto získané informace dále rozvine – pro každý termín zjistí jeho příbuzné výrazy za pomocí databáze Derivancze. Kromě toho zkontroluje, zda se jedná o pojmenovanou entitu či nikoliv. V případě, že ano, z dat nástroje NER zároveň získá její kategorii. Zkoumání polarity každé věty je zajištěno pomocí analýzy sentimentu dostupné v rámci projektu Propaganda.

Pokud je jako způsob výstupu zvolen graf, potom jsou pro potřeby slučování uzlů zjištěna taktéž synonyma pro každou relaci dle dat slovníku VerbaLex. Jestliže se uživatel rozhodl také pro slučování podobných výrazů, jsou tyto obstarány s použitím programu fastText nalezením nejbližších sousedů slova.

Nakonec jsou u každého výrazu jeho gramatické kategorie, interně reprezentované jako značky, převedeny na termíny v přirozeném jazyce (např. značka *k5* odpovídá slovnímu druhu slovesa, *nS* zase jednotnému číslu a tak podobně). Tyto gramatické kategorie jsou pro každý termín uživateli ve webové aplikaci taktéž k dispozici k nahlédnutí.

Všechna tato data jsou agregována tak, aby bylo možné je zobrazit v tabulce, popřípadě síťovém grafu. O tom, jak jsou zjištěné informace uživateli prezentovány, pojednávají sekce 3.3 Výstup tabulkou a 3.4 Síťový graf.

3.3 Výstup tabulkou

Základní podobou, ve které jsou uživateli extrahované informace systémem TextVis podávány, je tabulka. Každý řádek tabulky odpovídá jedné větě v textu. Jednotlivé sloupce pak odpovídají entitám, které jsou rozděleny do kategorií dle jejich postavení ve větě a to konkrétně na *podmět*, *předmět*, *příjemce*, příslovečné určení *času*, *místa* a *způsobu*. Jeden ze sloupců tabulky vždy reprezentuje *relaci*, která tyto entity propojuje vztahem daným právě touto relací. Relace v drtivé většině případů odpovídá přísudku dané věty. V případě, že je relace uvedena s modálním slovesem, je toto zahrnuto ve sloupci *modalita*. Pokud se některá z těchto informací v celém textu nevyskytuje, je vynechán celý sloupec, aby byla zachována přehlednost tabulky.

Jestliže se v některé buňce tabulky nachází pojmenovaná entita, je tato buňka označena ikonou odpovídající typu pojmenované entity (postava pro typ *osoba*, kalendář pro typ *datum* a podobně). Tyto ikony pochází z volně dostupné kolekce FontAwesome.[2]

RELACE	PODMĚT	PŘEDMĚT	ČAS	MÍSTO	POLARITA
vzít	Rodiče (I2)	dcera (I1)	narozeniny	do Paříže	
být šťastná	Eva (I3)				😀
přát si	Eva (I3)	to (I4)			😀

Obrázek 3.1: Tabulka vygenerovaná webovou aplikací TextVis pro text „Rodiče vzali na narozeniny dceru do Paříže. Eva je šťastná, to si přála.“ při povoleném zpracování anafor

V tabulce nalezneme pro každou větu také ohodnocení její polarity. Tato hodnota je vyjádřena pomocí emodži[3] a to konkrétně jako 😀 v případě pozitivní polarity a ☹️ v případě polarity negativní. Pro větu s neutrální polaritou není zobrazen žádný piktogram. Na obrázku 3.1 je vyobrazena ukázka tabulky vygenerované systémem TextVis.

2. `https://fontawesome.com/`
3. Piktogram, většinou zobrazující obličej.

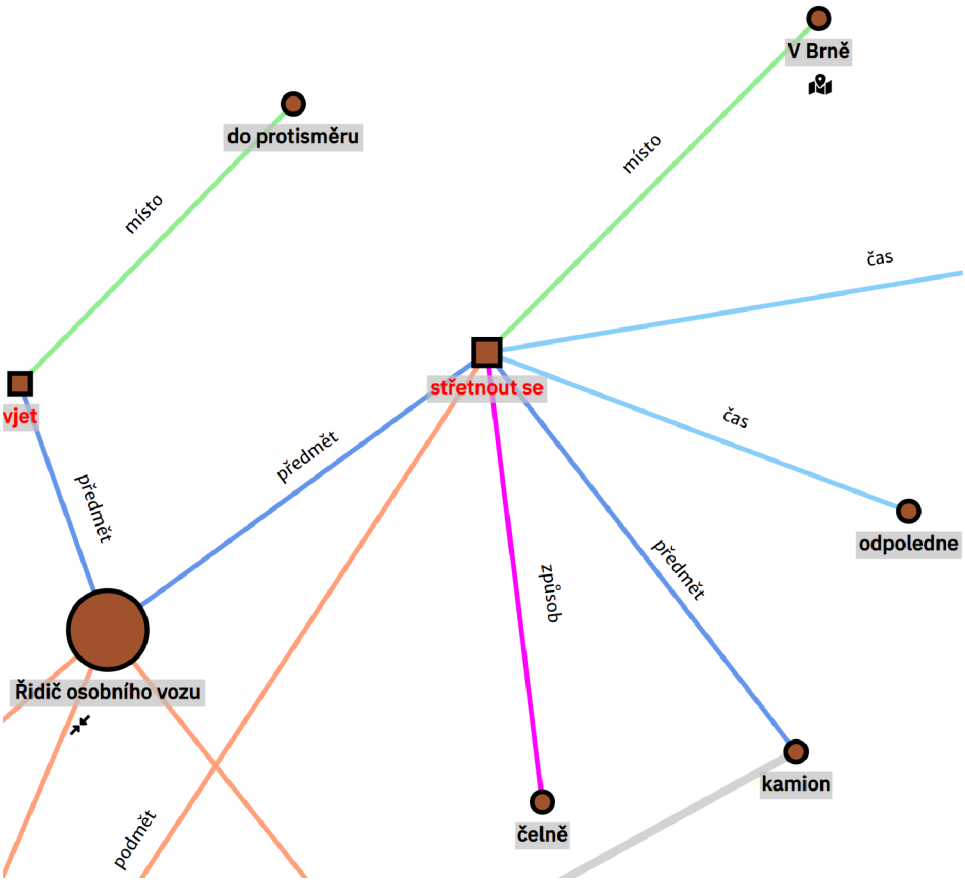
V případě, že uživatel zatouží po zajímavější vizualizaci, může si zvolit výstup síťovým grafem. Bližšímu představení této formy výstupu se dále věnuji v sekci 3.4.

3.4 Síťový graf

Při zvolení tohoto způsobu výstupu je sestrojen síťový graf s pomocí knihovny `D3.js`. Uzly tohoto grafu odpovídají buňkám tabulky v tabulkovém zobrazení, tedy relacím a entitám uvedeným v textu. Zatímco entity jsou vyobrazeny jako kolečka, relace představují čtverce. Název každé entity a relace je uveden pod uzlem, který jí náleží. Názvy relací jsou navíc zvýrazněny červenou barvou. Za předpokladu, že byl na vstupu zadán více než jeden text, jsou uzly každého dílčího vstupního souboru odlišeny různou barvou uvnitř uzlu. V opačném případě mají všechny uzly stejnou barvu.

Každá věta je pak reprezentována podgrafem, jehož středem je relace v této větě obsažená. Hrany spojují jednotlivé relace s entitami a dalšími výrazy, které jsou obsažené ve stejné větě jako daná relace. Na hranách je navíc označeno, zda se vzhledem k dané relaci jedná o podmět, předmět nebo jiný větný člen, přičemž každý větný člen se pyšní jinou barvou hrany a pro přehlednost je druh větného členu na hraně uveden také textem. Text zobrazený na hranách je však možné skrýt stisknutím tlačítka, které je k tomu určené. Podobně jako v případě tabulky je u uzlu, který náleží pojmenované entitě, její druh naznačen piktogramem. Příklad grafového výstupu je pro lepší představu dostupný na obrázku 3.2.

Graf je uživateli prezentován formou náhledu na webové stránce. Uživatel se přitom může po celé ploše plátna pohybovat kliknutím na prázdné místo a pohybem myši do libovolného směru. Zde si také uživatel může graf stáhnout na svůj stroj ve vektorovém formátu SVG a spolu s grafem zároveň na boční části obrazovky vidí i vstupní text včetně polarity jednotlivých vět, podobně jako je tomu při výstupu tabulkou. Výhoda tohoto dvojího zobrazení bude patrná záhy.

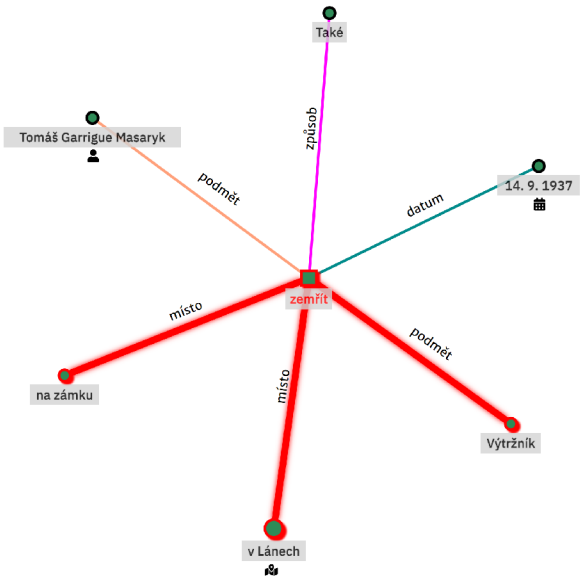


Obrázek 3.2: Část síťového grafu sestrojeného systémem TextVis pro článek o autonehodě ze serveru Brněnská Drbna

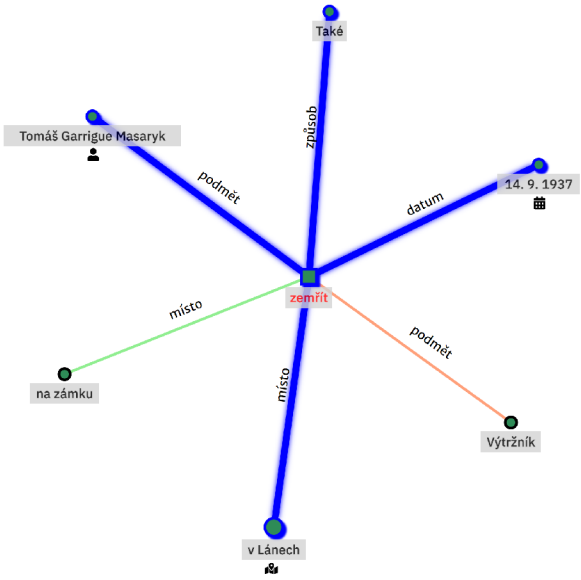
3.4.1 Navigace v grafu

Po umístění kurzoru myši na libovolný uzel grafu se totiž barevně zvýrazní celý podgraf, reprezentující větu, do které tento uzel patří. Stejně tak se navíc v bočním náhledu se zadaným textovým vstupem touto barvou zvýrazní věta samotná. To může být užitečné v případě, že mají dvě věty společný uzel, obzvlášť jedná-li se o uzel relace. Ukázku takového barevného rozlišení vidíme na porovnání 3.3.

Pokud je kurzor umístěn na uzel sdílený více větami, pak je každá věta zvýrazněna vlastní barvou. Avšak uzel, na kterém je kurzor umístěn, je vždy zvýrazněn barvou pouze jednou. Je tomu tak z důvodu, že



(a) Červené zvýraznění odpovídající věty při umístění kurzoru na uzel *Výtržník*



(b) Modré zvýraznění odpovídající věty při umístění kurzoru na uzel *Tomáš Garrigue Masaryk*

Obrázek 3.3: Porovnání barevného zvýraznění vět v grafu pro text „Výtržník zemřel na zámku v Lánech. Také Tomáš Garrigue Masaryk zemřel 14. 9. 1937 v Lánech.“

není možné adekvátně zvýraznit uzel vícero barvami naráz, a i kdyby bylo, není možné zajistit, že uzel nebude patřit do více vět než kolik je rozlišitelných barev, neboť zadaný text sám může alespoň hypoteticky obsahovat tolik vět, že jejich počet překročí počet člověkem odlišitelných barev. Celkově je v systému dostupných 63 různých barev pro zvýrazňování, poté se barvy začnou opakovat. Tyto barvy pochází z knihovny jazyka JavaScript, jQuery. Funkce zvýrazňování uzlů je dostupná pouze v náhledu grafu, po stažení SVG souboru tato funkcionalita dostupná není, neboť SVG soubor je pouze statický, nelze tedy zajistit reagování na události jako je pohyb kurzoru.

Kliknutím a potáhnutím myši může uživatel s uzly grafu libovolně pohybovat, ne však mimo hranice plátna. Udržení uzlu v hranicích plátna je zajištěno tak, že uživateli není dovoleno uzel posunout pod souřadnici 0 v případě obou os. V případě, že je uzel posunut příliš blízko k opačné hranici, je plátno rozšířeno tak, aby uzel zůstal v jeho hranicích. Stejně jako v předchozím případě je toto chování dostupné pouze před stažením grafu.

Další možností, jak může uživatel ovlivnit vzhled grafu, je přiblížení pomocí kolečka myši a zvětšení či zmenšení velikosti písma prostřednictvím tlačítek umístěných na okraji náhledového okna. Zde je také umístěno tlačítko, které všechny uživatelem provedené změny zahodí a to včetně přesunu jednotlivých uzlů. Graf je tím navrácen do původního stavu před provedením jakýchkoliv změn.

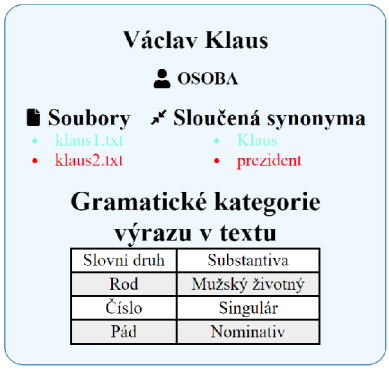
3.4.2 Obsah grafu

Jak již bylo řečeno, stěžejní komponentou grafu jsou uzly zastupující jednotlivé entity, propojené hranami skrze relace. Uzly synonymních výrazů jsou navíc slučovány dle jasně daných pravidel. Ke slučování sloves je kromě jejich základních tvarů použit také slovník slovesných valencí VerbaLex, díky kterému se slučují i slovesa se stejným významem o jiném základním tvaru, jako *utíkat* a *běhat*.

Ostatní slova jsou pak sloučena, pokud se jedná o naprostá synonyma (*Havel* a *Václav Havel*), popřípadě zdrobněliny (*Tomáš* a *Tomášek*) či odlišně zapsaná totožná slova (*komunizmus* a *komunismus*), dle dat derivačního analyzátoru Derivancze. Navíc si může uživatel zvolit, zda chce sloučit i slova, která jsou významově podobná (např. *pes* a *dobrman*). V takovém případě je podobnost slov vyhodnocena na základě

slovních vektorů pro český jazyk v nástroji fastText (viz sekce 2.2.12), pomocí kterého je vybráno n nejbližších sousedů každého výrazu. S těmito vybranými sousedy se pak daný výraz sloučí jako by se jednalo o synonyma. Původní model byl za účelem zrychlení systému zredukován na vektory o velikosti 50 (z předchozích 300). Může tak kvůli menší přesnosti modelu docházet k nechtěnému slučování. Aby se dopad tohoto neduhu zmírnil, má uživatel možnost ovlivnit, jak agresivní slučování bude a to tím, že si zvolí právě parametr n.

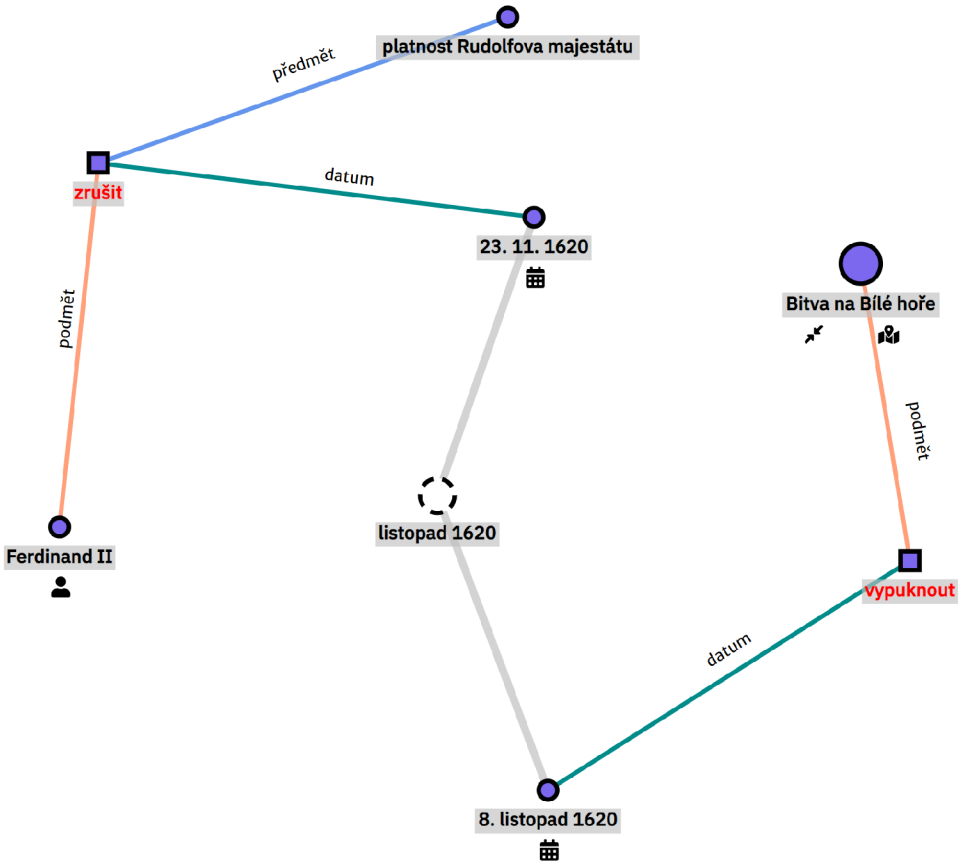
Aby bylo uživateli zřejmé, proč uzly, které zanikly při slučování, nemůže v grafu najít, jsou uzly, které vznikly sloučením jiných, označeny zvláštní ikonou – dvěma k sobě namířenými šipkami. S každým sloučeným výrazem uzel zároveň nabývá na velikosti. Po přesunu kurzoru na takový uzel se zobrazí informace o tom, sloučením kterých výrazů došlo ke vzniku tohoto uzlu. Uživatel zde taktéž nalezne seznam vstupních textů, ve kterých se tyto výrazy vyskytovaly. Na stejném místě jsou pak uvedeny také mluvnické kategorie náležící danému uzlu. Ukázku toho, jak vypadají informace předložené při podržení ukazatele myši nad libovolným uzlem, je možné vidět na obrázku 3.4.



Obrázek 3.4: Informace, které se zobrazí při umístění kurzoru na uzel *Václav Klaus* v grafu, který vznikl zadáním dvou souborů, přičemž soubor *klaus1.txt* obsahoval větu „Václav Klaus byl prezidentem do 7. března 2013.“ a soubor *klaus2.txt* obsahoval text „Klaus se netají láskou k lyžování. Jako prezident mimo jiné podepisoval zákony.“

Nakonec se v grafu zvláštní hranou propojují některé další výrazy. Jedná se zejména o výrazy, které jsou dle dat analyzátoru Derivancze

navzájem příbuzné, jako například *Kanada–Kanaďan*, ale také o data, která mají společný rok, měsíc nebo oba dva údaje. Tento jev je možné pozorovat na obrázku 3.5. K propojení pouze na základě shodného dne nedochází.



Obrázek 3.5: Část síťového grafu vygenerovaného pro text, který obsahuje věty „Bitva na Bílé hoře vypukla 8. listopadu 1620.“ a „Ferdinand II zrušil 23. 11. 1620 platnost Rudolfova majestátu.“. V grafu je patrné propojení uzlů *23. 11. 1620* a *8. listopad 1620* pomocí speciální hrany.

3.5 Konzolová aplikace

Konzolová aplikace se spouští přes skript `textvis.py`. Ten bere několik volitelných a dva povinné argumenty. Mezi povinné patří vždy rozhodnutí, zda má být vytvořena vizualizace tabulková (parametr `-t` či `--table`) nebo grafová (varianty `-g` a `--graph`). Tento parametr musí následovat buď text určený k analýze, případně může místo něj být uveden parametr `-f` či `--file` následovaný cestami k souborům, jejichž obsah má být analyzován, URL odkazy nebo kombinací obojího.

Za volitelné argumenty jsou pak považovány následující:

- `-a` pro povolení funkce zpracování anafor nástrojem Aara.
- `-s` nebo `--search` následovaný seznamem hledaných výrazů, jenž určí, které věty mají být ve výstupu ponechány. Věty jsou ponechány, pokud obsahují alespoň jeden hledaný výraz.
- `-o`, případně `--output_file` následovaný cestou k souboru, do kterého se má uložit výstup analýzy. Pokud není tento argument uveden, je použit standardní výstup.
- `-m` či `--merge_vector` následovaný kladným celým číslem pro sloučení n nejbližších sousedů slova. Počet sousedů n přitom určuje právě zadané číslo. Tento argument je dostupný pouze při výstupu grafem.

Na pořadí zadaných parametrů programu nezáleží. Na obrázku 3.6 vidíme použití systému z konzole a také jeho tabulkový výstup.

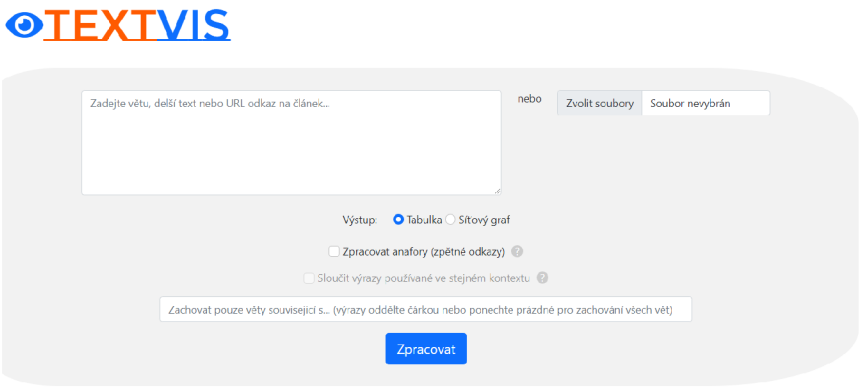
```
xmejzl13@aurora:/nlp/projekty/textvis/src$ ./textvis.py -t -f ../test/twitter-21.txt -a
MODALITA RELACE          PODMĚT       PŘEDMĚT            PŘÍJEMCE                          ČAS                  MÍSTO
-------- --------------- ------------ ------------------ --------------------------------- -------------------- -----------------
         dávat           oni (I2)     osvětový pořad (I1)                                  Dnes                 na Primě (LOKACE)
         ilustrovat      pacient (I3)
         být donucen ke  pacient (I6)                    zdrženlivost (I4), abstinence (I5)
mít se   začít           pacient (I6)                    svět (I7)                         relativně krátká doba
```

Obrázek 3.6: Tabulkový výstup systému TextVis pro text „Dnes na Primě dávali osvětový pořad z oblasti adiktologie, který dobře ilustroval, že pokud je pacient donucen ke zdrženlivosti až abstinenci, začne se mít po relativně krátké době docela k světu.“ Sloupec tabulky s určením způsobu, obsahující v druhém řádku slovo *dobře* a v posledním *docela* je na obrázku vynechán.

3.6 Webová aplikace

Součástí systému TextVis je webová aplikace,[4] která uživateli dovoluje veškerou funkcionalitu spustit odkudkoliv z pohodlí webového prohlížeče. Stejně jako v případě spouštění přes konzoli je k dispozici zobrazení výstupu v podobě tabulky nebo grafu. Zobrazení obou dvou způsobů výstupu v prohlížeči je realizováno pomocí knihovny `D3.js`. Dostupné jsou také veškerá nastavení a přepínače (pro vyžádání zpracování anafor a další), možnost zadání textu a URL nebo nahrání souboru. Webová aplikace tak dokonale kopíruje funkcionalitu konzolové aplikace ve webovém prostředí.

Webová aplikace je navíc jediným místem, kde je možné nahlédnout na vygenerovaný síťový graf. To znamená, že i v případě spuštění TextVis z konzole je uživatel přesměrován do webové aplikace. Úvodní stránku webové aplikace ukazuje obrázek 3.7.



Obrázek 3.7: Úvodní stránka webové prezentace systému TextVis, vyzývající uživatele k zadání vstupu.

4. `https://nlp.fi.muni.cz/projekty/textvis/`

4 Vyhodnocení úspěšnosti systému

Sekce 4.1 této kapitoly uvádí věty, popřípadě krátké texty, které posloužily pro ladění systému TextVis tak, aby na nich systém poskytoval uspokojivé výsledky. Jedná se tedy o reprezentativní vzorky toho, s jakými texty je TextVis připraven pracovat a jaké typy informací je v nich schopen identifikovat. Pozorovanou úspěšnost systému na skutečných textech dále přináší sekce 4.2.

4.1 Ladící sada

Pro každý druh extrahovaného údaje zde uvádím několik příkladů, na nichž byla laděna správná funkcionalita systému. Zvýrazněny jsou ty výrazy, které jsou v dané kategorii systémem identifikovány, případně na co se některé výrazy transformují, jedná-li se kupříkladu o vztažná zájmena. Kromě zde zmíněných příkladů byl systém laděn i na mnoha dalších textech.

Předpokládám, že jsem zvolil dostatečně vhodné příklady, aby systém obdobným způsobem pracoval i na textech se kterými se nikdy předtím nesetkal. Přesto lze očekávat jistou míru chybovosti, která by mohla být předmětem dalších úprav systému či nástrojů, které systém využívá.

- Relace
  1. Pes **běží** po chodníku.
  2. Rodiče **se domluvili** s Pepíkovou učitelkou na schůzce.
  3. David **je nemocný a smutný**.
- Modalita
  1. Karel **umí** opravit rozbitou pračku.
  2. Vašek **nechtěl** prohrát finále turnaje.
  3. Sandra **by mohla** nadělat dluhy.
- Podmět
  1. **Magda a Lenka** odjely na dovolenou.
  2. **Evropská iniciativa** zahájila sběr podpisů v září 2019.
  3. Při nehodě dvou vozidel byl **viník** pod vlivem alkoholu.

Kromě základní skladební dvojice musí systém zvládat také zpracování rozvíjejících větných členů. V kontextu předmětu věty systém

rozlišuje situace, kdy je předmět příjemcem a kdy nikoliv. Níže uvádím příklady právě takových extrahovaných údajů.

- Příjemce
  1. Marika donesla **rodičům** oběd.
  2. Vůdce promluvil k **lidem**.
  3. **Realu Madrid** se nepodařilo zvítězit.
- Předmět
  1. Babička upekla dětem **čokoládový dort**.
  2. Dominik se s **Láďou** od posledně nebaví.
  3. Tomáš nám vyprávěl o **svých prázdninách** v Itálii.

Mezi další extrahované údaje patří určení místa, způsobu a času.

- Určení místa
  1. Albrecht se přestěhoval a teď bydlí **u Brna**.
  2. Auto se vecpalo **mezi dva kamiony**.
  3. Pláchl **odtud** včas.
- Určení způsobu
  1. Závodníci kolem nás **rychle** prosvištěli.
  2. Pavel každé ráno běhá, aby něco dělal **pro svoje zdraví**.
  3. Renata rozhořčené matce **zcela úmyslně** neodpovídala.
- Určení času
  1. Porod začne **co nevidět**.
  2. Škola začíná v **8:00**.[1]
  3. Pavel sedí u počítače **od rána do večera**.

Speciální kategorii určení času představují data. Systém dokáže rozpoznat data hned v několika formátech. Zde uvádím jen několik příkladů. Propojování dat se společným rokem či měsícem v síťovém grafu přitom funguje napříč všemi rozpoznávanými formáty.

1. Útok na Dvojčata proběhl **11. 9. 2001**.
2. První světová válka vypukla **28. července 1914**.
3. Velký pád na burze nastal **roku 2008**.

Na následujících příkladech demonstruji propojení dat. U příkladu 1 dojde k propojení obou dat z důvodu shody v letopočtu, zatímco v příkladu 2 dojde k propojení vzhledem ke shodnému měsíci.

1. 28. října **1918** vznikl samostatný československý stát. V **roce 1918** byl také stržen Mariánský sloup.

1. Stejně tak by systém rozeznal čas obsahující sekundy.

2. Mikuláš a čert navštíví děti navečer 5. **prosince**. 31. **12**. dospělí oslaví příchod nového roku.

Datové i hodinové údaje mohou navíc v některých případech posloužit k vypočtení nových údajů. Přesně to ukazují následující příklady.

1. Alice začala vařit už v 6:00. **Po pěti hodinách** měla hotový oběd a husička byla vynikající.
   - Po pěti hodinách ≈ **11:00**
2. Inflace dosáhla v březnu 2022 jedenácti procent. **Před rokem** se přitom pohybovala kolem tří procent.
   - Před rokem ≈ **březen 2021**
3. Mnichovská dohoda byla přijata 30. 9. 1938. **O 6 měsíců později** už v Protektorátu rozhodovala Říše.
   - O 6 měsíců později ≈ **30. 3. 1939**

Systém se pokouší pro každou větu odvodit odpovídající polaritu na třístupňové stupnici. Zde uvádím příklady pro každý stupeň.

1. Maruška má ráda napínavé akční filmy.
   - **pozitivní** polarita věty
2. Františkův otec nesouhlasí se zavedením povinného očkování.
   - **negativní** polarita věty
3. Nejhezčí ženy se rodí v České republice.
   - **pozitivní** polarita věty
4. Olympijské hry se budou v roce 2024 konat v Paříži.
   - **neutrální** polarita věty

Pokud je detekována pojmenovaná entita, je rovněž určena její kategorie. Příklad 1 představuje větu, v níž systém odhalil jméno osoby, kdežto příklad 2 obsahuje entitu, jejíž typ systém rozpozná jako lokaci. Poslední příklad pak ukazuje entitu, ve které TextVis pozná adresu.

1. **Václav Havel** zemřel v neděli 18. prosince 2011 na následky dlouhodobých zdravotních problémů.
2. Za největší překážku rozvoje cyklistiky v **Praze** se uvádí nedostatek bezpečných tras.
3. Zájemci mohou životopisy posílat na **Kříženeckého náměstí 322/5**.

Některá zájmena se transformují na odkazovanou entitu. Osamocená přídavná jména se zase TextVis pokusí připojit k nejbližší entitě odpovídajícího rodu a čísla. Situaci, kdy systém k takovému kroku přistoupí, je možné vidět na příkladu 2 níže.

1. Běžel buvol kolem řeky, která zahnula doprava.
   - která ≈ **řeka**

2. Budova shořela 16. října 2015. O rok později stála nová.
   - nová ≈ **nová budova**
3. U Kulaté báby vám dnes jako předkrm nabízí jitrnici, jakou svět neviděl.
   - jakou ≈ **jitrnice**

TextVis používá nástroj Aara jednak ke zpracování anafor, které nejčastěji nabývají podoby ukazovacích zájmen, jednak také k odhalení nevyjádřeného podmětu věty. Oba jevy prezentuji na těchto příkladech.

1. David pokazil poslední pokus zkoušky, protože se pořádně neučil.
   - nevyjádřený podmět **on** ≈ **David**
2. Pepovi se líbí Eliška a včera **ji** pozval na rande.
   - nevyjádřený podmět **on** ≈ **Pepa**
   - ji ≈ **Eliška**
3. Klobásy jsou jako zákony, **ty** je také lepší nechat vidět.
   - ty ≈ **ty zákony**

Nevyjádřený podmět systém identifikuje i v případě, že uživatel program spustí bez volitelného zpracování anafor. Děje se tak na základě tvaru relace (přísudku), popřípadě modálního slovesa.

1. Běžela zpátky domů.
   - nevyjádřený podmět ≈ **ona**
2. Domluvili se na dalším postupu.
   - nevyjádřený podmět ≈ **oni**
3. V závodě chtěl vyhrát za každou cenu.
   - nevyjádřený podmět ≈ **on**

Důležitou funkcí systému je slučování synonymních uzlů grafu. V následujících příkladech tuto funkcionalitu demonstruji. Zvýrazněná slova přitom značí, které výrazy se sloučí do jediného uzlu. Příklad 2 ukazuje slučování na základě deminutiv pomocí dat nástroje Derivancze, zatímco příklad 3 předvádí slučování dle nejbližších sousedů slova nalezených programem fastText.

1. V čele státu stojí **Miloš Zeman**. **Zeman** je prvním prezidentem, který byl zvolený v přímé volbě.
2. **Malý Tomášek** dospěl a odstěhoval se od rodičů. **Tomáš** nyní pracuje v bankovním sektoru.
3. Katka na plese vyhrála **termolahev**. **Tuto termosku** teď s sebou nosí do školy každý den.

4.2 Úspěšnost se skutečnými texty

Pro účely testování systému TextVis jsem zvolil tři kategorie textů, které se na Internetu objevují nejčastěji. Jedná se o články zpravodajského charakteru, příspěvky na sociálních sítích a amatérské články z blogů. Pro každý text jsem pomocí systému TextVis vygeneroval tabulkový výstup s povoleným zpracováním anafor nástrojem Aara. Ruční kontrolou jsem poté ověřil správnost extrahovaných údajů. V následujících sekcích prezentuji zjištění plynoucí ze získaných dat v jednotlivých kategoriích a rovněž několik statistických údajů o zkoumaných textech.

Při kontrole výstupu programu jsem si zaznamenal každou chybu, které se systém dopustil a u každé z nich jsem se pokusil stanovit příčinu, tedy který z použitých nástrojů je za danou chybu zodpovědný.

U analyzátoru Majka jsem proto považoval za chybu, pokud určité slovo neměl vůbec v databázi nebo došlo při desambiguaci ke zvolení nevhodného lemmatu či značky.

Co se týče nástroje Unitok, ten penalizuji chybou, jestliže rozdělil větu, popřípadě slovo tam, kde to není vhodné. To se děje v některých případech u slov, která obsahují pomlčku, například označení stíhacích letounů *F-35* a podobně. Dále jsem toto chování pozoroval v případě příspěvků na sociálních sítích, kde se stává, že lidé větu neukončí interpunkčním znaménkem, nýbrž emotikonem. Věty v takovém případě nejsou správně rozděleny. To sice není zcela vinou nástroje Unitok, jako spíš „nesprávným“ chováním uživatelů, ale vzhledem k četnosti tohoto jevu by určitě stálo za zvážení doplnit funkcionalitu, která by takovou situaci řešila.

Analyzátor SET jsem považoval za chybující, když přiřadil nalezenou frázi k nesprávné větě. To se děje téměř výhradně u rozvitých souvětí o několika vedlejších větách. V této oblasti je jistě prostor pro zlepšení. Příkladem takové komplikovanější věty je „Ovšem problém je, pane Pešáne, nejspíš v tom, že nám se na vrcholných akcích zřejmě puky neodrážejí tak, jak bychom si přáli, už 10 let...“

Dalším problémem, se kterým jsem se u tohoto syntaktického analyzátoru setkal, bylo, že spojoval do jediného konstituentu slova, která k sobě nepatří. Větou demonstrující tento jev je „Najdete tu placené parkoviště za 50 Kč na den“, kdy dojde ke spojení výrazů do jediného konstituentu *tu placené parkoviště*, což znemožňuje nástroji

EFA správně označit *tu* jako určení místa a *placené parkoviště* jako předmět věty, neboť musí celý konstituent zařadit do jediné kategorie.

Posledním zdrojem chyb, který jsem připisoval na vrub tomuto nástroji je problém přesně opačný, a tedy nespojení všech tokenů, které být spojeny měly, do jednoho konstituentu. Týká se to výrazů, které v sobě skrývají předložku, kupříkladu „Státní úřad pro jadernou bezpečnost“ a mnohé jiné. Jiným nesprávně zvládnutým jevem je zacházení se slovesy v infinitivním tvaru, kdy SET nepřipouští jejich spojení s podstatným jménem do jednoho konstituentu kupříkladu ve výrazech „ztratit schopnost mluvit“ nebo „mít možnost falšovat volby“.

Při řešení zpětných odkazů pomocí nástroje Aara vznikají převážně dva problémy. Jedním je, že Aara může odkazovanou entitu hledat i v textu, který teprve následuje. To není samozřejmě vždy špatně, neboť se může stát, že specifický slovosled vět způsobí, že odkazovaný subjekt bude uveden až za zájmenem, které na něj odkazuje. Stává se to však velmi zřídka, a proto je toto chování spíše na škodu.

Modul EFA si čas od času plete příslovečné určení času a místa v předložkových frázích. Je to logické, neboť toto rozlišení nelze snadno popsat morfologickými ani syntaktickými vlastnostmi, ale vyžaduje porozumění kontextu. Proto by mohlo být vhodné extrakci faktů v budoucnosti rozšířit s využitím strojového učení. Podobným problémem extraktoru faktů je občasné nesprávné začlenění v kategoriích podmět a předmět. Další problém, který se zde vyskytuje, je nedostatečné pokrytí pravidly, navzdory jejich rozšíření v rámci této práce. Pravidla totiž nepokrývají některé výjimečné situace, jako je například podmět v podobě slovesa ve větě „Špička, kterou představují Kanada, USA, Rusko a obě severské země, nám již poněkud utekla a v našich silách sice je občas některý z těchto týmů **porazit**, ...“

V následujícím vyhodnocení označuji nástroje NER a multiNER souhrnně jako *NER*. U obou modulů jsou jediné jejich dvě možné chyby zřejmé – jedná se o neodhalení pojmenované entity anebo nesprávné určení její kategorie. Takovou chybou je například označení entity *vrch Sokol* jakožto organizace. Jedná se totiž o skalní útvar, který by bylo lépe označit jako lokaci. Tu si ale uvedené nástroje pletou s jistým tělovýchovným oddílem, a proto jejich rozhodnutí v takové situaci označuji jako chybné.

Vzhledem k úloze nástroje Propaganda v systému TextVis jej označuji za chybující pokaždé, kdy nesprávně určí sentiment věty a to jak falešně pozitivně (věta je ohodnocena jako negativní anebo pozitivní, i když její sentiment tomu neodpovídá), tak falešně negativně (věta je hodnocena jako neutrální, přestože je evidentně zabarvená).

Konečně chybou, kterou přisuzuji programu jusText je situace, kdy v zadaném textu ponechá části, které zjevně ponechávat nemá. Jsou to například názvy videí, ustanovení o autorských právech v zápatí stránky, náhled jiného článku než hlavního, ale také ponechání nadpisů, navzdory tomu, že si je systém TextVis explicitně žádá odstranit. Další potíží, se kterou jsem se u tohoto nástroje setkal, byla jeho špatná práce s kódováním, kdy nedokázal text v zadaném HTML souboru správně zpracovat. Stávalo se tak v případě kódování Windows-1250.

Pokud některý nástroj udělal chybu, která je ve své podstatě záměnou, nepovažoval jsem to za jedinou chybu, nýbrž za chyby dvě. Příkladem budiž situace, kdy modul EFA zamění podmět a předmět.

4.2.1 Zpravodajské články

Co se týče textů zpravodajského charakteru, typickým kamenem úrazu je bezesporu přímá řeč, kterou se ne vždy podaří správně tokenizovat, což má neblahý vliv na celkové výsledky analýzy. Tento problém se u jiných typů textů nevyskytuje. Mezi nejčastěji chybující nástroje dále patří SET, EFA a Aara. To je dáno složitostí souvětí, která se v těchto textech vyskytují a analyzátoru SET působí problémy. Důsledkem toho má software Aara často potíže se správným zpracováním anafor, kdy jsou čas od času doplňovány nevhodné nevyjádřené podměty či entity nehodící se do dané věty.

Největším problémem zdá se být právě zpracování zpětných odkazů, několikrát jsem totiž byl svědkem situace, kdy Aara doplnila do jedné z posledních vět jako nevyjádřený podmět některou entitu ze samotného začátku článku, což je samozřejmě nežádoucí. I kvůli takovým problémům závisí použití nástroje Aara při analýze textu systémem TextVis na rozhodnutí uživatele. U nejčastěji chybujícího modulu EFA pak dochází k občasné záměně některých větných členů, zejména co se týče určení času a místa, ale také podmětu a předmětu. Naproti tomu chybovost nástrojů Unitok, Majka a jusText je téměř mizivá.

Zdrojem pro texty zpravodajského charakteru mi byly portály iDNES,[2] Novinky,[3] Seznam Zprávy,[4] Drbna,[5] TN.cz,[6] Sport[7] a Blesk.[8] V tabulce 4.1 uvádím ke každému nástroji, který systém používá, konkrétní pozorovaný počet chyb v této kategorii textů, procentuálně také vyjadřuji, jakým poměrem se dotyčný nástroj podílel na všech chybách systému. Tabulka dále obsahuje také celkový počet nalezených chyb a testovaných vět.

Tabulka 4.1: Údaje získané při testování textů ze zpravodajských článků.

Testovaných textů	10	EFA	365 (34,11 %)
Testovaných vět	637	SET	213 (19,91 %)
Chyb celkem	1070	Aara	297 (27,76 %)
Chyb na větu	1,679	Declension	33 (3,08 %)
Unitok	17 (1,59 %)	NER	20 (1,87 %)
Majka	9 (0,84 %)	jusText	15 (1,40 %)
Propaganda Sentiment		101 (9,44 %)	

4.2.2 Příspěvky na sociálních sítích

Zdrojem textů pro tuto kategorii byly příspěvky lidí angažovaných v české politické scéně, jako například Markéta Pekarová Adamová, Petr Fiala, Tomio Okamura, Andrej Babiš nebo Jiří Ovčáček a některých ostatních na sociální síti Twitter a odpovědi na tyto příspěvky, dále pak nahodilé příspěvky na síti Facebook a také komentáře lidí na stejné platformě na profilech zpravodajských webů ČT24, TN.cz, iDNES a dalších. Čerpal jsem taktéž z komentářů pod fotografiemi nejsledovanějších celebrit na síti Instagram.

Potíží při analýze těchto textů jsou jevy, které systém nemůže předpovídat, jako jsou překlepy nebo nesprávná interpunkce. To činí pro-

2. https://www.idnes.cz/
3. https://www.novinky.cz/
4. https://www.seznamzpravy.cz/
5. https://www.drbna.cz/
6. https://tn.nova.cz/
7. https://www.sport.cz/
8. https://www.blesk.cz/

blémy zejména při desambiguaci, neboť kupříkladu slovo „Nesouhlasim“ psáno s krátkým písmenem „i“ je poté analyzátorem Majka klasifikováno jako podstatné jméno. Nesprávná interpunkce má potom vliv již při samotné tokenizaci textu. Zde působí problémy i emotikony, které jsou na sociálních sítích hojně využívány k rozdělení vět. Příkladem může být „Dobře, uvidíme se zítra 🙂 Budu se těšit.“. Nástroj Unitok však použití emotikonu namísto interpunkce neočekává a věty tak neoddělí. Nesprávná tokenizace má za následek také nesprávnou syntaktickou analýzu a další na ni navazující extrakce informací.

Dalším problémem v této kategorii je, že nástroj Propaganda neumí rozeznat ironii, která je na sociálních sítích také hojně používána a to způsobuje nesprávné rozeznání polarity textu. V této kategorii jsem nezapočítával žádnou chybu nástroji jusText, neb jeho použití není pro sociální sítě vhodné a jeho chybovost by tak byla stoprocentní. Namísto použití jusText jsem text z komentářů a příspěvků extrahoval do textového souboru ručně.

Tabulka 4.2: Nasbíraná data v kategorii sociálních sítí.

Testovaných textů	87	EFA	157 (23,72 %)
Testovaných vět	536	SET	195 (29,46 %)
Chyb celkem	662	Aara	196 (29,61 %)
Chyb na větu	1,235	Declension	21 (3,17 %)
Unitok	9 (1,36 %)	NER	10 (1,51 %)
Majka	20 (3,02 %)		
Propaganda Sentiment		54 (8,16 %)	

4.2.3 Blogy – amatérské články

Blogy lze vnímat jako střední cestu předchozích dvou typů textů. Autoři zpravidla nejsou profesionálové, ale své myšlenky se snaží rozvinout do složitějších struktur a souvětí. Oproti předchozím textům jsem zde pozoroval o něco větší chybovost nástroje pro zpracování anafor, Aara. Domnívám se, že důvodem může být poněkud chaotičtější stavba souvětí, která v těchto textech bývá chvílemi přítomna, spolu s větší tendencí autorů používat zpětné odkazy v podobě zájmen, než je tomu u žurnalistů.

Pro testování v této kategorii jsem texty čerpal především ze dvou nejpopulárnějších zdrojů v této kategorii, tedy Blog iDNES[9] a Blogy – Názory Aktuálně,[10] ale také z turistického webu Veruce.[11]

Tabulka 4.3: Pozorovaná chybovost systému při zpracování textů z blogů.

Testovaných textů	9	EFA	198 (27,09 %)
Testovaných vět	524	SET	144 (19,70 %)
Chyb celkem	731	Aara	249 (34,06 %)
Chyb na větu	1,395	Declension	14 (1,92 %)
Unitok	13 (1,78 %)	NER	25 (3,42 %)
Majka	10 (1,37 %)	jusText	12 (1,64 %)
Propaganda Sentiment		66 (9,03 %)	

Třebaže se chybovost systému na první pohled může jevit vysoká, je třeba pamatovat na nekompromisní metodiku při ověřování úspěšnosti, kterou jsem zvolil. Zejména tedy, že jako chybu jsem započítal jakýkoliv jev, který se lišil od ideálního vyhodnocení a že stejnou chybu na více místech jsem k celkovému počtu přičetl opakovaně.

Dalším faktorem, který přispívá k vysokému počtu pozorovaných chyb je jistě samotný počet údajů, které TextVis extrahuje. Kromě všech možných druhů pojmenovaných entit a určení vztahů mezi nimi, je to i polarita každé z vět. Každá odhalená entita navíc musí projít správným vyskloňováním a zařazením ke správnému větnému členu. Kromě toho by systém v textu měl rozpoznat zpětné odkazy. Každý z těchto dílčích kroků přitom může být zdrojem chyby.

Ačkoli je tak třeba konstatovat, že na systému je ještě mnoho co zlepšovat, jsem toho názoru, že vzhledem k modulárnímu charakteru architektury systému, kdy se jednotlivé používané nástroje dají snadno nahradit jinými, vylepšenými, se do budoucna naskýtá možnost jeho vlastnosti dále vylepšovat, přičemž vlastní schopnost vizualizovat výsledky analýz zůstane zachována. Tato vylepšení dílčích nástrojů by proto nemělo být složité do systému zakomponovat a mohou být předmětem budoucí práce Centra zpracování přirozeného jazyka.

9. `https://blog.idnes.cz/`
10. `https://blog.aktualne.cz/`
11. `https://blog.veruce.cz/`

4.3 Časová náročnost analýzy

Doba běhu analýzy, kterou TextVis provádí, se dohromady s následným sestavením vizualizace pohybuje od pár sekund po několik málo minut. Na rychlost analýzy má přitom vliv řada faktorů. Tím hlavním je samotná délka zadaného textu. Dalšími činiteli, které čas potřebný k dokončení extrakce informací a zhotovení výsledné vizualizace ovlivňují, jsou parametry, se kterými uživatel aplikaci spouští. Významným je totiž případný požadavek na odhalení zpětných odkazů nástrojem Aara. Při vyžádání této funkcionality se totiž značně prodlouží celý proces zpracování textu. Svůj podíl nese také druh vizualizace, kterou uživatel žádá. V případě výstupu grafem je totiž zapotřebí kromě samotné extrakce informací provést také sloučení uzlů a s ním spojené operace, což celkový čas běhu programu prodlouží.

Dalším provinilcem v tomto ohledu je program fastText, který se použije v případě výstupu grafem, pokud uživatel projeví zájem o slučování uzlů výrazů s podobným významem. Navzdory svému názvu totiž i fastText stráví touto náročnou operací poměrně dlouhou dobu. Tento časový úsek se však v mém pozorování neukázal být přímo ovlivněn tím, jaký počet nejbližších sousedů výrazu se má dle uživatelova přání sloučit. To jsem si ověřil tak, že jsem měření provedl s požadavkem na sloučení deseti nejbližších sousedů slova a poté jsem tento počet zvýšil na dvojnásobek. Celkový čas výpočtu se tím v zásadě nezměnil. Naopak větší počet sousedů ke sloučení vyšel na mnou zvolených textech v průměru na jednu větu dokonce jako mírně rychlejší varianta.

Pro otestování časové náročnosti systému jsem jej spouštěl s různými parametry i různě dlouhými texty. Délku běhu programu jsem pak měřil pomocí profileru[12] zabudovaného v jazyce Python, cProfile. Měření jsem dle použitých parametrů rozdělil do sedmi kategorií, a to na tabulkový výstup, tabulkový výstup spolu s žádostí o zpracování anafor, výstup síťovým grafem, grafový výstup s anaforami, výstup grafem s žádostí o sloučení deseti nejbližších sousedů slova, stejný případ s dvaceti sousedy a nakonec grafový výstup, kdy uživatel žádá jak zpracování zpětných odkazů, tak sloučení nejbližších sousedů slova, a to konkrétně opět deseti.

12. Nástroj pro analýzu výkonnosti programu.

Celkově jsem rychlost analýzy zkoumal na datové sadě 25 textů, kdy deset tvořily zpravodajské články, dalších deset příspěvky ze sociálních sítí Facebook a Twitter a posledních pět představovaly amatérské články z blogů. Nejdelší text čítal 82 vět, zatímco nejkratší pouhé 4 věty. Rozdílná délka analýzy u různých druhů textů je dána počtem vět, které se v daných typech textů typicky objevují.

Podle očekávání jsem svým pozorováním zjistil, že nejrychlejší je analýza v případě, že uživatel žádá pouhý výstup tabulkou bez dalších požadavků a naopak nejdelší zpracování si vyžádá výstup grafem při zpracování zpětných odkazů spolu se sloučením uzlů nejbližších sousedů výrazu. Tabulka 4.4 obsahuje pozorované průměrné délky zpracování pro každý druh textů a pro všechny kategorie vstupních parametrů taktéž průměrnou délku zpracování jedné věty.

Tabulka 4.4: Rychlost zpracování vybraných textů v různých kategoriích. Všechny uvedené číselné údaje reprezentují průměrnou délku běhu analýzy v dané kategorii v sekundách zaokrouhlenou na jedno desetinné místo.

Kategorie	Zpravodaj. články	Sociální sítě	Blogy	Jedna věta
Tabulka (bez anafor)	403	32,2	396,5	7,8
Tabulka (s anaforami)	617	48,5	640,9	12,1
Síťový graf (bez anafor)	883,2	53,4	1173,5	18,8
Síťový graf (s anaforami)	1110,5	73,6	1620,4	24,6
Síťový graf (10 sousedů[i])	1859,6	187,9	2305,1	39,4
Síťový graf (20 sousedů[i])	1901,35	184,4	2157,2	39
Síťový graf (10 sousedů[i] a anafory)	2204,6	208,6	2726,7	46,5

[i] Slučování na základě nejbližších sousedů slova

Závěr

Cílem této bakalářské práce bylo na jedné straně čtenáře seznámit s různými přístupy a technikami, které je možné použít v oblasti extrakce informací a rovněž poskytnout přehled skutečných systémů ve světě, které se touto problematikou zabývají. Konkrétně se práce zaměřovala na extrakci informací z nestrukturovaného textu. Na straně druhé bylo smyslem práce navrhnout a implementovat vlastní systém pro extrakci informací z volného textu v českém jazyce. Vyvinutý systém dostal název TextVis. Ten pro svoji činnost využívá nástroje pro extrakci faktů, rozpoznání pojmenovaných entit a základní analýzu polarity textu vytvořené v Centru zpracování přirozeného jazyka Fakulty informatiky Masarykovy univerzity. Získaná data TextVis agreguje a zobecňuje tak, aby mohl být výstup uživateli předán v podobě vizualizace, přičemž aplikace nabízí dva režimy zobrazení. Jedním je prezentace získaných dat ve formě přehledné tabulky. Dalším je pak vyobrazení v režimu interaktivního síťového grafu, kdy jsou nalezené entity a relace zobrazeny jako uzly, které se propojují hranami na základě nalezených vztahů. Systém nadto umožňuje uživatelům takový vygenerovaný graf exportovat do souboru vektorové grafiky. Pro ještě lepší výsledky systému bylo také zakomponováno volitelné zpracování anafor a vytvořen propracovaný modul pro slučování uzlů grafu, který kromě dalších nástrojů CZPJ FI MU využívá též program fastText. Mimo to jsem některé z uvedených nástrojů pro potřeby systému rozšířil o další funkcionalitu. Jako vedlejší produkt implementace jsem tak například dal vzniknout portům nástrojů SET a Aara pro novější verzi jazyka Python. Nakonec jsem provedl vyhodnocení úspěšnosti systému na skutečných textech o celkové délce 1697 vět. K tomuto účelu mi posloužily texty z internetových blogů, stejně jako články zpravodajského charakteru, ale i nahodile vybrané příspěvky uživatelů sociálních sítí. Pro každou z těchto kategorií textů jsem také zdůraznil nejslabší místa systému. Moje zjištění tak mohou posloužit při jeho dalším případném vývoji. Na vybraných textech jsem také porovnal oba způsoby výstupu z hlediska časové náročnosti analýzy při spuštění programu s různými parametry.

Bibliografie

1. SARAWAGI, Sunita. *Information extraction*. Now Publishers Inc, 2008.
2. GRISHMAN, Ralph. Information extraction: Techniques and challenges. In: *International summer school on information extraction*. 1997, s. 10–27.
3. JUNG, Keechul; KIM, Kwang In; JAIN, Anil K. Text information extraction in images and video: a survey. *Pattern recognition*. 2004, roč. 37, č. 5, s. 977–997.
4. VALENZA, Robin; ROBINSON, Tony; HICKEY, Marianne; TUCKER, Roger. Summarisation of spoken audio through information extraction. In: *ESCA Tutorial and Research Workshop (ETRW) on Accessing Information in Spoken Audio*. 1999.
5. CUNNINGHAM, Hamish. Information extraction, automatic. *Encyclopedia of language and linguistics*, 2005, roč. 3, č. 8, s. 10.
6. APPELT, Douglas E. Introduction to information extraction. *Ai Communications*. 1999, roč. 12, č. 3, s. 161–172.
7. TANG, Jie; HONG, Mingcai; ZHANG, Duo Liang; LI, Juanzi. Information extraction: Methodologies and applications. In: *Emerging Technologies of Text Mining: Techniques and Applications*. IGI Global, 2008, s. 1–33.
8. *What is Information Extraction?* [Online]. New York: Ontotext USA, Inc., 2021 [cit. 2021-10-20]. Dostupné z: `https://www.ontotext.com/knowledgehub/fundamentals/information-extraction/`.
9. KUMAR, G Ravi; BASHA, S Rahamat; RAO, Surya Bhupal. A summarization on text mining techniques for information extracting from applications and issues. *Journal of Mechanics of Continua and Mathematical Sciences, Special Issue*. 2020, č. 5.
10. PERERA, Nadeesha; DEHMER, Matthias; EMMERT-STREIB, Frank. Named entity recognition and relation detection for biomedical information extraction. *Frontiers in cell and developmental biology*. 2020, roč. 8, s. 673.

11. WANG, Sufen; PANG, Minmin; PAN, Changqing; YUAN, Junyi; XU, Bo; DU, Ming; ZHANG, Hong. Information Extraction for Intestinal Cancer Electronic Medical Records. *IEEE Access*. 2020, roč. 8, s. 125923–125934. Dostupné z DOI: 10.1109/ACCESS.2020.3005684.

12. BOSE, Priyankar; SRINIVASAN, Sriram; SLEEMAN, William C; PALTA, Jatinder; KAPOOR, Rishabh; GHOSH, Preetam. A Survey on Recent Named Entity Recognition and Relationship Extraction Techniques on Clinical Texts. *Applied Sciences*. 2021, roč. 11, č. 18, s. 8319.

13. MAKHOUL, John; KUBALA, Francis; SCHWARTZ, Richard; WEISCHEDEL, Ralph. Performance Measures For Information Extraction. *Proceedings of DARPA Broadcast News Workshop*. 2000.

14. CARMEL, David; MEJER, Avihai; PINTER, Yuval; SZPEKTOR, Idan. Improving Term Weighting for Community Question Answering Search Using Syntactic Analysis. In: *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. Shanghai, China: Association for Computing Machinery, 2014, s. 351–360. CIKM '14. ISBN 9781450325981. Dostupné z DOI: 10.1145/2661829.2661901.

15. GÓMEZ-RODRÍGUEZ, Carlos; ALONSO-ALONSO, Iago; VILARES, David. How important is syntactic parsing accuracy? An empirical evaluation on rule-based sentiment analysis. *Artificial Intelligence Review*. 2019, roč. 52, č. 3, s. 2081–2097.

16. MANSOURI, Alireza; AFFENDEY, Lilly Suriani; MAMAT, Ali. Named entity recognition approaches. *International Journal of Computer Science and Network Security*. 2008, roč. 8, č. 2, s. 339–344.

17. ZELENKO, Dmitry; AONE, Chinatsu; RICHARDELLA, Anthony. Kernel methods for relation extraction. *Journal of machine learning research*. 2003, roč. 3, č. Feb, s. 1083–1106.

18. YANG, Xi; YU, Zehao; GUO, Yi; BIAN, Jiang; WU, Yonghui. Clinical Relation Extraction Using Transformer-based Models. *arXiv preprint arXiv:2107.08957*. 2021.

19. LIU, Bing. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*. 2012, roč. 5, č. 1, s. 1–167.

20. HO, Vong Anh; NGUYEN, Duong Huynh-Cong; NGUYEN, Danh Hoang; THI-VAN PHAM, Linh; NGUYEN, Duc-Vu; VAN NGUYEN, Kiet; NGUYEN, Ngan Luu-Thuy. Emotion recognition for vietnamese social media text. In: *International Conference of the Pacific Association for Computational Linguistics*. 2019, s. 319–333.

21. RAMBOCAS, Meena; PACHECO, Barney G. Online sentiment analysis in marketing research: a review. *Journal of Research in Interactive Marketing*. 2018.

22. GAO, Chen; ZHANG, Xuan; HAN, Mengting; LIU, Hui. A review on cyber security named entity recognition. *Frontiers of Information Technology & Electronic Engineering*. 2021, roč. 22, č. 9, s. 1153–1168.

23. SOLIHIN, Firdaus; BUDI, Indra; AJI, Rizal Fathoni; MAKARIM, Edmon. Advancement of information extraction use in legal documents. *International Review of Law, Computers & Technology*. 2021, roč. 35, č. 3, s. 322–351. Dostupné z DOI: 10.1080/13600869.2021.1964225.

24. XU, Feng; PAN, Zhenchun; XIA, Rui. E-commerce product review sentiment classification based on a naive Bayes continuous learning framework. *Information Processing & Management*. 2020, roč. 57, č. 5, s. 102221.

25. DEVLIN, Jacob; CHANG, Ming-Wei; LEE, Kenton; TOUTANOVA, Kristina. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*. 2018.

26. LIU, Yinhan; OTT, Myle; GOYAL, Naman; DU, Jingfei; JOSHI, Mandar; CHEN, Danqi; LEVY, Omer; LEWIS, Mike; ZETTLEMOYER, Luke; STOYANOV, Veselin. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. Dostupné z arXiv: 1907.11692 [cs.CL].

27. RADFORD, Alec; WU, Jeffrey; CHILD, Rewon; LUAN, David; AMODEI, Dario; SUTSKEVER, Ilya et al. Language models are unsupervised multitask learners. *OpenAI blog*. 2019, roč. 1, č. 8, s. 9.

28. *Geneea Knowledge Base* [online]. Praha: GENEEA Analytics s.r.o., 2021 [cit. 2021-10-20]. Dostupné z: https://help.geneea.com/api_general/guide/entities.html#x3-guide-entities.

29. SHELAR, Hemlata; KAUR, Gagandeep; HEDA, Neha; AGRAWAL, Poorva. Named Entity Recognition Approaches and Their Comparison for Custom NER Model. *Science & Technology Libraries*. 2020, roč. 39, č. 3, s. 324–337. Dostupné z DOI: 10.1080/0194262X.2020.1759479.

30. *Models & Languages · spaCy Usage Documentation* [online]. Explosion AI, 2021 [cit. 2021-11-01]. Dostupné z: https://spacy.io/usage/models.

31. *Visualizers · spaCy Usage Documentation* [online]. Explosion AI, 2021 [cit. 2021-11-01]. Dostupné z: https://spacy.io/usage/visualizers.

32. *Character Network - A project on using network graph, NLP techniques (entity recognition, sentiment analysis) to analyse the relationships among characters in a novel* [online]. 2018 [cit. 2021-11-01]. Dostupné z: https://opensourcelibs.com/lib/character-network.

33. *Avengers: Web Scraping, Entity Extraction and Network Graphs in Python* [online]. 2020-08-31 [cit. 2021-11-01]. Dostupné z: https://medium.com/swlh/avengers-web-scraping-entity-extraction-and-network-graphs-in-python-ea6dc323eb7d.

34. DEMŠAR, Janez; CURK, Tomaž; ERJAVEC, Aleš; GORUP, Črt; HOČEVAR, Tomaž; MILUTINOVIČ, Mitar; MOŽINA, Martin; POLAJNAR, Matija; TOPLAK, Marko; STARIČ, Anže et al. Orange: data mining toolbox in Python. *the Journal of machine Learning research*. 2013, roč. 14, č. 1, s. 2349–2353.

35. *Orange Data mining - Corpus to network* [online]. University of Ljubljana, 2021 [cit. 2021-10-20]. Dostupné z: https://orangedatamining.com/widget-catalog/text-mining/corpustonetwork/.

36. *Orange Data mining - Widget catalog* [online]. University of Ljubljana, 2021 [cit. 2021-10-20]. Dostupné z: https://orangedatamining.com/widget-catalog/.

37. BAIL, Christopher Andrew. Combining natural language processing and network analysis to examine how advocacy organizations stimulate conversation on social media. *Proceedings of the National Academy of Sciences*. 2016, roč. 113, č. 42, s. 11823–11828.

38. BOY, John D. textnets: A Python Package for Text Analysis with Networks. *Journal of Open Source Software*. 2020, roč. 5, č. 54, s. 2594. Dostupné z DOI: 10.21105/joss.02594.

39. TRAAG, Vincent; ZANINI, Fabio; GIBSON, Ryan; BEN-KIKI, Oren; KUPPEVELT, Dafne van. *vtraag/leidenalg 0.8.2*. Zenodo, 2020. Ver. 0.8.2. Dostupné z DOI: 10.5281/zenodo.4047113.

40. *API Reference — textnets 0.6.0 documentation* [online]. John D. Boy, 2021 [cit. 2021-11-02]. Dostupné z: https://textnets.readthedocs.io/en/stable/_modules/textnets/corpus.html.

41. *Tutorial — textnets 0.6.0 documentation* [online]. John D. Boy, 2021 [cit. 2021-11-02]. Dostupné z: https://textnets.readthedocs.io/en/stable/tutorial.html.

42. PARANYUSHKIN, Dmitry. InfraNodus: Generating Insight Using Text Network Analysis. In: *The World Wide Web Conference*. San Francisco, CA, USA: Association for Computing Machinery, 2019, s. 3584–3589. WWW '19. ISBN 9781450366748. Dostupné z DOI: 10.1145/3308558.3314123.

43. *InfraNodus: Generate Insight Using Text Network Analsysis* [online]. Nodus Labs, 2021 [cit. 2021-11-02]. Dostupné z: https://infranodus.com/#pricing.

44. CUNNINGHAM, Hamish; TABLAN, Valentin; ROBERTS, Angus; BONTCHEVA, Kalina. Getting More Out of Biomedical Documents with GATE's Full Lifecycle Open Source Text Analytics. *PLOS Computational Biology*. 2013, roč. 9, č. 2, s. 1–16. Dostupné z DOI: 10.1371/journal.pcbi.1002854.

45. CUNNINGHAM, Hamish; MAYNARD, Diana; BONTCHEVA, Kalina; TABLAN, Valentin. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In: *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*. 2002.

46. *Plugins included in the GATE distribution* [online]. The University of Sheffield, 2021 [cit. 2021-11-19]. Dostupné z: `https://gate.ac.uk/gate/doc/plugins.html#internal-plugins`.

47. BONTCHEVA, Kalina; DERCZYNSKI, Leon; FUNK, Adam; GREENWOOD, Mark A.; MAYNARD, Diana; ASWANI, Niraj. TwitIE: An Open-Source Information Extraction Pipeline for Microblog Text. In: *Proceedings of the International Conference on Recent Advances in Natural Language Processing*. Association for Computational Linguistics, 2013.

48. MICHELFEIT, Jan; POMIKÁLEK, Jan; SUCHOMEL, Vít. Text Tokenisation Using unitok. In: *RASLAN*. 2014, s. 71–75.

49. ŠMERK, Pavel. Fast morphological analysis of Czech. *Proceedings of recent advances in slavonic natural language processing, RASLAN*. 2009, s. 13–16.

50. ŠMERK, Pavel. Unsupervised learning of rules for morphological disambiguation. In: *International Conference on Text, Speech and Dialogue*. 2004, s. 211–216.

51. KOVÁŘ, Vojtěch. *Syntaktická analýza s využitím postupné segmentace věty*. Brno, 2009. Dostupné také z: `https://is.muni.cz/th/nl1iv/`. Diplomová práce. Masarykova univerzita, Fakulta informatiky. Vedoucí práce Aleš HORÁK.

52. BAISA, Vít; KOVÁŘ, Vojtěch. Information Extraction for Czech Based on Syntactic Analysis. In: VETULANI, Zygmunt; MARIANI, Joseph (ed.). *Human Language Technology Challenges for Computer Science and Linguistics*. Cham: Springer International Publishing, 2014, s. 155–165. ISBN 978-3-319-08958-4.

53. PALA, Karel; SMRŽ, Pavel et al. Building Czech WordNet. *Romanian Journal of Information Science and Technology*. 2004, roč. 7, č. 1-2, s. 79–88.

54. POMIKÁLEK, Jan. *Removing Boilerplate and Duplicate Content from Web Corpora*. Brno, 2011. Dostupné také z: `https://is.muni.cz/th/o6om2/`. Disertační práce. Masarykova univerzita, Fakulta informatiky. Vedoucí práce Karel PALA.

55. POMIKÁLEK, Jan. *jusText*. 2011. Dostupné také z: http://hdl.handle.net/11858/00-097C-0000-000D-F696-9. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

56. NEVĚŘILOVÁ, Zuzana. Paraphrase and textual entailment generation in Czech. *Computación y Sistemas*. 2014, roč. 18, č. 3, s. 555–568.

57. PALA, Karel; ŠMERK, Pavel. Derivancze — Derivational Analyzer of Czech. In: KRÁL, Pavel; MATOUŠEK, Václav (ed.). *Text, Speech, and Dialogue*. Cham: Springer International Publishing, 2015, s. 515–523. ISBN 978-3-319-24033-6.

58. BAISA, Vít; HERMAN, Ondřej; HORÁK, Aleš et al. Manipulative Propaganda Techniques: Technical Report. 2017.

59. BILÍK, Jan; BAISA, Vít. *Webová aplikace pro analýzu sentimentu*. 2019. Dostupné také z: https://nlp.fi.muni.cz/projekty/propaganda/sentiment/.

60. FINKEL, Jenny Rose; GRENAGER, Trond; MANNING, Christopher D. Incorporating non-local information into information extraction systems by gibbs sampling. In: *Proceedings of the 43rd annual meeting of the association for computational linguistics (ACL'05)* [online]. 2005, s. 363–370 [cit. 2022-04-04]. Dostupné z: http://nlp.stanford.edu/~manning/papers/gibbscrf3.pdf.

61. SEDLÁŘ, Lubomír. *Hybridní systém pro detekci pojmenovaných entit v českém textu*. Brno, 2014. Dostupné také z: https://is.muni.cz/th/rij93/. Diplomová práce. Masarykova univerzita, Fakulta informatiky. Vedoucí práce Zuzana NEVĚŘILOVÁ.

62. STARÝ, Michal; NEVĚŘILOVÁ, Zuzana; VALCÍK, Jakub. Multilingual Recognition of Temporal Expressions. In: *RASLAN*. 2020, s. 67–78.

63. HLAVÁČKOVÁ, Dana; HORÁK, Aleš. Verbalex – new comprehensive lexicon of verb valencies for Czech. In: *Proceedings of the Slovko Conference*. 2005.

64. STRAŇÁKOVÁ-LOPATKOVÁ, Markéta; ŽABOKRTSKÝ, Zdeněk. Valency dictionary of Czech verbs: Complex tectogrammatical annotation. In: *IN LREC2002, PROCEEDINGS, VOLUME III*. 2002.

65. NEVĚŘILOVÁ, Zuzana. Declension of Czech noun phrases. In: *Actes du 31e Colloque International sur le Lexique et la Grammaire*. [N.d.], s. 134–138.

66. BOJANOWSKI, Piotr; GRAVE, Edouard; JOULIN, Armand; MIKOLOV, Tomás. Enriching Word Vectors with Subword Information. *CoRR*. 2016, roč. abs/1607.04606. Dostupné z arXiv: 1607.04606.

67. MIKOLOV, Tomás; SUTSKEVER, Ilya; CHEN, Kai; CORRADO, Greg S; DEAN, Jeff. Distributed representations of words and phrases and their compositionality. In: *Advances in neural information processing systems*. 2013, s. 3111–3119.

68. GRAVE, Edouard; BOJANOWSKI, Piotr; GUPTA, Prakhar; JOULIN, Armand; MIKOLOV, Tomás. Learning Word Vectors for 157 Languages. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*. 2018.

69. BOSTOCK, Michael; OGIEVETSKY, Vadim; HEER, Jeffrey. D^3 data-driven documents. *IEEE transactions on visualization and computer graphics*. 2011, roč. 17, č. 12, s. 2301–2309.