

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

FIIT-5212-72264

Martin Nemček

Spracovanie učebných textov

Bakalárska práca

Vedúci práce: Ing. Miroslav Blšták

Máj 2016

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

FIIT-5212-72264

Martin Nemček

Spracovanie učebných textov

Bakalárska práca

Študijný program: Informatika

Študijný odbor: 9.2.1 Informatika

Miesto vypracovania: Ústav informatiky, informačných systémov a softvérového
inžinierstva, FIIT STU, Bratislava

Vedúci práce: Ing. Miroslav Blšták

Máj 2016

Anotácia

Slovenská technická univerzita v Bratislave

FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLOGIÍ

Študijný program: Informatika

Autor: Martin Nemček

Bakalárska práca: Spracovanie učebných textov

Vedúci práce: Ing. Miroslav Blšták

Máj 2016

Sme zavalení množstvom informácií z rôznych zdrojov. V procese výučby je náročné vytvoriť poznámky, ktoré zahŕňajú podmnožinu dôležitých informácií zo zdroja. Existuje niekoľko spôsobov, ako extrahovať informácie z textu. V tejto práci navrhujeme systém na extrakciu informácií a tvorbu poznámok z textu, ktoré sú dôležité z pohľadu výučby. Systém umožní študentom vytvárať personalizované poznámky. Využívame najmä syntaktickú analýzu textu a poznámky sa vytvárajú pomocou využitia značiek slovných druhov, názvoslovných entít, lema tvarov slov a závislostí medzi slovami vo vetách. Výsledkom je interaktívny systém na tvorbu poznámok, ktoré sa vytvárajú na základe pravidiel naučených od používateľa. Pravidlo sa skladá najmä zo závislostí, je závislé od štruktúry viet a aplikovateľné na viacero viet. Overili sme eliminovanie irelevantných informácií pomocou pravidiel, aplikovanie pravidiel na viacero viet a zlepšovanie systému zo zvyšujúcim sa počtom pravidiel z hľadiska spoľahlivosti a personalizácie.

Annotation

Slovak University of Technology Bratislava

FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES

Degree Course: Informatika

Author: Martin Nemček

Bachelor thesis: Spracovanie učebných textov

Supervisor: Ing. Miroslav Blšták

May 2016

We are overwhelmed by information from various topics. The challenge in education is to create notes which cover important subset of information. There are known methods to extract information from text. In this thesis we propose a system to extract the notes from text which are important for educational purpose. System allows students to create personalized notes. We use mainly syntactic text analysis and the notes are created by help of part-of-speech tags, named entities, lemmas and dependencies between words in sentences. The outcome is an interactive system for creating notes based on the learned rules from the user. The rule consists mainly from dependencies, depends on the structure of sentence and is applicable on various sentences. We verified the elimination of the irrelevant information by the rules, application of the rules on various sentences and enhancements to reliability and personalization by the increased amount of the rules.

POĎAKOVANIE

Chcem sa poďakovať môjmu vedúcemu Ing. Miroslavovi Blštákovi za odbornú pomoc, cenné rady, veľkú ochotu a nasmerovanie ma pri písaní práce. Taktiež by som sa chcel poďakovať ľuďom, ktorí sa zúčastnili používateľského experimentu a všetkým, ktorí akokoľvek pomohli pri tvorbe tejto práce.

ČESTNÉ PREHLÁSENIE

Čestne prehlasujem, že bakalársku prácu s názvom: Spracovanie učebných textov som vypracoval samostatne, na základe konzultácií a štúdia odbornej literatúry. Neporušila som autorský zákon a zoznam použitej literatúry som uviedol na príslušnom mieste.

.....

Martin Nemček

Obsah

1	Úvod	1
1.1	Motivácia	1
1.2	Prehľad práce	1
2	Spracovanie prirodzeného jazyka	3
2.1	Úlohy spracovania prirodzeného jazyka	3
2.1.1	Značkovanie slovných druhov	4
2.1.2	Rozpoznávanie názvoslovných entít	5
2.1.3	Identifikácia koreferencií	5
2.1.4	Identifikácia gramatických závislostí	5
2.1.5	Extrakcia informácií	6
2.2	Nástroje na spracovanie prirodzeného jazyka	7
2.2.1	WordNet	7
2.2.2	StanfordNLP	8
2.2.3	Google Ngram	9
2.2.4	AlchemyAPI	10
2.3	Zhrnutie	10
3	Analýza nástrojov na správu paralelných textov	11
3.1	InterText	11
3.2	NOVA Text Aligner	12
3.3	LF Aligner	13
3.4	Google Translate	14
3.5	Zhrnutie	15
4	Návrh systému na tvorbu poznámok	16
4.1	Tvorba poznámok	16
4.1.1	Pravidlo na spracovanie	16
4.1.2	Určenie relevantných informácií	17
4.1.3	Viacnásobné poznámky	17
4.2	Určenie názvoslovných entít	18
4.3	Použitie závislostí pri tvorbe poznámok	19

4.4	Štruktúra viet a pravidiel	19
4.4.1	Reprezentácia stromom	20
4.4.2	Reprezentácia vo vete	20
4.5	Uchovávanie textov v databázach	21
4.5.1	Relačné databázy	21
4.5.2	Textové databázy	21
4.5.3	Porovnanie a výber typu databázy	22
4.6	Návrh uchovávania textov v databázach	24
4.6.1	Kolekcia articles	24
4.6.2	Kolekcia notes	25
4.6.3	Kolekcia sentences	26
4.6.4	Kolekcia structures	26
4.6.5	Kolekcia rules	28
4.6.6	Kolekcia and rules	28
4.6.7	Zhrnutie kolekcií	29
4.7	Zhrnutie	29
5	Systém na tvorbu poznámok	31
5.1	Použité technológie	31
5.2	Manažment dát	31
5.2.1	Vyhľadanie pravidla	32
5.2.2	Aplikovanie pravidla	36
5.2.3	Vytvorenie pravidla	38
5.2.4	Úprava pravidla	39
5.3	Zhrnutie	40
6	Experimenty	41
6.1	Porovnanie systému	41
6.1.1	Výsledky	41
6.1.2	Vyhodnotenie porovnania	42
6.2	Použitie pravidiel	42
6.3	Používateľský experiment	42
6.3.1	Výsledky	43

6.3.2	Vyhodnotenie experimentu	45
6.4	Zhrnutie	45
7	Zhodnotenie	47
7.1	Možnosti rozšírenia systému	48
	Zoznam použitej literatúry	49
A	Používateľská príručka	50
A.1	Inštalácia	50
A.2	Spustenie systému	50
A.2.1	Možnosti spustenia systému	50
A.3	Používateľské rozhranie	50
B	Zoznam vzťahov závislostí	56
C	Legenda diagramov kolekcí	57
D	Ukážka článkov použitých v experimente	61
E	IIT.SRC článok	66

Zoznam obrázkov

1	Strom vzťahov	6
2	Vzťahy vo vete	6
3	StanfordNLP online demo	8
4	Google Ngram Viewer	9
5	Aplikácia InterText	12
6	Aplikácia NOVA Text Aligner	13
7	Aplikácia LF Aligner	14
8	Google Translate	15
9	Stromová reprezentácia štruktúry	20
10	Reprezentácia štruktúry vo vete	21
11	Databázový model	24
12	Model kolekcie articles	25
13	Model kolekcie notes	25
14	Model kolekcie sentences	26
15	Model kolekcie structures	27
16	Model kolekcie rules	28
17	Model kolekcie and rules	29
18	Príklad určenia zhody viet	34
19	Závislostí jednoduchej vety	38
20	Príklad štruktúry pravidla	38
21	Výstupy porovnaných systémov	41
22	Spracovanie informácií porovnanými systémami	42
23	Porovnanie všeobecných výsledkov oboch častí experimentu	43
24	Detailné výsledky prvej časti experimentu	44
25	Detailné výsledky druhej časti experimentu	44
26	Grafické rozhranie - Hlavné okno	51
27	Grafické rozhranie - Menu File	52
28	Grafické rozhranie - Menu Edit	52
29	Grafické rozhranie - Okno na načítanie textu	52
30	Grafické rozhranie - Okno na dotiahnutie textu	53
31	Grafické rozhranie - Editovacie okno poznámky	54

32	Grafické rozhranie - Editovacie okno viacnásobných poznámok . .	55
33	Zoznam závislostí	56
34	Legenda diagramov kolekcí	57
35	Databázový model	58
36	Model kolekcie articles	58
37	Model kolekcie notes	58
38	Model kolekcie sentences	59
39	Model kolekcie structures	59
40	Model kolekcie rules	60
41	Model kolekcie and rules	60

Zoznam tabuliek

1	Viacnásobná poznámka	17
2	Porovnanie poskytovaných prvkov	22
3	Porovnanie používaných pojmov	23

Zoznam ukážok

1 Úvod

V dnešnej dobe sme obklopení veľkým množstvom dát z rôznych zdrojov, či už internetu, kníh alebo iných. V procese vzdelávania je problém vytvoriť poznámky zo zdroja, ktoré by zahrňovali podmnožinu dôležitých informácií.

Väčšina učebných textov je písaná v prirodzenom jazyku a má neštruktúrovanú formu. Stroje zatiaľ nedokážu úplne pochopiť prirodzený jazyk z dôvodu komplexnosti a variácie jednotlivých jazykov. Spracovanie prirodzeného jazyka je disciplína, ktorá sa zaoberá spracovaním prirodzeného jazyka strojmi. V našom systéme využívame niekoľko úloh tejto disciplíny.

Náš navrhovaný systém sa zameriava na tvorbu poznámok z viet pomocou extrakcie relevantných informácií z nich. Na tvorbu poznámok sa používa najmä syntaktická analýza viet a extrakcia vzťahov a závislostí medzi slovami viet. Výstupom nami navrhnutého systému sú personalizované poznámky. Systém implementuje prvky interaktivity, ktoré umožňujú používateľovi modifikáciu automaticky vytvorených poznámok. Na základe zmien sa systém „naučí“ nové pravidlá ako spracovávať vety. Tieto nové pravidlá zohľadní pri nasledujúcom spracovávaní zdrojov.

1.1 Motivácia

Študenti musia často spracovávať veľké množstvá dát, pričom dôležitá je pre nich iba časť dát, ktorá obsahuje relevantné informácie. Proces selekcie relevantných informácií časovo náročný. Pomocou nami navrhnutého systému by si vedeli z dát vytvoriť poznámky a tak získať podmnožinu, pre nich relevantných informácií, rýchlejšie. Vytvorené poznámky by boli personalizované, teda priamo prispôbené tvorbe poznámok používateľa a tým väčšmi zredukujú potrebný čas, keďže používateľ nebude musieť získané relevantné informácie ešte následne upravovať.

1.2 Prehľad práce

Práca začína časťou analýzy 2 Spracovanie prirodzeného jazyka, v ktorej je spracovanie prirodzeného jazyka, jeho úlohy a nástroje na podporu spracovania priro-

dzeného jazyka. Analýza pokračuje opisom a rozborom nástrojov na spracovanie paralelných textov v časti 3 Analýza nástrojov na správu paralelných textov. V časti 4 Návrh systému na tvorbu poznámok sú rozpísané návrhové rozhodnutie systému na tvorbu poznámok. Implementačné rozhodnutie sú definované v časti 5 Systém na tvorbu poznámok. Nasleduje časť 6 Experimenty, v ktorej sa nachádzajú vyhodnotenia naším experimentov. Práca je ukončená časťou 7 Zhodnotenie.

2 Spracovanie prirodzeného jazyka

Spracovanie prirodzeného jazyka (angl. Natural Language Processing - NLP) odkazuje na počítačové systémy, ktoré spracovávajú, snažia sa pochopiť alebo generujú jeden alebo viacero ľudských jazykov. Vstupom môže byť text alebo hovorená reč s cieľom prekladu do iného jazyka, pochopenie a reprezentácia obsahu textu, udržanie dialógu s používateľom a iné [1]. Počítače doposiaľ nedokážu plne porozumieť ľudskému jazyku, či už sa jedná o písaný alebo hovorený, a preto hlavným cieľom spracovania prirodzeného jazyka je vybudovať výpočtové modely prirodzeného jazyka pre jeho analýzu a generovanie [2].

Porozumenie ľudskej reči je mnohokrát náročné aj pre samotných ľudí a nie to ešte pre počítače. Na svete je veľké množstvo jazykov, ktoré sa od seba líšia typickými charakteristikami. Navyše, každý človek je odlišný, čo spôsobuje, že výslovnosť rovnakého slova viacerými ľuďmi môže byť odlišná. Ďalej máme slangové slová a slová typické len pre určité územie. Pri spracovávaní prirodzeného jazyka treba vziať do úvahy viaceré aspekty. Dosiahnutie tohto cieľa je preto často veľmi náročné.

V súčasnosti najpoužívannejšie algoritmy na spracovanie prirodzeného jazyka využívajú strojové učenie.

2.1 Úlohy spracovania prirodzeného jazyka

Spracovanie prirodzeného jazyka má niekoľko hlavných úloh. V nasledujúcich častiach sú podrobnejšie opísané tie, ktoré sú relevantné vzhľadom na spracovanie učebných textov. Úlohy spracovania prirodzeného jazyka: [5]

- Značkovanie slovných druhov (angl. Part-of-speech tagging) 2.1.1,
- Rozdelenie vety na menšie časti (angl. Chunking),
- Rozpoznávanie názvoslovných entít (angl. Named Entity Recognition) 2.1.2,
- Označovanie sémantického postavenie (angl. Semantic Role Labeling),
- Rozpoznanie koreferencií (angl. Coreference resolution) 2.1.3,
- Morfologické segmentovanie (angl. Morphological Segmentation),
- Generovanie prirodzeného jazyka (angl. Natural Language Generation),

- Optické rozoznávanie textu (angl. Optical Character Recognition),
- Rozloženie vzťahov (angl. Dependency parsing) 2.1.4,
- a ďalšie.

Hlavné úlohy spracovania prirodzeného jazyka sú implementované a využívané vo viacerých smeroch. Z hľadiska spracovania učebných textov je pre nás najdôležitejšie využitie extrakcie informácií. Ďalšie využitia spracovania prirodzeného jazyka sú napríklad: [9]

- strojový preklad (angl. Machine Translation),
- rozpoznávanie reči (angl. Speech Recognition),
- sumarizáciu textu (angl. Text Summarization),
- dialógové systémy (angl. Dialogue Systems),
- vyhľadávanie informácií (angl. Information Retrieval),
- a ďalšie.

2.1.1 Značkovanie slovných druhov

Hlavnou úlohou značkovania slovných druhov (angl. Part-of-speech tagging) je každému slovu vo vete priradiť unikátnu značku označujúcu jeho syntaktickú úlohu vo vete [5]. Sú to, napríklad v slovenskom jazyku podmet, prísudok, príslovkové určenie alebo v anglickom jazyku noun, adverb, verb, atď. Okrem vymenovaných označení to môže byť aj označenie určujúce množné číslo, napríklad singulár alebo plurál.

Problémom pri značkovaní slovných druhov je mnohoznačnosť. Mnohoznačnosť je vlastnosť slova spôsobujúca, že slovo môže mať viacero významov a môže byť viacerými slovnými druhmi. V slovenskom jazyku napríklad slovo *kry* môže predstavovať sloveso s významom rozkazu *prikry!*, ale taktiež môže predstavovať podstatné meno s významom *kríky*. V anglickom jazyku napríklad slovo *book*, ktoré môže predstavovať podstatné meno *kniha* alebo sloveso vo význame *rezervovať*.

2.1.2 Rozpoznávanie názvoslovných entít

Rozpoznávanie názvoslovných entít (angl. Named Entity Recognition) označuje mená a názvy (entity), ktoré sa vyskytujú v texte. Tie následne rozdeľuje do kategórií, ako sú napríklad *osoby*, *organizácie* alebo *lokácie* [5].

Problémy pri rozpoznávaní názvoslovných entít spôsobuje kapitalizácia slov, takzvané písanie entít s veľkým začiatočným písmenom. V anglickom jazyku je to jednoduché, z dôvodu písania entít s veľkým začiatočným písmenom.

Príkladom je *Slovak University of Technology*. Avšak v iných jazykoch to neplatí a entity sa nemusia písať s veľkým začiatočným písmenom.

2.1.3 Identifikácia koreferencií

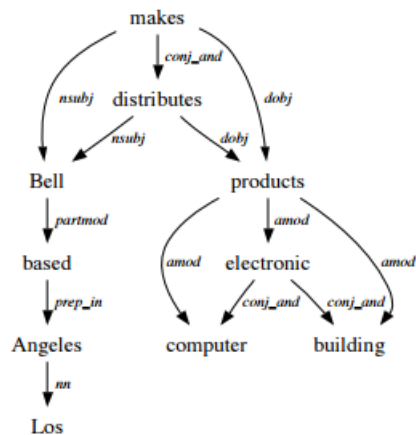
Identifikácia koreferencií (angl. Coreference resolution) predstavuje nájdenie alternatívnych pomenovaní rovnakej entity na základe referencií a kontextu. V texte sa často používajú zámena (angl. pronouns) *to*, *tí*, *on*, anglicky *it*, *those*, *he* alebo menné frázy (angl. noun phrase). Tieto referencie sa odkazujú na iné podstatné mená alebo mená a názvy. Rozpoznávanie koreferencií identifikuje entitu, väčšinou z reálneho sveta, na ktorú sa referencia odkazuje. Táto úloha spracovania prirodzeného jazyka sa využíva v aplikáciách spracovania prirodzeného jazyka, ako sú extrakcia informácií (viď. 2.1.5 Extrakcia informácií) a odpovedanie na otázky [3].

Príklad: **Martin Nemček** napísal túto bakalársku prácu. **On** študuje na FIIT STU BA.

Zámeno *on* sa odkazuje na meno *Martin Nemček*.

2.1.4 Identifikácia gramatických závislostí

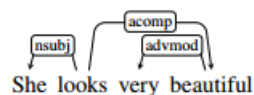
Identifikácia vzťahov nám poskytuje jednoduchý opis gramatických vzťahov slov vo vete. Aplikovaním identifikácie vzťahov na vetu *Bell, based in Los Angeles, makes and distributes electronic, computer and building products.* vznikne strom vzťahov (angl. dependency tree) (viď. Obr. 1) [4].



Obr. 1: Strom vzťahov

V tomto orientovanom stromovom grafe jednotlivé slová vety predstavujú vrcholy, pričom prechody medzi vrcholmi, hrany, reprezentujú vzťahy medzi nimi.

Ďalšia reprezentácia závislostí zapisuje vzťahy priamo do vety. Na Obr. 2 je vidno, že medzi slovami *She* a *looks* je vzťah **nsubj** - nominal subject, medzi *looks* a *beautiful* je vzťah **acomp** - adjectival complement, a v neposlednom rade medzi slovami *very* a *beautiful* je vzťah **advmod** - adverb modifier [4].



Obr. 2: Vzťahy vo vete

Celá závislosť sa skladá primárne z nadradeného tokenu, podradeného tokenu a vzťahu medzi nimi. Na Obr. 2 vidno, okrem iných aj závislosť, ktorej nadradený token je slovo *looks*, podradený token je slovo *She* a vzťah je *nsubj*.

2.1.5 Extrakcia informácií

Systémy a aplikácie zamerané na extrakciu informácií vyhľadávajú a extrahujú informácie z textov, článkov a dokumentov, pričom reagujú na používateľove

požiadavky. Výstup nepozostáva iba zo zoznamu kľúčových slov, ktoré by sa dali pokladať za extrahované informácie, ale naopak sú v tvare preddefinovaných šablón [9].

Extrakcia informácií využíva niekoľko z hlavných úloh spracovania prirodzeného jazyka. Sú to *značkovanie slovných druhov*, *rozpoznávanie názvoslovných entít*, a ďalšie [9].

Vyhľadanie informácií a extrakcia informácií spolu úzko súvisia. Prvé využitie slúži na vyhľadávanie relevantných zdrojov informácií v databázach textov, článkov a dokumentov podľa používateľových potrieb. Na vyhľadaných zdrojoch následne prebehne extrakcia informácií.

2.2 Nástroje na spracovanie prirodzeného jazyka

V súčasnosti je vyvinutých, alebo sú vo vývoji, viacero nástrojov, ktoré sa dajú použiť pri spracovávaní prirodzeného jazyka. Vývoj takýchto nástrojov je podporovaný na známych univerzitách ako sú napríklad Princeton, Stanford alebo Cambridge, ale aj mimo univerzít napríklad v Google.

2.2.1 WordNet

WordNet¹ je databáza anglických slov vyvíjaná na Princetonskej univerzite. Databáza obsahuje podstatné mena, prídavné mená, slovesá a príslovky, ktoré sú zatriedené do synonymických sád, synsetov.

Slová do synetov sú zaraďované podľa významu. To znamená, že slová *auto* a *automobil*, ktoré sú pre svoj význam zameniteľné vo vete, sa zaraďujú do rovnakého synsetu. WordNet v súčasnosti (r. 2015) obsahuje 117 000 synsetov. Každý z nich obsahuje aj krátku ukážku použitia slova.

Vo WordNet sa nachádzajú aj vzťahy medzi slovami v zmysle nadradenosti. To znamená, že *stolička* je nábytok a *nábytok* je fyzická vec a takto to pokračuje až po najvyššie slovo, od ktorého „dedia” všetky - entita. Okrem vzťahu nadradenosti WordNet obsahuje aj vzťah zloženia. *Stolička* sa skladá z *operadla* a *nôh*. Toto zloženie je typické len pre konkrétne slovo a neprenáša sa hore stromom nadradenosti, lebo pre *stoličku* je typické, že sa skladá z *operadla* a *nôh*, ale to už

¹www.wordnet.princeton.edu

nie je typické pre nábytok. Prídavné mená obsahujú aj vzťah antonymity, takže slovo *suchý* bude prepojené so slovom *mokrý* ako so svojím antonymom.

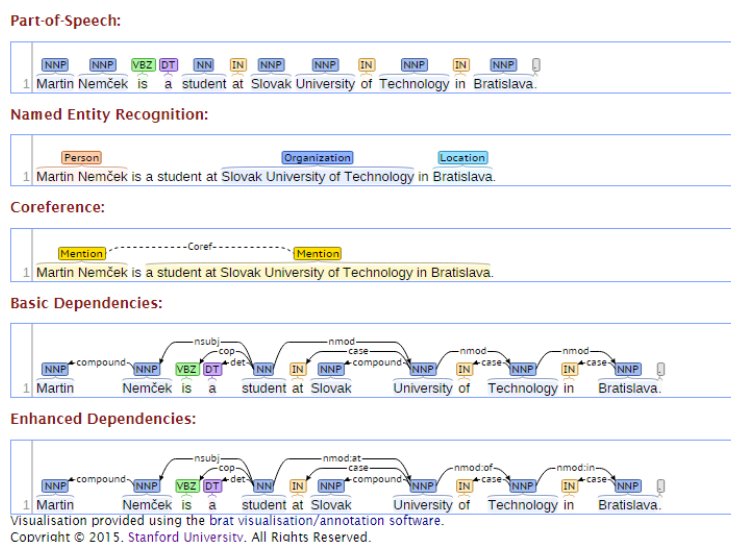
Tento nástroj je dostupný vo webovej verzii, ale ponúka aj stiahnutie jeho databázových súborov.

2.2.2 StanfordNLP

Nástroj StanfordNLP² je vyvíjaný na Stanfordskej univerzite. Skladá sa z niekoľkých softvérov, ktoré sa zameriavajú na úlohy spracovania prirodzeného jazyka. Sú to *Stanford Parser*, *Stanford POS Tagger*, *Stanford EnglishTokenizer*, *Stanford Relation Extractor* a mnoho ďalších. *Stanford CoreNLP* zahŕňa viacero zo spomenutých softvérov.

Nástroje StanfordNLP sú implementované v Jave, ale dostupné aj v iných programovacích jazykoch ako C#, PHP alebo Python.

Dostupné je aj online webové demo. Na Obr. 3 vidno výstupy z nástrojov ponúkaných balíkom StanfordNLP pre jednoduchý vstupný text skladajúci sa z jednej vety „Martin Nemček is a student at Slovak University of Technology in Bratislava.”.



Obr. 3: StanfordNLP online demo

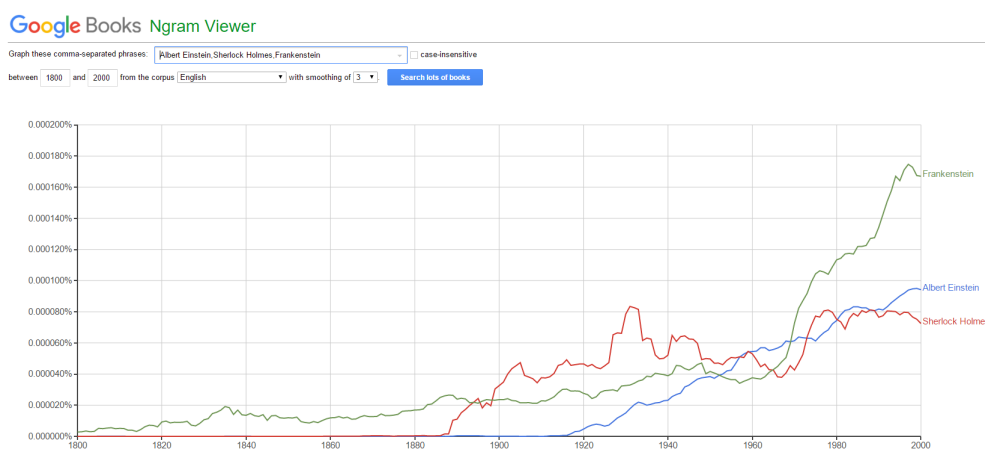
²www.nlp.stanford.edu

2.2.3 Google Ngram

Google Ngram³ je vyvinutý spoločnosťou Google. V knihách, napísaných od roku 1500 až do súčasnosti, vyhľadáva výskyty n-gramov. Podporuje len niektoré jazyky, ako angličtina, francúzština, ruština alebo čínština. Na vyhľadávanie v knihách využíva optické rozpoznávanie textu, pričom dokáže spracovať aj regulárne výrazy, avšak tie môžu byť použité iba ako náhrada celého slova, ale nie uprostred slova. Regulárny výraz „* Einstein” spracuje, pričom „Albert Einste*n” nie.

N-gram je podľa oxfordského slovníka⁴ definovaný ako postupnosť n za sebou idúcich slov alebo znakov. *Martin* je n-gram veľkosti jedna, teda 1-gram alebo unigram. *Martin Nemček* je n-gram veľkosti dva, 2-gram alebo bigram a tak ďalej, pričom n môže byť ľubovoľné kladné, celé číslo.

Google Ngram Viewer poskytuje vizualizáciu vyhľadaných dát a je dostupný vo webovom rozhraní. Na Obr. 4 vidno vizualizáciu výskytu mien *Albert Einstein*, *Sherlock Holmes*, *Frankenstein* v knihách od roku 1800 do roku 2000.



Obr. 4: Google Ngram Viewer

Tento nástroj okrem iného ponúka aj surové (angl. raw) dáta na stiahnutie.

³www.books.google.com/ngrams

⁴<http://www.oxforddictionaries.com/>

2.2.4 AlchemyAPI

AlchemyAPI⁵ je súbor dvanástich nástrojov, z ktorých sú niektoré zamerané na úlohy spracovania prirodzeného jazyka popísané v časti 2.1 Úlohy spracovania prirodzeného jazyka, ako napríklad extrakcia entít, extrakcia kľúčových slov, extrakcia vzťahov, ale aj iné zaujímavé funkcie, napríklad extrakcia autora z textu.

Na používanie tohto nástroja je potrebná registrácia pre obdržanie API kľúču. S týmto kľúčom je tisíc dopytov denne zdarma. Dostupnosť v programovacích jazykoch je široká. Ponúka knižnicu v deviatich najpoužívanejších programovacích jazykoch. Dostupné je aj online webové demo.

Dáta sú vo formáte JSON a okrem spracovania prirodzeného jazyka AlchemyAPI ponúka aj nástroje na extrahovanie obsahu z obrázku alebo rozpoznávanie tvárí na obrázkoch.

2.3 Zhrnutie

Spracovanie prirodzeného jazyka napomáha spracovávaniu, pochopeniu alebo generovaniu ľudského jazyka strojmi. Rieši viacero úloh, ktoré sú využívané v niekoľkých smeroch. Z pohľadu spracovania učebného textu sú dôležité hlavne úlohy *značkovanie slovných druhov*, *rozpoznávanie názvoslovných entít*, *rozloženie vzťahov* využívané v smere *extrakcií informácií*. Existuje viacero nástrojov na podporu spracovania prirodzených jazykov. Veľká časť je vyvíjaná na univerzitách, ako aj nástroj *StanfordNLP*, ktorý ponúka informácie ako sú značky slovných druhov, typy názvoslovných entít, závislosti a ďalšie.

⁵www.alchemyapi.com

3 Analýza nástrojov na správu paralelných textov

Dostupnosť aplikácií na spracovanie prirodzeného jazyka je veľká a široká. My sa zameriavame na aplikácie, ktoré umožňujú editovať paralelný text.

Nástroje na správu paralelných textov uľahčujú spracovanie viacerých druhov a verzií toho istého textu. V jednej časti nástroja je zdrojový text alebo súbor a v druhej časti výsledný text alebo súbor. Hlavný dôraz sa kladie práve na transformáciu zo zdrojového textu na cieľový. Transformácia môže mať viacero podôb, ako preklad, zarovnanie, zjednodušenie textu, a mnoho ďalších. Texty sú zväčša, pre zjednodušenie transformácie, rozdelené podľa viet, pričom vety na jednej úrovni zvyčajne spolu súvisia podľa určitej vlastnosti.

V nasledujúcich častiach je zanalyzovaných niekoľko nástrojov tohto typu.

3.1 InterText

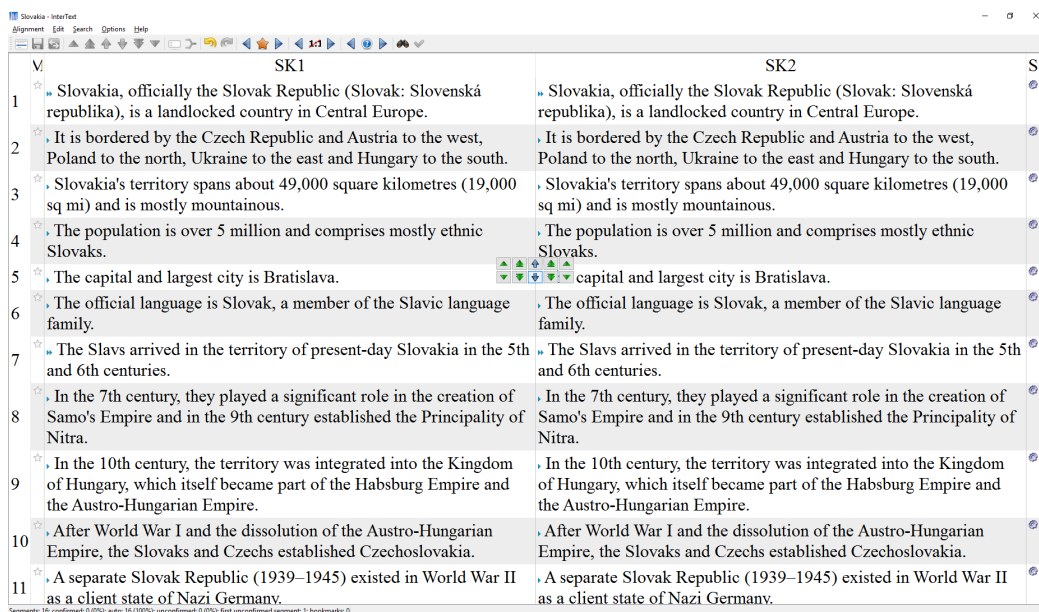
InterText⁶ je textov, využívaný na správu viacerých paralelne zarovnaných verzií textu rôznych jazykov na úrovni viet. Táto aplikácia je dostupná vo verzií pre desktop a server.

Podporuje viacero formátov textu, či už čistý (angl. plain) text alebo XML a taktiž zobrazuje aj HTML značky. Riadky obsahujú vety oddelené znakom konca riadku a sú očíslované. Umožňuje funkcie ako presúvanie riadkov textu alebo zoskupenie viacerých do jedného, krok vpred a vzad. V spracovávanom texte sa dá vyhľadávať a je možné tento text aj upraviť podľa vlastných potrieb.

InterText nezohľadňuje používateľove úpravy textu počas používania a pri následnom spracovávaní textu sa tak neprispôsobí používateľovi. Okrem toho zjednodušovanie textu v tomto nástroji by bolo pomerne náročné.

Na Obr. 5 je zobrazená aplikácia InterText s testovacím vstupom, na ktorom je vidno väčšinu už spomenutej funkcionality, ako presúvanie a zoskupovanie riadkov, číslovanie, atď.

⁶<http://wanthalf.saga.cz/intertext>



Obr. 5: Aplikácia InterText

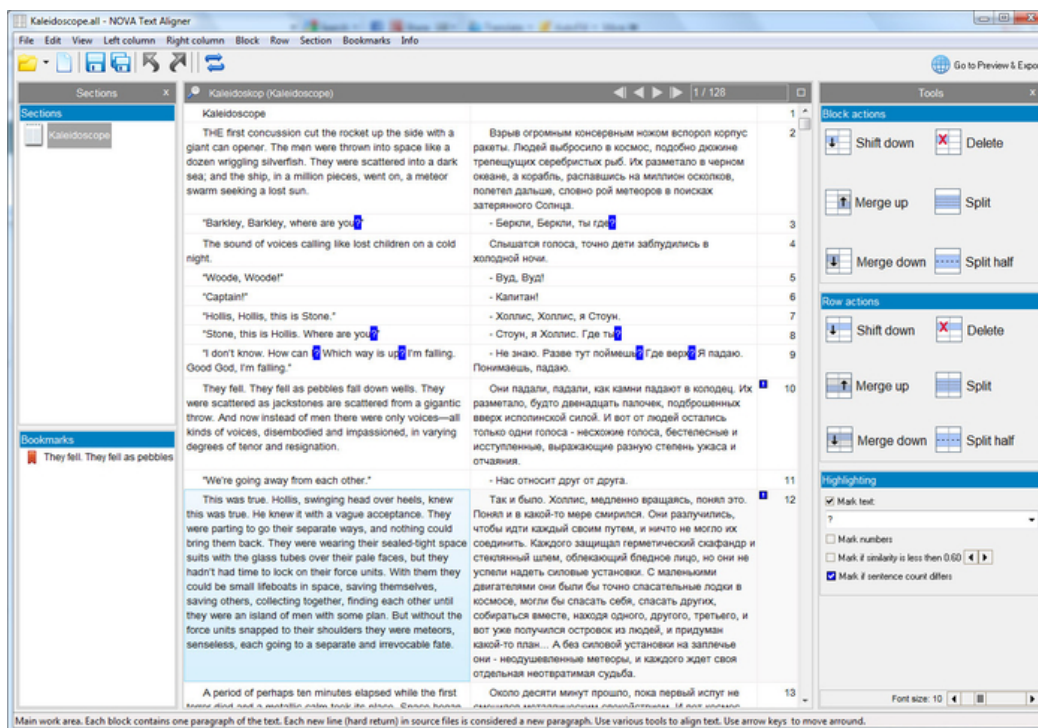
3.2 NOVA Text Aligner

NOVA Text Aligner⁷ je aplikácia na zarovnávanie textu, pričom nevyužíva algoritmy na zarovnávanie textu, ale používateľ si musí sám určiť zarovnanie.

Ako vidno na Obr. 6, hlavná editovacia časť aplikácie je rozdelená do dvoch častí. Umožňuje do ľavej aj pravej časti načítať rôzny text, v ktorom sa dá veľmi jednoducho vyhľadávať, k čomu napomáha zvýraznenie vyhľadaných slov. Načítaný text je možné premiestňovať a zoskupovať, či už podľa riadkov alebo aj v celých blokoch a nechýba možnosť editovať text. Je možné si túto aplikáciu prispôbiť. Ponúka možnosti ako zmena typu písma a pod. Finálny spracovaný text sa dá exportovať do viacerých formátov, z ktorých populárne sú formáty elektronických knížiek EPUB a MOBI.

Aplikácia je zameraná hlavne na usporadúvanie textu, nezaznamenáva si používateľove zmeny textu a neprispôbuje sa podľa toho pri ďalšom použití a funguje iba lokálne. NOVA Text Aligner je dostupná iba v skúšobnej verzii, pre dlhodobé používanie si treba zakúpiť licenciu.

⁷<http://www.supernova-soft.com/wpsite/products/text-aligner/>



Obr. 6: Aplikácia NOVA Text Aligner⁸

3.3 LF Aligner

Aplikácia LF Aligner⁹ je zameraná na spracovanie textu rôznych jazykov. Ponúka možnosť použiť až 99 jazykov, čo ale znamená 99 vstupných súborov, každý so zvoleným jazykom. Dokáže spracovať rôzne typy vstupných súborov od čistého textu, PDF súborov, cez URL stránok s textom až po správy Európskeho parlamentu, ktoré automaticky stiahne. Výstup môže byť taktiež viacerých druhov, napríklad cez grafické rozhranie LF Aligner alebo vygenerovanie XLS súboru. Na Obr. 7 vidno grafické rozhranie tejto aplikácie, ktoré ponúka mnohé vymoženosti. Samozrejmosťou je možnosť premiestňovať a zoskupovať riadky, doplnenie ďalšieho súboru na spracovávanie, uloženie zmien súboru prepísaním jeho dát a mnohé ďalšie.

⁸<http://parallel-text-aligner.en.softonic.com/>

⁹www.sourceforge.net/projects/aligner

LF Aligner Editor 1.5 - aligned_tmp-tmp2.txt

1	Slovakia (Slovak: Slovensko) (Official name The Slovak Republic, Slovenská republika) is a country with no access to the ocean in Central Europe.	Czech Republic (Czech: Česká republika) is a country in Central Europe, sometimes also known as Czechia (Czech: Česko).	.tmp-.tmp2
2	It is bordered by Austria in the southwest, Hungary in the south, Ukraine in the east, Poland in the north and Czech Republic in the northwest.	The capital and the biggest city is Prague. The currency is the Czech Crown (koruna česká - CZK).	.tmp-.tmp2
3	Its capital city is Bratislava.	1 € is about 27 CZK.	.tmp-.tmp2
4		The president of the Czech Republic is Miloš Zeman.	.tmp-.tmp2
5	Other main cities are Košice, Banská Bystrica, Žilina, Trenčín, Nitra, Prešov, and Trnava.	The Czech Republic's population is about 10.5 million. The local language is Czech language.	.tmp-.tmp2
6	Slovakia is a member of the European Union.	The Czech language is a Slavic language.	.tmp-.tmp2
7		It is related to languages like Slovak and Polish.	.tmp-.tmp2
8		In 1993 the Czech Ministry of Foreign Affairs announced that the name Czechia be used for the country outside of formal official documents.	.tmp-.tmp2
9		This has not caught on in English usage.	.tmp-.tmp2
10		Czech Republic has no sea; its neighbour countries are Germany, Austria, Slovakia, and Poland.	.tmp-.tmp2

Merge (F1) Split (F2) Shift up (F3) Shift down (F4)

Obr. 7: Aplikácia LF Aligner

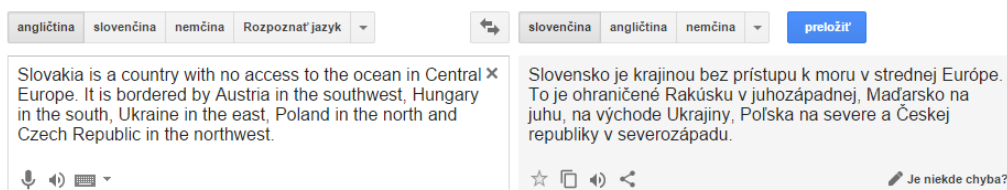
3.4 Google Translate

Za najznámejšieho zástupcu webových nástrojov na spracovanie paralelných textov sa dá pokladať nástroj Google Translate¹⁰. Využíva sa na preklad slov, viet, ale dokáže spracovať aj celé texty. Momentálne podporuje preklad z a do 91 jazykov. Dokáže rozpoznať a preložiť hovorenú reč aj písaný text. Pri preklade jednotlivých slov zobrazuje viacero možných prekladov do druhého jazyka, pričom pri preklade z anglického jazyka ponúka aj ukážky viet, v ktorých sa prekladané slovo môže použiť. Správnu výslovnosť preloženého aj prekladaného slova alebo textu, si používateľ môže vypočúť na krátkej zvukovej ukážke.

Na Obr. 8 je zobrazený preklad anglického textu do slovenského. Vidno, že preklad do minoritných jazykov ešte nie je dokonalý.

⁹<http://parallel-text-aligner.en.softonic.com/>

¹⁰translate.google.com



Obr. 8: Google Translate

3.5 Zhrnutie

AnalYZovali sme aplikácie, ktoré umožňujú spravovať a editovať paralelný text. Za ich pomoci dokážeme zo vstupného textu získať výstupný text. Napríklad pri preklade máme vstupný text množinu viet v anglickom jazyku, ktorú chceme preložiť do slovenského jazyka a výstupný text je preloženú množinu viet. Pri zjednodušovaní textu je na vstupe taktiež množina viet a na výstupe je každá veta zo vstupnej množiny zjednodušená podľa istých pravidiel. Výstupný text vzniká určitou transformáciou vstupného textu, aplikovaním transformácie na každú vetu zdrojového textu.

AnalYZované nástroje nespĺňajú všetky požiadavky na systém schopný spoznávkovať učebný text v takom rozsahu, ktorý by umožňoval používateľovi prispôbiť si, ako systém spracováva text. Systém musí umožňovať editáciu jednotlivých viet výstupného textu podľa vôle používateľa. Tieto úpravy musí zohľadniť pri následnej aplikácii transformácií vstupného textu. Dáta ohľadne spoznávkovania textu musia byť uložené na externom úložisku, ako napríklad v databáze na serveri.

4 Návrh systému na tvorbu poznámok

Systém na tvorbu poznámok vyžaduje splnenie viacerých požiadaviek. Hlavné z nich sú vytvorenie poznámky a možnosť jej následnej interaktívnej úpravy. Ďalej sú to tvorba viacnásobných poznámok, vytváranie poznámok na základe pravidiel s využitím závislostí slov, ktoré môžu byť aplikované na viacero viet, výber vhodnej databázy s adekvátne vytvoreným databázovým modelom. Spomenuté požiadavky sú opísané v nasledujúcich častiach.

4.1 Tvorba poznámok

Pri spracovávaní viet je potrebné extrahovať dôležité a konkrétne informácie, v našom prípade slová, a z extrahovaných informácií vytvoriť poznámku. Poznámka je zložená výhradne zo slov obsiahnutých vo vete, z ktorej bola vytvorená. Tým zabezpečíme, že poznámka nebude obsahovať irelevantné informácie z hľadiska vzťahu ku vete.

Spracovávané vety sa môžu opakovať, a preto je dôležité, aby sa rovnaké a podobné vety spracovali vždy takým istým spôsobom. Pre tento účel je vhodné mať definované pravidlo, ktoré bude určovať, ako sa má daná veta spracovať. Aplikovaním pravidla na vetu vytvoríme poznámku. Používateľ má možnosť interaktívne upraviť poznámku, a tým pádom aj pravidlo, ktoré bolo použité na vytvorenie danej poznámky, v takom rozsahu, že sa vytvorená poznámka zmení podľa vykonaných zmien. Zmeny v pravidle sa musia prejaviť aj pri nasledujúcom aplikovaní pravidla na rovnakú alebo podobnú vetu. Používateľovi to poskytne kontrolou nad spôsobom, akým sa vytvárajú poznámky a bude si ich vedieť prispôbiť - personalizovať.

4.1.1 Pravidlo na spracovanie

Pravidlo na spracovanie určuje, ktoré informácie vo vete sú relevantné a majú byť použité v poznámke. Okrem určenia relevantnosti slov pravidlo taktiež určuje, na ktorej pozícii sa má konkrétna informácia vo výslednej poznámke nachádzať. Výsledná poznámka sa nemusí skladať iba z jednej vety, ale môže pozostávať z

ľubovoľného počtu viet. Počet viet, z ktorých bude poznámka pozostávať a miesta, na ktorých má byť pôvodná poznámka rozdelená na vety určuje pravidlo.

Skladá sa primárne zo štruktúry, ktorá je zložená najmä zo závislostí. Okrem štruktúry obsahuje aj informácie o pozíciách, na ktorých sa má výsledná poznámka rozdeliť na menšie časti - vety.

Pravidlo je aplikovateľné nielen na totožné vety, ale aj na vety s rovnakou štruktúrou je nezávislé od obsahovej časti vety.

4.1.2 Určenie relevantných informácií

Určenie relevantnosti informácií pri učebných textoch a poznámkach je subjektívne. Informáciu, ktorú niekto považuje za relevantnú, môže byť pre iného irelevantná. Relevantnosť informácií preto určuje používateľ pomocou vytvárania nových a úpravy existujúcich pravidiel cez spracovávanie článkov, viet a úpravou vytvorených poznámok. Tým si používateľ prispôsobuje tvorbu poznámok, aby zodpovedala jeho predstave relevantnosti a celkovo tvorbe poznámok. Na tento účel mu systém ponúka interaktívne a intuitívne rozhranie.

4.1.3 Viacnásobné poznámky

Viacnásobná poznámka je poznámka, ktorá sa skladá z viacerých viet, ktoré majú rovnaký základ a sú doplnené o prvok z množiny spojky *a* (angl. *and*). V praxi to vyzerá tak, že z vety „*It is bordered by Austria in the southwest, Hungary in the south, Ukraine in the east, Poland in the north and Czech Republic in the northwest.*” sa dá vytvoriť viacnásobná poznámka v tvare zobrazenej v tabuľke 1 Viacnásobná poznámka.

Tabuľka 1: Viacnásobná poznámka

It is bordered by Austria in the southwest.
It is bordered by Hungary in the south.
It is bordered by Ukraine in the east.
It is bordered by Poland in the north.
It is bordered by Czech Republic in the northwest.

Z tabuľky vidno, že rovnaký základ viet vo viacnásobnej poznámke je *It*

is border by, ktorý je rozšírený o prvky z množiny {*Austria in the southwest, Hungary in the south, Ukraine in the east, Poland in the north, Czech Republic in the northwest*}.

Množina sa určuje pomocou dvoch závislostí. Prvá je závislosť so vzťahom *conjunction* a špecifikom *and*, ktorá nám určuje, ktoré slová sú prepojené spojkou *a*. Druhá využívaná závislosť, je závislosť so vzťahom *appositional modifier*, ktorá určuje slova prepojené čiarkou, keďže spojka *a* môže byť nahradená čiarkou. Pomocou týchto závislostí vieme extrahovať z vety jednotlivé prvky množiny a množinu vytvoriť.

Základ viet vo viacnásobnej poznámke je relevantná informácia z pôvodnej vety určená pravidlom, okrem prvkov množín. Po aplikovaní pravidla a získaní základu viet sa následne namnoží a spojí s jednotlivými prvkami množiny a vytvoria tak viacnásobnú poznámku.

4.2 Určenie názvoslovných entít

Názvoslovné entity nám určujú typ entity slova. Pre využitie týchto informácií rozdelíme názvoslovné entity do deviatich kategórií:

- lokácia,
- osoba,
- organizácia,
- mena,
- percentá,
- dátum,
- čas,
- číslo,
- ostatné.

Kategórie názvoslovných entít sú súčasťou pravidiel. Pomocou nich dokážeme presnejšie identifikovať relevantnú informáciu vo vete. Pri určovaní zhôd viet vieme, vďaka kategóriám názvoslovných entít, presnejšie určiť zhodu, keďže dokážeme identifikovať, či vety obsahujú rovnaké kategórie názvoslovných entít. Je potrebné používateľovi odlíšiť jednotlivé kategórie, ako pomoc pri úprave

poznámky a výbere relevantnej informácie.

4.3 Použitie závislostí pri tvorbe poznámok

Závislosti majú viacero využití pri tvorbe poznámok. Hlavné z nich sú:

- jednoznačná identifikácia informácie,
- pravidlo závislé iba od štruktúry,
- menší počet potrebných pravidiel.

Závislosť nám zoskupuje informácie o jej tokenoch a vzťahu medzi nimi. Na základe nich vieme, bez potreby iných vstupov, jednoznačne identifikovať konkrétnu informáciu vo vete alebo poznámke.

Pravidlo, ktoré je tvorené prevažne zo závislostí, nie je viazané na konkrétnu vetu, pri spracovávaní ktorej vzniklo, ale viaže sa na štruktúru vety. Táto vlastnosť nám umožňuje použiť jedno a to isté pravidlo na spracovanie viacerých viet.

Keby sme vytvárali pravidlá napríklad, iba na základe značiek názvoslovných druhov a ukladali by sme si v akom poradí boli značky v pôvodnej vete a v akom poradí v poznámke, potrebovali by sme pre každú obmenu vety nové pravidlo. Keďže je pravidlo naviazané na štruktúru vety a nie na obsahovú časť, teda aj poradie slov, vieme jedným pravidlom spracovať viacero obmien jednej vety, za predpokladu, že sa obmenou nezmenila štruktúra. Aplikovateľnosťou pravidla na viacero viet sa nám redukuje počet všetkých možných potrebných pravidiel.

4.4 Štruktúra viet a pravidiel

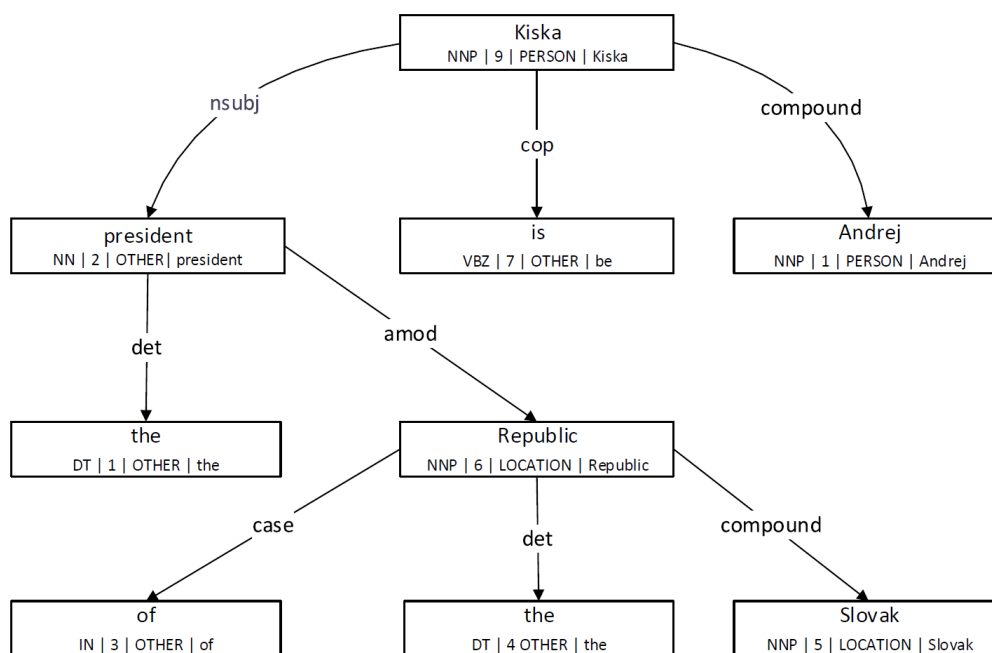
Všeobecná štruktúra je definovaná závislosťami, vzťahmi závislostí, pozíciami závislostí, a informáciami o nadradenom tokene a podradenom tokene závislostí. Tieto informácie sa skladajú zo značky slovného druhu, indexu, názvoslovnej entity a lemy. Štruktúra pravidla je oproti všeobecnej štruktúre, ktorú využíva veta, rozšírená o dopĺňujúce informácie. Sú to informácie potrebné pre správny chod systému. Skladajú sa z typu porovnania a typu tokenu. V náčrtoch sa nezobrazujú keďže sú využívané interne a nie sú nevyhnutné na pochopenie danej časti.

Štruktúru viet prezentujeme dvoma spôsobmi:

- stromom,
- vo vete.

4.4.1 Reprezentácia stromom

Stromová reprezentácia uľahčuje reprezentáciu vzťahov a prepojení jednotlivých slov vo vete. Táto reprezentácia môže byť pre vetu „*The president of the Slovak Republic is Andrej Kiska.*” vyjadrená Obr. 9. Reprezentácia je podobná reprezentácií na Obr. 1 z časti 2.1.4 Identifikácia gramatických závislostí, avšak je ešte rozšírená o značky slovných druhov, indexy, názvoslovné entity a lemy. V tomto poradí sú tieto informácie zobrazené v stromovej reprezentácii pod príslušným slovom.

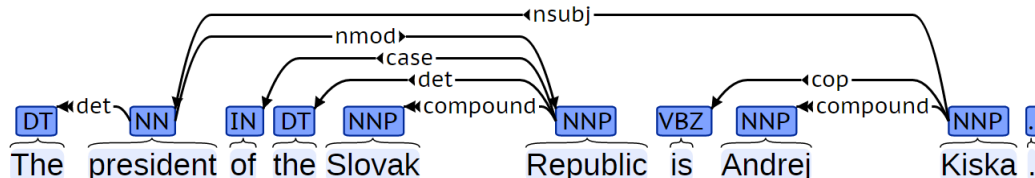


Obr. 9: Stromová reprezentácia štruktúry

4.4.2 Reprezentácia vo vete

Reprezentácia vo vete taktiež reprezentuje vzťahy a prepojenia slov vo vete. Zo zobrazenia tejto reprezentácie sa dá jednoduchšie pochopiť celá štruktúra, keďže zobrazuje slová v pôvodnom poradí a celkovo vetu v pôvodnom formátovaní. Pre

vetu „*The president of the Slovak Republic is Andrej Kiska.*” vyzerá reprezentácia vo vete ako na Obr. 10.



Obr. 10: Reprezentácia štruktúry vo vete

4.5 Uchovávanie textov v databázach

Text je špecifický údajový model s variabilnou štruktúrou, ktorý potrebujeme efektívne ukladať v databáze. Je nevyhnutné použiť vhodnú databázu, ktorá je prispôbená ukladaniu textov. Databáza musí obsahovať vlastnosti, ktoré umožňujú jednoduché narábanie s dátami s variabilnou štruktúrou a nemajú prebytočnú spotrebu pamäti pri ich uchovávaní. Taktiež je potrebná podpora bezproblémového ukladania, získavania, vyhľadávania a spracovania textov na úrovni databázy. Pri výbere vhodnej databázy sme vybrali z viacerých alternatív.

4.5.1 Relačné databázy

Relačné databázy pracujú spoľahlivo pri obmedzenom množstve dát [6]. Pri potrebe aplikácie s obrovským množstvom dát sa rozšíriteľnosť (angl. scalability) a schéma stávajú problémom [8]. Tento typ databáz ukladá dáta v tabuľkách zložených z riadkov a stĺpcov. Výhodou relačných databáz je možnosť jednoduchého vytvorenia prispôbeného pohľadu na dáta [7]. Z pohľadu ukladania textu, a celkovo dát s variabilnou štruktúrou, je nevýhoda relačných databáz v ťažko prispôsobiteľnej štruktúre.

4.5.2 Textové databázy

Textové databázy ukladajú dáta vo forme dokumentov, vďaka čomu ponúkajú vysoký výkon, horizontálnu rozšíriteľnosť a podporu pre ukládanie dát s variabilnou štruktúrou [8]. Uložené dokumenty môžu nadobúdať rôzne formáty, ako napríklad

JSON, BSON, XML a BLOB, ktoré poskytujú veľkú flexibilitu pre dáta. Každý záznam v takejto databáze preto môže mať inú štruktúru, napríklad počet alebo typ polí, čo šetrí úložným priestorom, keďže neobsahuje nepotrebné prázdne polia [8]. Dokumenty uľahčujú zmenu štruktúry záznamov jednoduchým pridaním, odstránením alebo zmenou typu poľa. Vďaka svojej štruktúre sú dokumenty ľahko namapovateľné na objekty z objektovo-orientovaných programovacích jazykov a odstraňujú tým potrebu pre použitie objektovo-relačnej mapovacej vrstvy. Dátový typ dokument v textových databázach vyhovuje požiadavkám na databázu pre náš systém. Dokáže jednoducho narábať s dátami s variabilnou štruktúrou, bez nároku na prebytočnú pamäť a s dobrou rozšíriteľnosťou.

4.5.3 Porovnanie a výber typu databázy

Porovnávame textové a relačné databázy. V oboch kategóriách sme vybrali jedného z najväčších predstaviteľov danej kategórie. Za relačné databázy to je *MySQL* a za textové databázy *MongoDB*. Porovnanie poskytovaných prvkov porovnávaných databáz je v tabuľke 2 Porovnanie poskytovaných prvkov.

Tabuľka 2: Porovnanie poskytovaných prvkov¹¹

	MySQL	MongoDB
Bohatý dátový model	Nie	Áno
Dynamická štruktúra	Nie	Áno
Dátové typy	Áno	Áno
Lokálnosť dát	Nie	Áno
Aktualizovanie polí	Áno	Áno
Ľahké pre programátorov	Nie	Áno
Komplexné transakcie	Áno	Nie
Audit	Áno	Áno
Auto-sharding	Nie	Áno

Bohatý dátový model (angl. Rich Data Model) je poskytovanie rozšírenej funkcionality dátovým modelom. Princípom *dynamickkej štruktúry* (angl. Dynamic Structure) je jednoduchá zmena štruktúry, pričom nemusí byť vôbec zadefinovaná

¹¹<https://www.mongodb.com/compare/mongodb-mysql>

a každý záznam môže mať odlišnú štruktúru. *Lokálnosť dát* (angl. Data Locality) znamená uchovávanie súvisiacich dát pokope. *Aktualizovanie polí* umožňuje vykonávať nad poliami operácie, ako sú inkrementácia podľa špecifikovaného množstva, vynásobenie hodnotou, premenovanie, aktualizácia iba ak je hodnota väčšia alebo menšia ako špecifická hodnota a ďalšie. *Audit* (angl. Auditing) je funkcionality, ktorá umožňuje administrátorom a používateľom sledovať aktivity systému. *Auto-sharding* zabezpečuje zabránenie poklesu priepustnosti operácií čítania a zapisovania pri náraste dát ukladaním dát automaticky na viacero strojov.

MongoDB má vlastnú konvenciu názvov svojich častí. Tie sa v niektorých prípadoch líšia s názvami relačných databáz. Rozdiely sú zobrazené v tabuľke 3 Porovnanie používaných pojmov.

Tabuľka 3: Porovnanie používaných pojmov [6]

MySQL	MongoDB
Databáza	Databáza
Tabuľka	Kolekcia
Index	Index
Riadok	BSON dokument
Stĺpec	BSON pole (angl. field)
Spojenie	Vnorené dokumenty a prepojenie
Primárny kľúč	Primárny kľúč
Zoskupenie	Agregácia

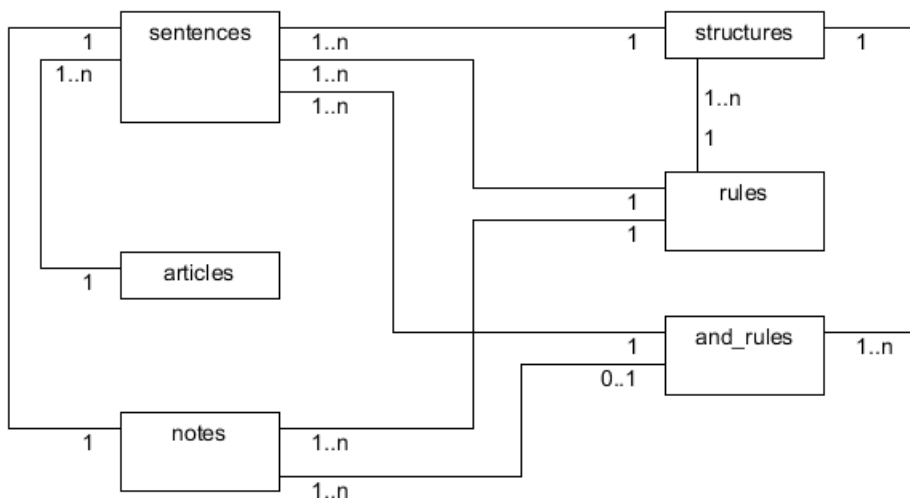
Pri ukladaní textov a iných častí systému v databáze potrebujeme dynamickú štruktúru z dôvodu variability štruktúry ukladaných dát. Namapovanie dát na pevne stanovenú štruktúru a vzťahy relačných databáz by bolo veľmi náročné. Preto je výhodnejšie uchovávať dáta pokope v dynamickej štruktúre. Z bodov *dynamická štruktúra* a *lokálnosť dát* z tabuľky 2 Porovnanie poskytovaných prvkov jasne vidno, že pre tento účel je textová databáza vhodnejšia. *Bohatý dátový model* je rovnako veľkou výhodou. V neposlednom rade *MongoDB* automaticky podporuje *sharding*, ktorý sa dá spraviť aj v databáze *MySQL*, ale za cenu výkonnosti. Z týchto dôvodov sme sa rozhodli pre použitie textovej databázy v našom systéme.

4.6 Návrh uchovávania textov v databázach

Ukladané dáta sa dajú rozdeliť do niekoľkých samostatných kolekcií. Sú to:

- rules,
- sentences,
- notes
- structures,
- articles,
- and rules.

Prepojenia medzi jednotlivými kolekciami sú zobrazené na Obr. 11. Nasledujúce časti stručne opisujú každú kolekciu. Všetky kolekcie obsahujú okrem polí špecifických pre danú kolekciu, aj polia časových značiek označujúcich vytvorenie a aktualizáciu záznamu.



Obr. 11: Databázový model

4.6.1 Kolekcia articles

V kolekcií *articles* sa ukladajú spracovávané texty.

Kolekcia obsahuje textové pole *text* na uloženie textu v pôvodnom tvare. Model kolekcie *articles* je zobrazený na Obr. 12.

Pole	Typ
_id	ObjectId
text	String
created_at	DateTime
updated_at	DateTime

Obr. 12: Model kolekcie *articles*

4.6.2 Kolekcia *notes*

Kolekcia *notes* uchováva vytvorené poznámky z viet.

Obsahuje textové pole *text* s hodnotou poznámky a dve referencujúce polia. Jedno sa odkazuje do kolekcie *rules* na pravidlo, ktoré bolo použité na vytvorenie poznámky. Druhé referencuje použité and-pravidlo v kolekcií *and_rules*. Toto pole môže byť prázdne, ak sa and-pravidlo pri vytváraní poznámky nepoužilo. Na Obr. 37 je vyobrazený model kolekcie *notes*.

Pole	Typ
_id	ObjectId
text	String
rule_ref_id	ObjectId
and_rule_ref_id	ObjectId
created_at	DateTime
updated_at	DateTime

Obr. 13: Model kolekcie *notes*

4.6.3 Kolekcia sentences

V nasledujúcej kolekci *sentences* sa ukladajú spracované vety aj s informáciami o článku, pravidlách a poznámke.

Kolekcia sa skladá z textového polia *text* uchovávajúce hodnotu vety a piatich referencujúcich polí *article_ref_id*, *structure_ref_id*, *rule_ref_id*, *and_rule_ref_id* a *note_id*. *Article_ref_id* odkazuje na článok z kolekcie *articles*, ktorého súčasťou je daná veta. Pole *structure_ref_id* odkazuje do kolekcie *structures*, ktoré reprezentuje štruktúru vety. Nasledujúce polia *rule_ref_id* a *and_rule_ref_id* odkazujú na použité pravidlo a and-pravidlo pri spracovávaní vety, v tomto poradí. Pole *note_ref_id* odkazuje na poznámku z kolekcie *notes*, ktorá bola vytvorená z vety.

Model tejto kolekcie je načrtnutý na Obr. 38.

Pole	Typ
<i>_id</i>	ObjectId
<i>text</i>	String
<i>article_ref_id</i>	ObjectId
<i>structure_ref_id</i>	ObjectId
<i>rule_ref_id</i>	ObjectId
<i>and_rule_ref_id</i>	ObjectId
<i>note_ref_id</i>	ObjectId
<i>created_at</i>	DateTime
<i>updated_at</i>	DateTime

Obr. 14: Model kolekcie *sentences*

4.6.4 Kolekcia structures

V kolekcií *structures* sú uložené štruktúry viet a pravidiel. Štruktúra je zložená hlavne zo závislostí, tokenov, názvoslovných značiek, indexov a iné.

Kolekcia sa skladá z jedného pola *structure_data*. Toto pole je zoznam dokumentov, obsahujúcich vyššie spomenuté údaje. Dokument v tomto zozname obsahuje textové pole *relation_name* s názvom vzťahu závislosti a zoznam doku-

mentov závislostí *dependencies* s týmto názvom vzťahu. Dokument v zozname *dependencies* sa skladá z polí *governor* a *dependent* typu dokument, celočíselného pola *position* uchovávajúceho pozíciu závislosti vo vete alebo poznámke, *comparison_type*, ktoré je celočíselnou reprezentáciou typu porovnania a pola *token_type*, ktoré je taktiež celočíselnou reprezentáciou typu tokenu. Dokumenty polí *governor* a *dependent* obsahujú údaje o prislúchajúcich tokenoch závislosti. V textovom poli *POS* sa ukladá značka slovného druhu, pole *index* uchováva index tokenu vo vete, *ner* je textové pole reprezentujúce názvoslovnú entitu tokenu a v poli *lemma* je textová reprezentácia lemy tokenu.

Celý model kolekcie je zobrazený na Obr. 39.

Pole	Typ																																						
_id	ObjectId																																						
structure_data	Array of Documents <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>relation_name</td><td>String</td></tr> <tr> <td>dependencies</td><td>Array of Documents <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>governor</td><td>Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table> </td></tr> <tr> <td>dependent</td><td>Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table> </td></tr> <tr> <td>position</td><td>Integer</td></tr> <tr> <td>comparison_type</td><td>Integer</td></tr> <tr> <td>token_type</td><td>Integer</td></tr> </table> </td></tr> </table>	Pole	Typ	relation_name	String	dependencies	Array of Documents <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>governor</td><td>Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table> </td></tr> <tr> <td>dependent</td><td>Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table> </td></tr> <tr> <td>position</td><td>Integer</td></tr> <tr> <td>comparison_type</td><td>Integer</td></tr> <tr> <td>token_type</td><td>Integer</td></tr> </table>	Pole	Typ	governor	Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table>	Pole	Typ	POS	String	index	Integer	ner	String	lemma	String	dependent	Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table>	Pole	Typ	POS	String	index	Integer	ner	String	lemma	String	position	Integer	comparison_type	Integer	token_type	Integer
Pole	Typ																																						
relation_name	String																																						
dependencies	Array of Documents <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>governor</td><td>Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table> </td></tr> <tr> <td>dependent</td><td>Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table> </td></tr> <tr> <td>position</td><td>Integer</td></tr> <tr> <td>comparison_type</td><td>Integer</td></tr> <tr> <td>token_type</td><td>Integer</td></tr> </table>	Pole	Typ	governor	Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table>	Pole	Typ	POS	String	index	Integer	ner	String	lemma	String	dependent	Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table>	Pole	Typ	POS	String	index	Integer	ner	String	lemma	String	position	Integer	comparison_type	Integer	token_type	Integer						
Pole	Typ																																						
governor	Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table>	Pole	Typ	POS	String	index	Integer	ner	String	lemma	String																												
Pole	Typ																																						
POS	String																																						
index	Integer																																						
ner	String																																						
lemma	String																																						
dependent	Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table>	Pole	Typ	POS	String	index	Integer	ner	String	lemma	String																												
Pole	Typ																																						
POS	String																																						
index	Integer																																						
ner	String																																						
lemma	String																																						
position	Integer																																						
comparison_type	Integer																																						
token_type	Integer																																						
created_at	DateTime																																						
updated_at	DateTime																																						

Obr. 15: Model kolekcie structures

4.6.5 Kolekcia rules

Rules je kolekcia, do ktorej sa ukladajú pravidla na vytvorenie poznámok z viet. Vďaka databázovému modelu na Obr. 11 a prepojeniam medzi kolekciami je táto kolekcia minimalistická.

Skladá sa z dvoch polí. Pole *sentence_terminators* je zoznam čísel, ktoré reprezentujú konce viet v poznámke. Referencujúce pole *structure_ref_id* odkazuje do kolekcie *structures* na štruktúru, ktorou sa má prípadná veta spracovať. Model kolekcie *rules* je vyjadrený Obr. 40.

Pole *sentence_terminators* zväčša obsahuje jeden záznam. Napríklad pri vete „*The president of the Slovak Republic is Andrej Kiska.*” a poznámke z tejto vety v tvare „*President is Kiska.*” bude obsahovať jeden záznam: 3. Číslo 3, lebo koniec vety, v tomto prípade bodka, sa nachádza na tretej pozícii spomedzi tokenov vo vete. Číslovanie pozícií začína od nuly. V prípade ak veta je súvetie, zložené z viacerých jednoduchých viet, môže pole *sentence_terminators* obsahovať viacero záznamov, ak napríklad chceme z každej jednoduchej vety súvetia získať zjednodušenú vetu a vytvoriť tak zloženú poznámku, skladajúcu sa zo zjednodušených viet.

Pole	Typ
<i>_id</i>	ObjectId
<i>sentence_terminators</i>	Array of Integers
<i>structure_ref_id</i>	ObjectId
<i>created_at</i>	DateTime
<i>updated_at</i>	DateTime

Obr. 16: Model kolekcie *rules*

4.6.6 Kolekcia and rules

Posledná kolekcia uchováva pravidlá pre spracovanie vety a vytvorenie viacnásobnej poznámky z vety. Táto kolekcia je veľmi podobná kolekcií *rules* a obsahuje rovnaké polia doplnené o ďalšie špecifické pole.

Špecifické pole, o ktoré je kolekcia rozšírená oproti kolekcií *rules* je celočíselné pole *set_position*. Toto pole uchováva pozíciu množiny vo viacnásobnej poznámke. Model kolekcie je vyobrazený na Obr. 41.

Pole	Typ
_id	ObjectId
sentence_terminators	Array of Integers
set_position	Integer
structure_ref_id	ObjectId
created_at	DateTime
updated_at	DateTime

Obr. 17: Model kolekcie and rules

4.6.7 Zhrnutie kolekcií

Pri návrhu databázového modelu a kolekcií sme vychádzali z princípu jednoduchých kolekcií so zoskupením súvisiacich dát a oddelenia ich od zvyšku. Vďaka využívaniu viacerých, medzi sebou prepojených, kolekcií sme zabezpečili neduplikovanie dát, jednoduché vyhľadanie, napríklad viet ku článku a iné. Okrem toho nám tento model umožňuje ďalšiu funkcionality, ako napríklad aplikovanie jedného pravidla na viacero viet so zhodnou štruktúrou. Oddelenie dát do samostatných kolekcií nám uľahčuje aj prípadne neskoršie rozšírenie databázového modelu alebo zmenu konkrétnych kolekcií. Taktiež uľahčuje prípadné klastrovanie databázy, ak by bolo nutné, keďže každá kolekcia by mohla byť na samostatnom serveri.

4.7 Zhrnutie

Systém dokáže vytvoriť poznámku z viet na základe pravidla. Pravidlo je tvorené primárne zo závislostí, značiek slovných druhov a typov názvoslovných entít. Závislostí majú viacero využití pri tvorbe poznámok, hlavné z nich sú jednoznačná identifikácia informácie a pravidlo závisle od štruktúry vety. Z toho vyplýva,

že pravidlo je aplikovateľné na viacero viet s rovnakou štruktúrou. Štruktúru si vieme reprezentovať stromom alebo priamo vo vete. Informácie o spracovaných článkoch, vetách a vytvorených poznámkach a pravidlách sa ukladajú do textovej databázy s dôkladne navrhnutým modelom.

5 Systém na tvorbu poznámok

Pri implementácii systému na tvorbu poznámok vyberáme konkrétnu textovú databázu a hlavný nástroj na podporu spracovania prirodzeného jazyka. Riešime fundamentálne princípy používania pravidiel na tvorbu poznámok, ako vyhľadanie, aplikovanie, vytvorenie a úprava pravidla.

5.1 Použité technológie

Systém je implementovaný v programovacom jazyku C# a na grafické rozhranie je použitý framework WinForms. Postavený je nad nástrojom StanfordNLP (bližšie opísaný v časti 2.2.2 StanfordNLP), ktorý umožňuje jednoduchým spôsobom získať esenciálne informácie pre náš systém ohľadne textov, ako závislostí slov vo vetách, značky slovných druhov, názvoslovné entity, indexy a lemy slov. StanfordNLP je pomocou knižnice IKVM sprístupnený z jazyka Java do balíka Stanford.NLP.CoreNLP v jazyku C#. V systéme je použitá verzia 3.5.2 tohto nástroja aj knižnice Stanford.NLP.CoreNLP, spolu so zdrojovými kódmi modelov z originálneho nástroja. Databázová vrstva systému je implementovaná pomocou textovej databázy MongoDB, ktorá uľahčuje narábanie s textom a využíva vlastný dopytovací jazyk [8]. Na komunikáciu medzi databázou a systémom sa používa knižnica MongoDB.Driver a na prácu s databázovými štruktúrami, ako sú dokumenty, je potrebná knižnica MongoDB.Bson. Knižnica HtmlAgilityPack je využívaná na spracovávanie HTML kódu pri získavaní vstupných článkov priamo z webovej stránky.

5.2 Manažment dát

Pre vytvorenie poznámky z vety potrebujeme pravidlo, ktoré sa na vetu aplikuje. Aby sa vytvorila zodpovedajúca poznámka z vety, je nutné použiť vhodné pravidlo. Ak spracovávame doposiaľ nespracovanú vetu, je potrebné nájsť vhodné pravidlo na základe podobnosti spracovávanej vety so spracovanými vetami. Pri extrakcii informácií z vety daným pravidlom, s cieľom vytvorenia poznámky, sa musia vybrať správne informácie. Extrakcia musí fungovať pri udržaní dostatoč-

nej všeobecnosti, aby sa dané pravidlo dalo aplikovať na viacero viet s rovnakou štruktúrou, ale odlišným obsahom.

5.2.1 Vyhľadanie pravidla

Pri spracovávaní vety, pred vytvorením poznámky, aplikovateľné pravidlo musí byť vyhladané v databáze. Vyhľadá sa v databáze veta, ktorej štruktúra korešponduje so štruktúrou spracovávanej vety. Štruktúry oboch alebo viacerých viet musia obsahovať rovnakú množinu závislostí. To znamená rovnaký počet záznamov a záznamy s rovnakými názvami vzťahov závislostí v zozname dát štruktúry.

Podobná alebo zhodná veta je vyhladaná, ak hlavná podmienka je splnená. Avšak, splnenie tejto podmienky nezaručí vyhládanie len jednej, najpodobnejšej vety, ale môže vyhľadať viacero viet. V takom prípade sa vypočíta zhoda viet. Po určení zhody sa extrahuje pravidlo vety, s ktorou má spracovávaná veta najväčšiu zhodu.

Výpočet zhody viet

Zhoda viet pozostáva z troch častí:

- štruktúrálna časť,
- obsahová časť,
- hodnotová časť.

Štruktúrálna časť zhody odzrkadľuje percentuálnu zhodu dvoch štruktúr. Pri tejto zhode zisťujeme, či štruktúry obsahujú tokeny závislostí s nadradenými značkami slovných druhov a ich konkrétny počet. Štruktúrálna časť zhody znamená, že vety „*The president of the Slovak Republic is Andrej Kiska.*” a „*Andrej Kiska is the president of the Slovak Republic.*”, ale aj „*Miloš Zeman is the president of the Czech Republic.*” majú rovnakú štruktúru, bez ohľadu na hodnoty a pozície slov vo vete, pokiaľ obsahujú také isté závislosti s tokenmi s rovnakými nadradenými značkami slovných druhov. Určenie štruktúrálnej zhody pozostáva z niekoľkých krokov. Najskôr sa separátne počíta zhoda nadradených značiek slovných druhov nadradeného a podradeného tokenu. Týmito krokmi sa určí, či veta obsahuje ľubovoľnú závislosť s rovnakou nadradenou značkou slovného druhu na ľubovoľnom

tokene. V nasledujúcom kroku sa určí úplná zhoda závislosti s nadradenými značkami slovných druhov, to znamená zistenie, či veta obsahuje ľubovoľnú závislosť s konkrétnymi značkami slovných druhov na nadradenom a zároveň podradenom tokene. V poslednom kroku sa zisťuje zhoda počtu závislostí s rovnakými nadradenými značkami slovných druhov u nadradeného a podradeného tokenu.

Obsahová časť zhody zodpovedá percentuálnej zhode obsahu dvoch viet. Kontrolujú sa indexy slov, konkrétne značky slovných druhov a názvoslovné entity. Pri obsahovej zhode majú vety „*The president of the Slovak Republic is Andrej Kiska.*” a „*The president of the Czech Republic is Miloš Zeman.*” obsahovú zhodu, bez ohľadu na konkrétne slova na pozíciách vo vete, ak obsahujú rovnaké značky slovných druhov a reprezentujú ich zhodné názvoslovné entity. Výpočet obsahovej zhody sa skladá z viacerých krokov. Začína sa výpočtom zhôd značiek slovných druhov podradeného a nadradeného tokenu. Zhody indexov nadradeného a podradeného tokenu sú taktiež vypočítané separátne. Tak isto sa vypočítajú zhody aj názvoslovných entít. Tieto prvé kroky určia, či veta obsahuje ľubovoľnú závislosť s rovnakou značkou slovného druhu alebo indexom a ľubovoľnú závislosť s rovnakou názvoslovnou entitou. V nasledujúcom kroku je určená polovičná zhoda závislostí. Polovičná zhoda závislosti je zhoda značky slovného druhu a indexu nadradeného alebo podradeného tokenu. Polovičná zhoda sa vypočíta rovnako aj pre názvoslovné entity. Nakoniec, v poslednom kroku, počítame počet úplne zhodných závislostí. Úplná zhoda závislosti je zhoda značky slovného druhu a indexu nadradeného, a zároveň podradeného tokenu. Tak isto sa vypočíta úplná zhoda závislostí aj pre názvoslovné entity.

Posledná časť zhody, hodnotová časť zhody, reprezentuje úplnú zhodu dvoch viet. Veta má hodnotovú časť zhody len s totožnou vetou.

Každý krok, pri výpočte všetkých častí zhody, má priradené ohodnotenie. Ak je podmienka v kroku vyhodnotená ako správna, ohodnotenie kroku je pripočítané do finálnej hodnoty. Finálna hodnota je percentuálne ohodnotenie zhody. Ohodnotenie krokov odzrkadľuje dôležitosť daného kroku vo výpočte presnej zhody, pričom závisí od počtu závislostí a krokov, tak že finálna zhoda nemôže presiahnuť hodnotu

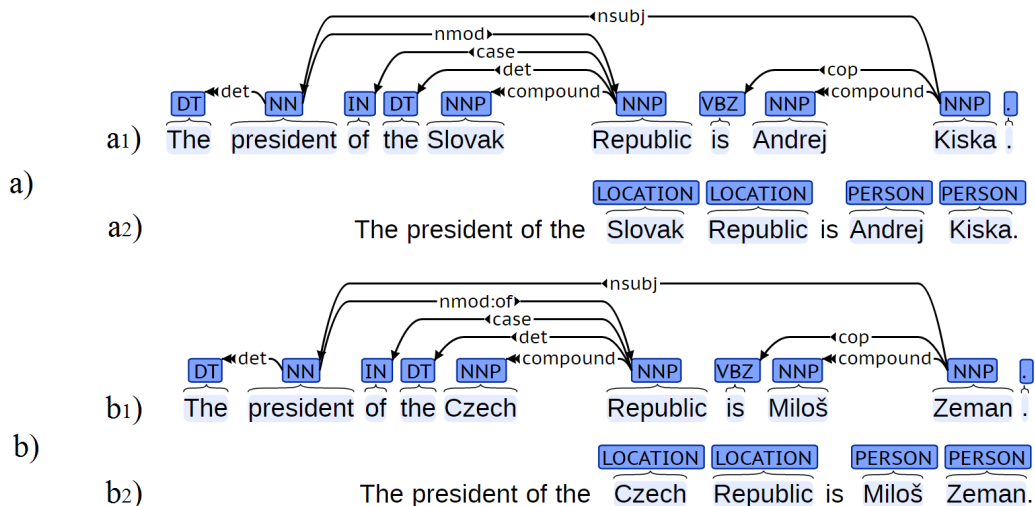
100%. Pseudokód 1 Výpočet zhody viet zobrazuje pseudo algoritmus výpočtu zhody. Konkrétny príklad je zobrazený na Obr. 18

Algoritmus 1 Výpočet zhody viet

```

1: procedure VYPOCETZHODYVIET(spracovávanáVeta, závislostiPorovnávanejVety)
2:   vypočítaj ohodnotenia krokov
3:   for all závislosti porovnávanej vety do
4:     for all závislosti spracovávanej vety do
5:       for all porovnaní do
6:         if aplikujPorovnanie(spracovávanáVeta, porovnanie, závislost) then
7:           do zhody na type porovnaní pripočítaj ohodnotenie kroku
   return zhoda

```



Obr. 18: Príklad určenia zhody viet

Predpokladajme situáciu z Obr. 18. V databáze máme uloženú spracovanú vetu *a* aj s pravidlom pre túto vetu. Spracovávame vetu *b*. Na časti *a1* Obr. 18 sú znázornené závislosti a na časti *a2* názvoslovné entity vety *a*. Na časti *b1* sú znázornené závislosti a na časti *b2* názvoslovné entity vety *b*. V tejto situácii potrebujeme vypočítať zhodu medzi vetami *a* a *b*.

Pri štruktúrálnej časti zhody sa prechádza cez všetky závislosti vety *a*. Prvá závislosť je závislosť so vzťahom *DET* a nadradeným tokenom s nadradenou značkou

slovného druhu *NN* a podradeným tokenom s nadradenou značkou slovného druhu *DT*. V prvom kroku sa pozrie, či veta *b* obsahuje ľubovoľnú závislosť s podradeným tokenom s nadradenou značkou slovného druhu *DT*. V ďalšom kroku, sa zistí, či veta *b* obsahuje ľubovoľnú závislosť s nadradeným tokenom s nadradenou značkou slovného druhu *NN*. Môže to byť aj iná závislosť, ako tá z prvého kroku. Pokračuje sa zistením úplnej zhody závislostí. Určí sa, či veta *b* obsahuje ľubovoľnú závislosť s podradeným tokenom s nadradenou značkou slovného druhu práve *DT* a zároveň s nadradeným tokenom s nadradenou značkou slovného druhu *NN*. Takto sa iteruje cez všetky závislosti vety *a*. Na záver sa určí zhoda počtu závislostí s rovnakou štruktúrou. Napríklad veta *a* obsahuje práve dve závislosti, ktoré majú na podradenom tokene nadradenú značku slovného druhu *DT* a na nadradenom tokene nadradenú značku slovného druhu *NN*. Zistí sa, či aj veta *b* obsahuje práve dve takéto závislosti.

Následne sa určuje obsahová časť zhody. Prvá závislosť vo vete *b* je so vzťahom *det*, nadradeným tokenom so značkou slovného druhu *NN* a indexom 1 a podradeným tokenom so značkou slovného druhu *DT* a indexom 0. Token slova *Slovak* má názvoslovnú entity typu *LOCATION* - *lokácia*. Ak slovo nemá vyobrazený typ názvoslovnej entity, znamená to, že má názvoslovnú entity typu *OTHER* - *ostatné*. V prvom kroku pri určovaní obsahovej časti zhody zisťujeme, či veta *b* obsahuje závislosť so vzťahom *det* a tokenmi so značkou slovného druhu *NN* alebo *DT* a indexmi rovnými 0 alebo 1. Toto je separátny výpočet značiek slovných druhov a indexov. V tomto istom kroku sa tiež pozrie, či veta *b* obsahuje ľubovoľnú závislosť s podradeným tokenom s názvoslovnou entitou typu *ostatné* (názvoslovná entita tokenu *THE* vo vete *a*) a ľubovoľnú závislosť s nadradeným tokenom s názvoslovnou entitou typu *ostatné* (názvoslovná entita tokenu *president* vo vete *a*). V nasledujúcom kroku zisťujeme, či veta *b* obsahuje závislosť so vzťahom *det* a nadradeným alebo podradeným tokenom so značkou slovného druhu *NN* a indexom 1 alebo značkou slovného druhu *DT* a indexom 0. Toto je polovičná zhoda. V poslednom kroku hľadáme vo vete *b* závislosť so vzťahom *det* a nadradeným tokenom práve so značkou slovného druhu *NN* a indexom 1 a zároveň podradeným tokenom práve so značkou slovného druhu *DT* a indexom 0. Zároveň v tomto kroku sa zisťuje, či veta *b* obsahuje závislosť s podradeným

tokenom s názvoslovnou entitou typu *ostatné* a zároveň s nadradeným tokenom s názvoslovnou entitou typu *ostatné*. Iterácia pokračuje s nasledujúcou závislosťou, pokiaľ sa nevyhodnotia všetky.

Pri určení hodnotovej časti zhody sa porovnávajú texty „*The president of the Slovak Republic is Andrej Kiska.*” a „*The president of the Czech Republic is Miloš Zeman.*” a určí sa, či sú zhodné.

Aplikovaním určenia zhody viet medzi vetami *a* a *b* zistíme, že veta *b* má štrukturálnu časť zhody s vetou *a* 100%. Tak isto má obsahovú časť zhody rovnú 100%. Hodnotová časť zhody je 0%.

Nadradená značka slovného druhu

Pod nadradenou značkou slovného druhu sa chápe značka slovného druhu zoskupujúca množinu značiek slovných druhov, do ktorej značka slovného druhu patrí.

Napríklad značka slovného druhu VBD (Verb, past tense - sloveso v minulom čase) patrí medzi skupinu značiek slovných druhov sloves {*VB, VBD, VBG, VBN, VBP, VBZ*}. Z toho vyplýva, že nadradená značka slovného druhu *VBD* je *VB* (Verb - sloveso).

5.2.2 Aplikovanie pravidiel

Procesom vyhľadania pravidla (viď. 5.2.1 Vyhľadanie pravidla) sme získali pravidlo na spracovanie vety. Aplikáciou pravidla na vetu vytvoríme poznámku.

Proces aplikovania pravidla na vetu s cieľom vytvorenia poznámky má viacero krokov. Pre všetky závislosti zo zoznamu *dát štruktúry* pravidla, príslušná závislosť je vyhľadaná v spracovávanej vete. Pri vyhľadávaní príslušnej závislosti sa závislosti neporovnávajú, okrem iného, na základe značiek slovných druhov svojich tokenov, ale podľa nadradených značiek slovných druhov (viď. Nadradená značka slovného druhu na strane 36) svojich tokenov. Tento spôsob vyhľadávania nám umožňuje aplikovať jedno pravidlo na viacero viet, ako už bolo spomenuté v časti 4.3 Použitie závislostí pri tvorbe poznámok. Avšak, môže to spôsobiť vy-

hľadanie viac ako jednej príslušnej závislosti. Preto musí byť vypočítaná zhoda závislostí. Po vypočítaní zhody závislostí a získaní závislosti s najväčšou zhodou, slovo korešpondujúce s tokenom, ktorý sa má z danej závislosti vybrať, sa pridá do poznámky na pozíciu pozície závislosti. Po spracovaní všetkých závislostí, posledné minoritné úpravy sú vykonané nad poznámkou, ako rozdelenie na viacero viet, ak tak určovalo pravidlo, kapitalizácia prvých písmen viet poznámky a iné. Pseudokód aplikovania pravidla na vetu s cieľom vytvoriť poznámku je zobrazený na algoritme 2 Aplikovanie pravidla.

Algoritmus 2 Aplikovanie pravidla

```

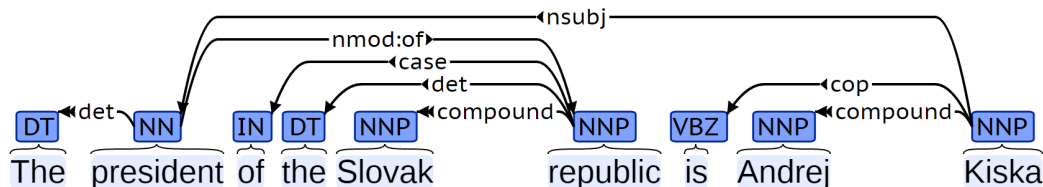
1: procedure APLIKUJPRavidlo(veta, pravidlo)
2:   poznámka  $\leftarrow$  vytvor prázdnu poznámku
3:   for all závislost in pravidlo do
4:     závislost  $\leftarrow$  nájdizávislost(veta, závislost pravidla)
5:     if závislost existuje then
6:       pridaj závislost do poznámky
7:   rozdeľ poznámku na vety podľa pravidla
   return poznámka

```

Pre vetu „*The president of the Slovak republic is Andrej Kiska.*” nám nástroj Stanford CoreNLP poskytne závislosti vyobrazené na Obr. 19. Ak na túto vetu aplikujeme pravidlo so štruktúrou v tvare zobrazenej na Obr. 20, výsledná poznámka bude „*President is Kiska.*”.

Aplikovanie pravidla prebieha nasledovným spôsobom. Prechádzajú sa všetky závislosti v štruktúre pravidla. Prvá závislosť v štruktúre pravidla je závislosť so vzťahom *nsubj* na pozícií jedna a podradeným korešpondujúcim tokenom. Má podradený token so značkou slovného druhu *NN*, typom názvoslovnej entity *OTHER* - *ostatné*, lemov *President* a indexom dva. Nadradený token má značku slovného druhu *NNP*, názvoslovnú entitu *PERSON* - *osoba*, lemu *Kiska* a index deväť. Takáto závislosť sa vyhľadá v štruktúre vety medzi závislosťami na Obr. 19. Závislosti sa vyhľadávajú podľa zhody všetkých informácií o ich tokenoch a vzťahu medzi nimi. Ak pre danú závislosť vyhovuje viacero závislostí, pomocou výpočtu zhody závislostí vyberieme tú s najväčšou zhodou. V tomto prípade vidíme, že vyhovujúca závislosť je len jedna a to prvá závislosť *nsubj* medzi slovom na druhej pozícií *president* a posledným slovom, na deviatej pozícií *Kiska*.

Z tejto závislosti sa zoberie podradený token, keďže tak určuje pravidlo v stĺpci *typ tokenu*. Slovo *president* sa pridá do poznámky na pozíciu jedna. Rovnakým spôsobom sa prechádzajú a spracujú všetky závislosti v štruktúre pravidla a podľa nich sa extrahuje slovo z vety a pridá do poznámky.



Obr. 19: Závislosti jednoduchej vety

Konca viet	Závislosti								
3	Vzťah	Pozícia	Porovnanie	Typ tokenu	Tokeny				
	nsubj	1	-	podradený	Typ	POS	NER	Lemma	Index
					nadradený	NNP	PERSON	Kiska	9
					podradený	NN	OTHER	President	2
	nsubj	8	-	nadradený	nadradený	NNP	PERSON	Kiska	9
					podradený	NN	OTHER	President	2
	cop	6	-	podradený	nadradený	NNP	PERSON	Kiska	9
					podradený	VBZ	OTHER	be	7

Obr. 20: Príklad štruktúry pravidla

Výpočet zhody závislostí

Princíp výpočtu zhody závislostí je veľmi podobný s výpočtom zhody viet zo sekcie 5.2.1 Vyhľadanie pravidla. Porovnávajú sa vždy nadradené aj podradené tokeny. Porovnanie má niekoľko krokov. Začína sa s porovnaním značiek slovných druhov. Pokračuje sa názvoslovnou entitou, indexom, lemov a nakoniec sa porovnáva vzdialenosť pozícií tokenov vo vetách. Každý krok je príslušne ohodnotený a ak porovnanie bolo úspešné, ohodnotenie sa pripočíta k finálnej hodnote reprezentujúcej percentuálnu zhodu závislostí.

5.2.3 Vytvorenie pravidla

Ak nám proces vyhľadania pravidla nevyhľadal žiadne pravidlo, znamená to, že sme doposiaľ nespracovávali takú istú alebo podobnú vetu. V tomto prípade sú

použité statické pravidlá na spracovanie vety a výstupom bude poznámka.

Zo závislostí pôvodnej vety a informácií o ich tokenoch sa vytvorí nový záznam o štruktúre pôvodnej vety. Tak isto sa vytvorí aj nový záznam o štruktúre poznámky. Z poznámky sa vytvorí záznam o novej poznámke a následne sa z nej vytvorí zoznam koncov viet. Tento zoznam spolu so štruktúrou poznámky vytvorí záznam nového pravidla. Z pôvodnej vety sa vytvorí záznam o vete a prepojí sa so štruktúrou pôvodnej vety, článkom, ktorý obsahoval danú vetu a pravidlom, ktoré bolo vytvorené a použité a s poznámkou, ktorá vznikla z vety. Týmto vznikne nové pravidlo na spracovanie takej istej alebo podobnej vety ako sme práve spracovali.

5.2.4 Úprava pravidla

Systém umožňuje používateľovi upraviť vytvorenú poznámku. Tá sa da upraviť z množiny slov pôvodnej vety, pričom sa dajú ľubovoľne usporiadať a definovať ľubovoľný počet viet, na ktoré poznámku rozdeliť. Úpravou poznámky sa systém „naučí“ nové pravidlo alebo upraví existujúce pravidlo. Ktorá z týchto dvoch akcií sa vykoná sa rozhoduje podľa zhody, ktorá bola určená pri spracovávaní pôvodnej vety poznámky.

Ak je štruktúrálna časť zhody menšia ako 100%, tým pádom aj obsahová a hodnotová zhoda je pod 100%, tak sa systém naučí nové pravidlo. Znamená to, že systém doposiaľ nespracovával vetu s rovnakou štruktúrou, a tak použil pravidlo podľa vety s najpodobnejšou štruktúrou a po úprave poznámky sa naučí, ako spracovať vetu s danou štruktúrou.

V prípade, že štruktúrálna časť zhody je 100%, ale obsahová nie je úplná a tým pádom ani hodnotová, znamená to, že systém spracoval vetu s rovnakou štruktúrou, ale iným obsahom. V tomto prípade systém pozná ako spracovať vetu s danou štruktúrou, takže sa štruktúra pravidla upraví podľa vykonaných zmien. Vytvorí sa nový záznam o poznámke a aj o pôvodnej vete a prepoja sa na upravené pravidlo, pomocou ktorého bude systém vedieť spracovať viacero viet.

Pri 100% štruktúrálnej, obsahovej aj hodnotovej časti zhody, systém spracoval identickú vetu, akú už v minulosti spracoval. Preto sa zmeny na vytvorenej poznámke prejavajú pri upravení pravidla. Tak isto sa upraví aj vytvorená poznámka, aby reflektovala vykonané zmeny.

5.3 Zhrnutie

Databázová vrstva nášho systému je implementovaná pomocou textovej databázy MongoDB. Ako hlavný nástroj na podporu spracovania prirodzeného jazyka využíva nástroj StanfordNLP, pre jednoduché získanie esenciálnych informácií o textoch. V procese vyhľadania pravidla je potrebné určiť zhodu štruktúry spracovávanej vety so štruktúrami viet v databáze, z dôvodu vyhľadania najvyhovujúcejšieho pravidla. Určenie zhody sa delí na tri časti, a to štruktúrna, obsahová a hodnotová. Pri aplikovaní pravidla sa počíta zhoda závislostí, kvôli udržaniu dostatočnej všeobecnosti pravidla. Po vytvorení poznámky je používateľ schopný si danú poznámku upraviť podľa vlastných predstáv a systém sa naučí nové, alebo upraví existujúce pravidlo, ako spracovávať vetu, z ktorej poznámka vznikla.

6 Experimenty

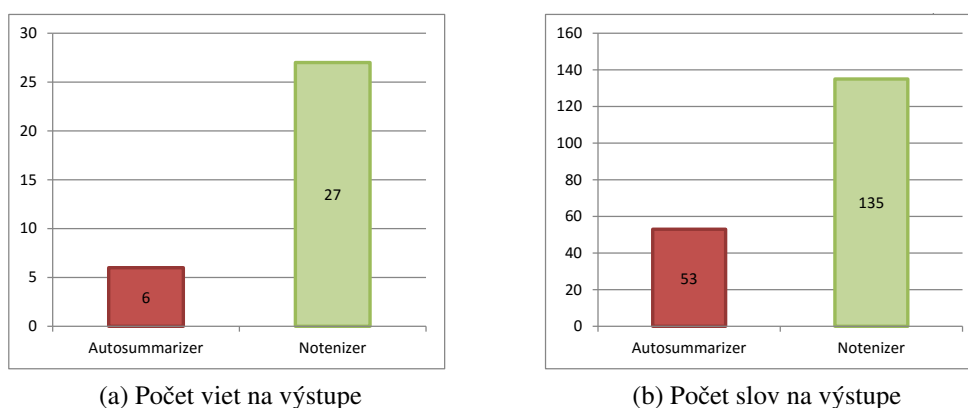
Na systéme boli vykonané tri experimenty. V prvom experimente bol systém porovnaný so systémom na sumarizáciu textu. Druhý experiment overoval použitie pravidiel na základe štruktúry viet. Posledný bol používateľský experiment, počas ktorého bol náš systém testovaný reálnymi používateľmi.

6.1 Porovnanie systému

V prvom experimente sme náš systém použili na spracovanie troch článkov z wikipédie¹². Výsledky sme porovnali s Autosummarizer¹³, systémom zameraným na sumarizáciu textu, využívajúci extrakčnú sumarizáciu. Články obsahovali spolu 27 viet a 294 slov.

6.1.1 Výsledky

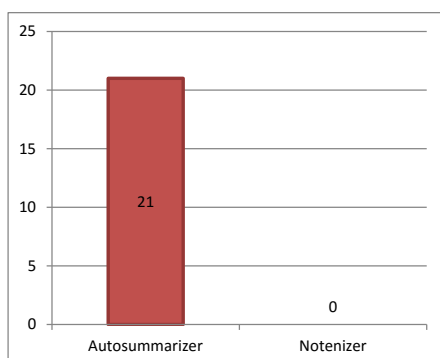
Porovnali sme počty viet a slov na výstupe systémov. Počty sú na Obr. 21. V časti *a* je zobrazený počet viet a v časti *b* je počet slov. V druhej časti porovnania sme sa zamerali na prístup systémov k spracovaniu viet a slov. Na Obr. 22 v časti *a* je zobrazený počet viet, ktoré systémy nespracovali a v časti *b* je priemerný počet eliminovaných irelevantných slov vo výslednej vete.



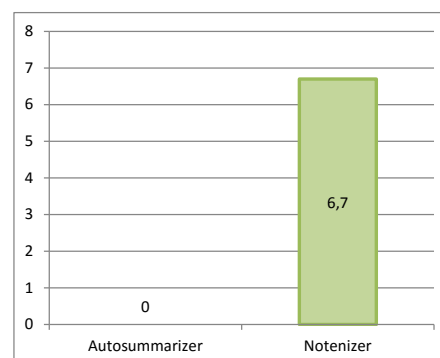
Obr. 21: Výstupy porovnaných systémov

¹²www.simple.wikipedia.org

¹³www.autosummarizer.com



(a) Počet nespracovaných viet



(b) Priemerný počet eliminovaných irelevantných slov vo výslednej vete

Obr. 22: Spracovanie informácií porovnanými systémami

6.1.2 Vyhodnotenie porovnania

Autosummarizer systém dosahoval lepšie výsledky v menšom počte viet a slov na výstupe, ale kvôli tomu vynechal veľa relevantných informácií. Náš systém na výstupe zobrazoval väčšie množstvo slov a viet, avšak spracoval všetky vety a eliminoval z nich irelevantné informácie, slová.

6.2 Použitie pravidiel

Otestovali sme aplikovateľnosť pravidiel závislých od štruktúry vety. Pri spracovaní 340 viet z tridsiatich článkov bolo potrebné vytvoriť a použiť 263 pravidiel. Z toho vyplýva, že pri 77 vetách bolo aplikované pravidlo na základe štruktúry vety a nebolo potrebné vytvárať nové. To predstavuje približne 22,65% všetkých viet, čím sa nám podarilo práve takéto množstvo pravidiel ušetriť.

6.3 Používateľský experiment

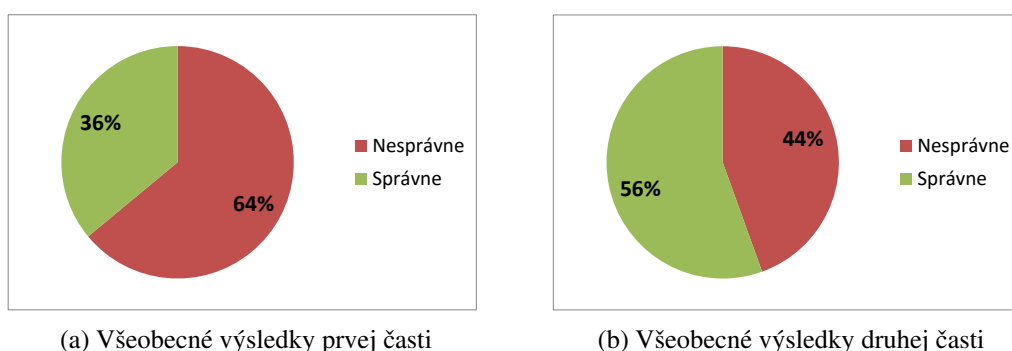
Na používateľskom experimente sa zúčastnilo pätnásť študentov, ktorí v systéme spracovali článok a získali z neho poznámky. Množina článkov experimentu pozostávala z článkov z wikipédie¹⁴ o štátoch z Európy. Účastník experimentu si zvolil článok, ktorý sa následne spracoval. Experiment sa skladal z dvoch častí.

¹⁴www.simple.wikipedia.org

V prvej časti sa článok vybraný účastníkom spracoval nad databázou, ktorá obsahovala počet pravidiel z piatich vopred spracovaných článkov. V druhej časti sa ten istý článok spracoval nad databázou s tridsiatimi vopred spracovanými článkami. Účastník v oboch častiach experimentu vytvorené poznámky z viet článku ohodnotil, prípadne upravil. Všetci účastníci testovali systém nad databázami s rovnakými dátami. Počas experimentu bolo spracovaných pätnásť článkov s 236 vetami.

6.3.1 Výsledky

Na Obr. 23 sú zobrazené pomery správnych a nesprávnych vytvorených poznámok. Na časti *a* tohoto obrázka vidno pomer pri spracovaní článku v prvej časti experimentu a v časti *b* je pomer z druhej časti experimentu.



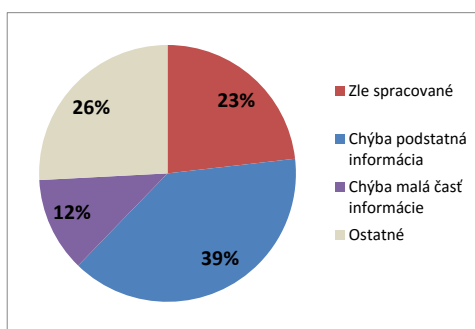
Obr. 23: Porovnanie všeobecných výsledkov oboch častí experimentu

Správne aj nesprávne poznámky boli podľa hodnotení účastníkov kategorizované do štrnástich kategórií. Najpočetnejšie z nich sú pre nesprávne poznámky *zlé spracované*, *chýba podstatná informácia*, *chýba mala časť informácie* a pre správne poznámky sú to *správne*, *OK - zlý slovosled*, *OK - subjektívne*.

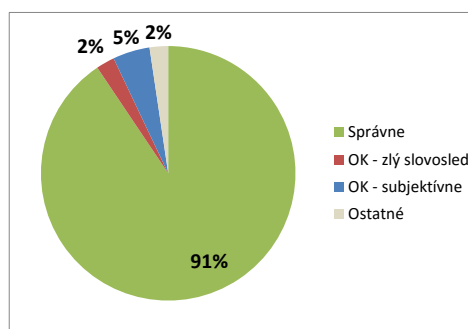
Zle spracované znamená, že systém zle spracoval vetu a vytvorená poznámka neobsahovala takmer žiadnu informáciu a nedávala zmysel. V kategórii *chýba podstatná informácia* sú zaradené poznámky, ktoré boli korektne vytvorené, avšak chýbala v nich podstatná informácia z vety potrebná na jej pochopenie. V poznámkach, ktoré obsahovali veľkú časť podstatnej informácie, prípadne celú podstatnú informáciu, ale chýbala v nej malá časť informácie z pôvodnej vety potrebná na

to, aby sa dala považovať za správnu poznámku, boli zaradené do kategórie *chýba mala časť informácie*. Do kategórie *správne* boli zaradené poznámky, ktoré boli správne vytvorené. Kategória *OK - zlý slovosled* obsahuje poznámky, ktoré obsahovali celú podstatnú informáciu z vety, ale slovosled vety bol zlý. Poznámky, ktoré boli označené za správne, ale účastník sa vyjadril, že si vie predstaviť, že pre niekoho by daná poznámka nemusela byť správna, boli zaradené do kategórie *OK - subjektívne*.

Na Obr. 24 a Obr. 25 sú zobrazené percentuálne zastúpenia kategórií pri prvej a druhej časti experimentu, v tomto poradí.

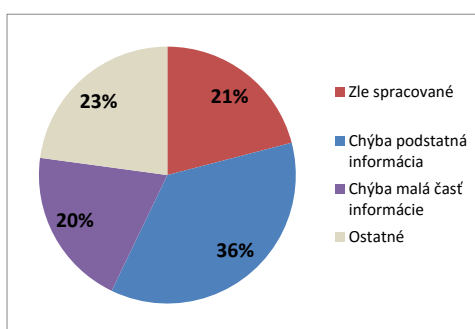


(a) Kategórie nesprávnych poznámok

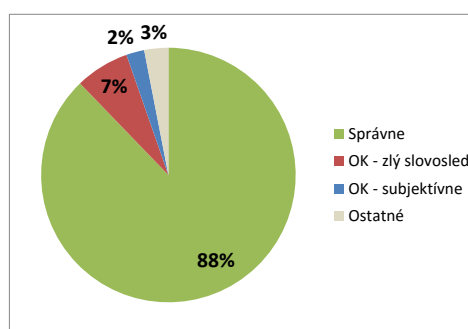


(b) Kategórie správnych poznámok

Obr. 24: Detailné výsledky prvej časti experimentu



(a) Kategórie nesprávnych poznámok



(b) Kategórie správnych poznámok

Obr. 25: Detailné výsledky druhej časti experimentu

6.3.2 Vyhodnotenie experimentu

Tvorba vyhovujúcich poznámok je subjektívna aktivita a preto hodnotenia účastníkov experimentu boli subjektívne a ovplyvnené ich predstavou o tvorbe poznámok. Avšak subjektívne hodnotenia boli objektívne roztriedené do kategórií a vyhodnotené. Z výsledkov je vidno 20% zlepšenie z pohľadu počtu správne vytvorených poznámok pri použití databázy s väčším počtom pravidiel. V druhej časti je oproti prvej časti znížený počet zle spracovaných poznámok a poznámok, v ktorých chýbala podstatná informácia. Zväčšilo sa percentuálne zastúpenie kategórie *chýbala časť informácie*, čo naznačuje zlepšenie aj v prípade nesprávne vytvorených poznámok. V správne vytvorených poznámkach je v druhej časti väčšie percentuálne zastúpenie kategórie *OK - zlý slovosled* oproti prvej časti. Taktiež nastali situácie kedy poznámka v prvej časti bola vytvorená správne a v druhej časti nesprávne. Spomenuté zhoršenia druhej časti oproti prvej súvisia s veľkou všeobecnosťou pravidla. Pravidlo sa aplikuje na vetu s rovnakou štruktúrou, ale rovnaká štruktúra môže mať viacero rôznych spracovaní.

80% účastníkov experimentu podalo pozitívnu spätnú väzbu a vyjadrilo sa, že by takýto alebo podobný systém na tvorbu poznámok používali.

6.4 Zhrnutie

Na systéme boli vykonané tri experimenty. V prvom experiment sme náš systém porovnali so systémom Autosummarizer, ktorý sa špecializuje na sumarizáciu textu. Z tohto experimentu sa ukázalo, že náš systém posielal na výstup viacero viet a slov ako porovnávaný systém, čo môže mať za dôsledok veľké množstvo dát na výstupe pri rozsiahlejších článkoch. Na druhej strane náš systém spracoval všetky vety a eliminoval v nich irelevantné informácie. Druhý systém vynechal vety pri spracovávaní a tak isto neeliminoval irelevantné informácie z viet.

Druhým experimentom sme dokázali, že pravidlá sú dostatočne všeobecné, aby sa dali použiť na viacero viet s rovnakou štruktúrou. Tým sa nám podarilo ušetriť podstatnú časť prípadných pravidiel, ktoré by systém musel vytvoriť, ak by pre každú vetu potreboval samostatné pravidlo.

Pomocou používateľského experimentu sme overili systémovú závislosť od počtu pravidiel. Čím viac pravidiel má systém k dispozícii, tým kvalitnejšie sprac-

cováva vety, vytvára lepšie poznámky a stáva sa viac personalizovaný pre používateľa. Zaznamenali sme podstatné zlepšenie pri použití väčšieho množstva pravidiel v systéme. Z experimentu vyplynulo, že aj keď je poznámka spracovaná nesprávne, neznamená to, že je úplne nepoužiteľná pre používateľa. V podstatnej časti nesprávne vytvorených poznámok chýbala iba mala časť informácie z vety na to, aby bola úplne pochopiteľná a použiteľná pre používateľa. Počas experimentu sme odhalili situácie, kedy sa veta pri menšom počte pravidiel spracovala správne a pri väčšom počte nesprávne. Za dôsledok to má veľká všeobecnosť pravidla a je potrebné, aby bolo viac reštriktívne.

7 Zhodnotenie

V práci sme zanalyzovali spracovanie prirodzeného jazyka, pričom sme sa zamerali na oblasti dôležité z hľadiska spracovania učebných textov. Vyskúšali a otestovali sme nástroje na spracovanie prirodzeného jazyka a nástroje na správu paralelných textov. Navrhli sme systém na spracovanie učebných textov. Systém vytvára poznámky z viet vstupného textu podľa pravidiel. Pravidlo určuje ako danú vetu spracovať, ktoré informácie sú podstatné a je naviazané sa štruktúru vety. Tým sme zabezpečili, že jedno pravidlo je aplikovateľné na viacero podobných viet. Pravidlo sa skladá z viacerých informácií, ale najmä zo závislostí slov vo vete. Ak veta obsahuje spojku a alebo informácie oddelené čiarkami, dokáže z nej vytvoriť viacnásobnú poznámku. Štruktúru vieme reprezentovať stromom alebo priamo vo vete. Okrem tvorby poznámok systém určuje názvoslovné entity deviatich kategórií. Databázový model využíva textovú databázu. Pri návrhu databázového modelu sme vychádzali z princípu jednoduchých kolekcí so zoskupením súvisiacich dát a oddelenia ich od zvyšku.

Navrhnutý systém sme implementovali z využitím textovej databázy MongoDB a nástroja StanfordNLP na spracovanie prirodzeného jazyka a získania esenciálnych informácií pre náš systém, ako závislosti, značky slovných druhov, názvoslovné entity a ďalšie. Pri implementácii sme vyriešili fundamentálne otázky o práci s pravidlami, ktoré sú vyhľadanie, aplikovanie, vytvorenie a úprava pravidla. Pri vyhľadaní aplikovateľného pravidla sa určuje zhoda spracovávanej vety a viet v databáze. Zhoda sa skladá zo štruktúrálnej, obsahovej a hodnotovej časti. V procese aplikovania pravidla a vytvorenia poznámky sa určuje zhoda závislostí pre udržanie dostatočnej všeobecnosti pravidla. Vytvorenú poznámku dokáže používateľ interaktívne upraviť, čím určuje, ktoré informácie sú pre neho relevantné.

Na systéme sme vykonali tri experimenty. V prvom experimente sme systém porovnali so systémom na sumarizáciu textu. Zistili sme, že náš systém spracováva všetky vety a eliminuje priemerne väčší počet irelevantných slov za cenu väčšieho počtu výstupných informácií. Pri druhom experimente sme overovali použitie pravidiel na viacero viet s rovnakou štruktúrou. Potvrdilo sa použitie už existujúcich pravidiel pre približne 22,65% viet. V poslednom používateľskom experimente sme verifikovali zlepšenie systému pri použití väčšej východiskovej množiny pra-

vidiel. Taktiež sa nám podarilo odhaliť chyby súvisiace s veľkou všeobecnosťou pravidiel a zistiť, že medzi študentami je veľký dopyt po podobnom systéme.

O systéme sme publikovali článok a následne ho na konferencií IIT.SRC prezentovali. Dostali sme viacero pozitívnych ohlasov, dobrých nápadov a pripomienok, z ktorých niektoré sa nám podarilo zapracovať.

7.1 Možnosti rozšírenia systému

Systém ponúka priestor na jeho ďalší vývoj. Dostupných je niekoľko oblastí, v ktorých je možné ho ďalej rozvíjať.

Aby bolo vytváranie poznámok čo najefektívnejšie, je potrebné vstupný text predspracovať. Elimináciou nepodstatných častí textu, konkrétne viet, by sa výrazne zredukoval počet viet, ktoré musí systém alebo používateľ spracovať. Eliminácia nepodstatných viet by sa nemusela vykonávať známymi algoritmami, ale mohla by sa vykonávať napríklad na základe kľúčových slov v poznámkach. Podľa toho, ako by si používateľ upravoval poznámky a tým definoval pravidla, by sa systém vedel naučiť podľa výskytu kľúčových slov v poznámkach, ktoré informácie sú pre používateľa dôležité. Na základe toho by vedel určiť dôležitosť vety pre používateľa a eliminovať ju ak by nevyhovovala rozmedziu, ktoré by stanovovalo, ktoré vety eliminovať, a ktoré nie.

Ďalšou oblasťou rozvoja by mohlo byť definovanie si vlastných závislostí. Vďaka vlastným závislostiam by nebol systém odkázaný na závislosti slov vo vete a hľadanie vzoru vo vetách na vytvorenie poznámky. Mohol by byť modifikovaný na hľadanie iných vzorov, napríklad vo vetách.

Bolo by potrebné nastaviť pravidlu väčšiu špecifickosť, aby nebolo závisle iba od štruktúry, ale aj od obsahovej časti vety. Pri hľadaní správnej miery závislosti od obsahovej časti treba vziať do úvahy, aby pravidlo ostalo dostatočné všeobecné a aplikovateľné na viacero viet, zatiaľ čo by bolo dostatočné špecifické na to, aby nezmenilo tvorbu poznámok nie veľmi podobných viet.

Zoznam použitej literatúry

- [1] James F. Allen. Natural language processing. In *Encyclopedia of Computer Science*, pages 1218–1222. John Wiley and Sons Ltd., Chichester, UK, 2003.
- [2] Akshar Bharati and Vineet Chaitanya. *Natural language processing: A Paninian perspective*. Prentice Hall of India, New Delhi, 2004.
- [3] Volha Bryl, Claudio Giuliano, Luciano Serafini, and Katerina Tymoshenko. Supporting natural language processing with background knowledge: Co-reference resolution case. In *9th International Semantic Web Conference (ISWC2010)*, November 2010.
- [4] Marie catherine De Marneffe and Christopher D. Manning. Stanford typed dependencies manual, 2008.
- [5] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.
- [6] C. Gyorodi, R. Gyorodi, G. Pecherle, and A. Olah. A comparative study: Mongodb vs. mysql. In *Engineering of Modern Electric Systems (EMES), 2015 13th International Conference on*, pages 1–6, June 2015.
- [7] David Maier. *The Theory of Relational Databases*. Computer Science Press, 1983.
- [8] Ameya Nayak, Anil Poriya, and Dikshay Poojary. Article: Type of nosql databases and its comparison with relational databases. *International Journal of Applied Information Systems*, 5(4):16–19, March 2013. Published by Foundation of Computer Science, New York, USA.
- [9] Preeti and BrahmaleenKaurSidhu. Natural language processing. *Int.J.Computer Technology & Applications*, 2013.

A Používateľská príručka

A.1 Inštalácia

Nástroj nie je potrebné inštalovať. Na elektronickom médiu sa nachádza spustiteľný súbor *Notenizer.exe*, ktorý spustí nástroj.

A.2 Spustenie systému

Nástroj ponúka prácu cez dve rozhrania - príkazový riadok a grafické používateľské rozhranie. Na príkazovom riadku nie sú dostupné všetky vymoženosti systému, ako napríklad interaktívna úprava poznámok. Podľa zadaných argumentov sa spustenie a správanie systému prispôsobuje.

A.2.1 Možnosti spustenia systému

Argumenty systému:

-t, --text	Text, ktorý sa má spracovať.
-u, --url	URL adresa článku na wikipédii ¹⁵ .
--country	Spracuje článok o krajine z wikipédie ¹⁵ .
-d, --db	Názov databázy, ku ktorej sa pripojí.
-v	Ladiace a pomocné výpisy programu.
-c	Program zbehne jednorázovo na príkazovom riadku.
--help	Zobrazí túto pomocnú správu.

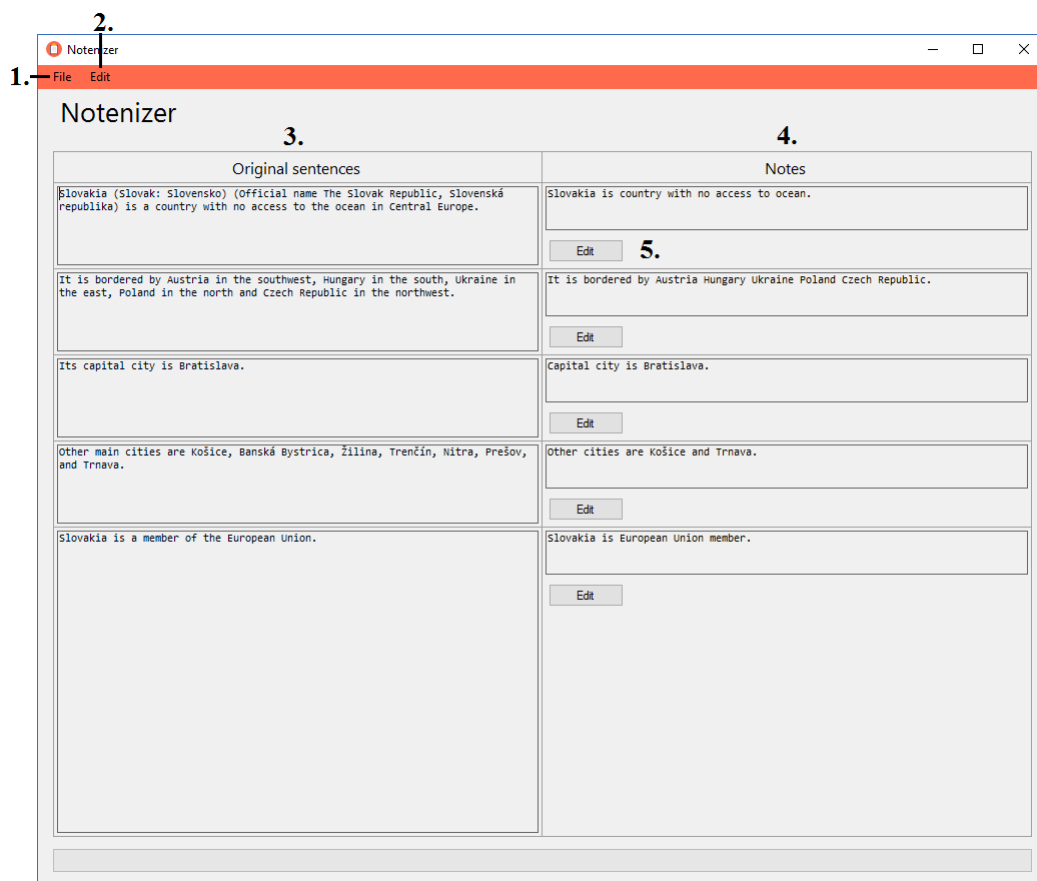
A.3 Používateľské rozhranie

Hlavné okno aplikácie (Obr. 26) pozostáva zo štyroch častí:

1. menu File (Obr. 27),
2. menu Edit (Obr. 28),
3. priestor na pôvodné vety,
4. priestor na poznámky,

¹⁵www.simple.wikipedia.org

5. zobrazí okno na editovanie poznámky (Obr. 31)



Obr. 26: Grafické rozhranie - Hlavné okno

Menu File (Obr. 27) ponúka viacero operácií s textami:

1. otvorí okno pre načítanie textu na spracovanie (Obr. 29),
2. získa text na spracovanie zo zvoleného textového súboru,
3. zobrazí okno na dotiahnutie textu z wikipédie (Obr. 30),
4. uloží vytvorené poznámky do textového súboru,
5. ukončí program.

1.	Open	Ctrl+O
2.	Open File	Ctrl+P
3.	Open Link	Ctrl+L
4.	Export	Ctrl+S
5.	Quit	Ctrl+Q

Obr. 27: Grafické rozhranie - Menu File

Menu Edit (Obr. 28) ponúka operácie:

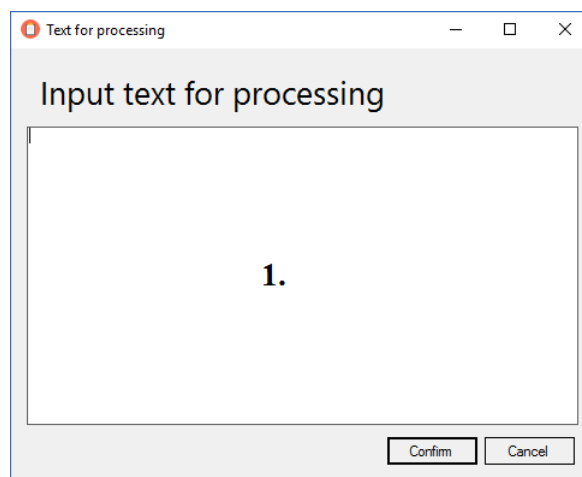
1. zo zobrazovacej plochy odstráni pôvodne vety a poznámky.

1.	Clear	Alt+C
----	-------	-------

Obr. 28: Grafické rozhranie - Menu Edit

Okno na načítanie textu na spracovanie (Obr. 29) obsahuje:

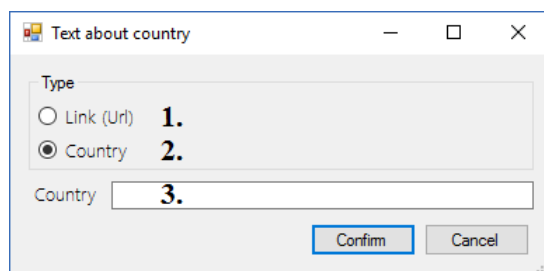
1. textovú plochu na načítanie textu na spracovanie.



Obr. 29: Grafické rozhranie - Okno na načítanie textu

Okno na dotiahnutie textu (Obr. 30) sa skladá z:

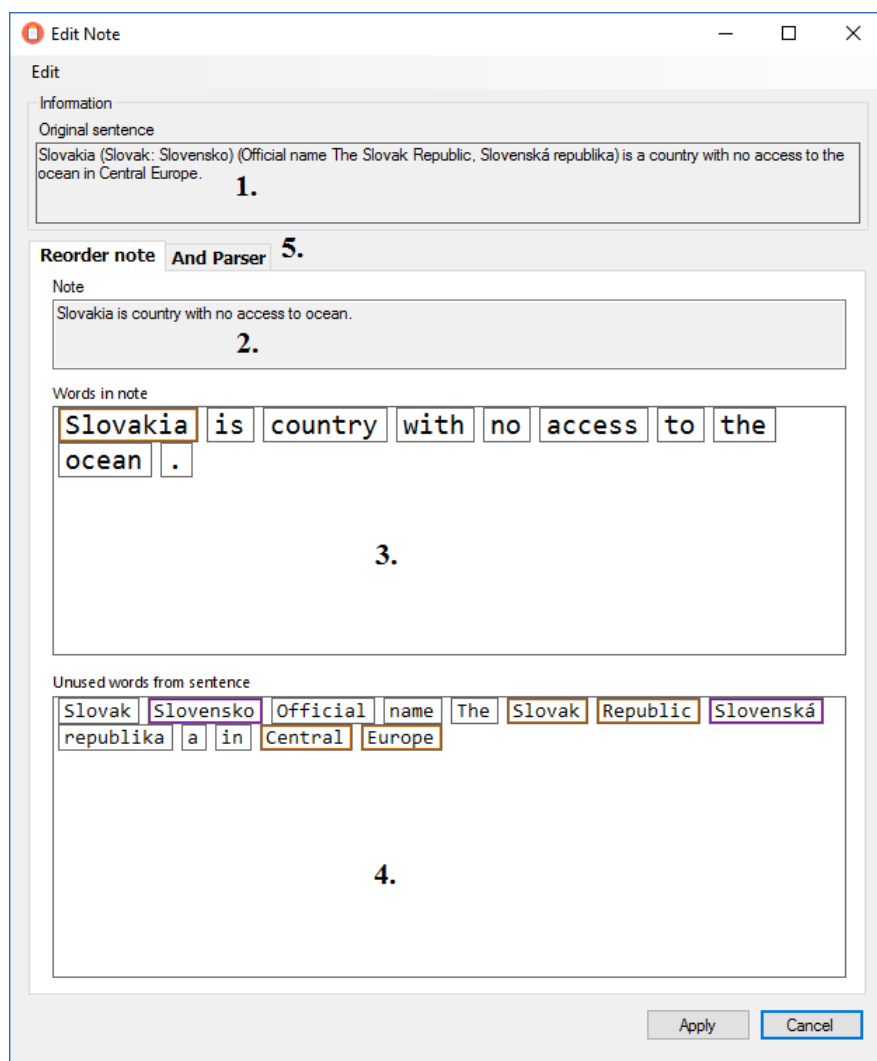
1. výber dotiahnutia textu cez URL článku na wikipédií,
2. výber dotiahnutia text z článku na wikipédií podľa názvu krajiny,
3. zadanie URL alebo názvu krajiny.



Obr. 30: Grafické rozhranie - Okno na dotiahnutie textu

Editovacie okno poznámky (Obr. 31) obsahuje prvky:

1. znenie pôvodnej vety,
2. znenie poznámky,
3. interaktívne upraviteľný tvar poznámky,
4. slová z pôvodnej vety nepoužíte v poznámke,
5. spoznámkovávač viacnásobných poznámok (Obr. 32).



Obr. 31: Grafické rozhranie - Editovacie okno poznámky

Editovacie okno viacnásobných poznámok (Obr. 32) ponúka:

1. znenie pôvodnej vety,
2. viacnásobnú poznámku,
3. interaktívne upraviteľný tvar viacnásobnej poznámky,
4. slová z pôvodnej vety nepoužívané vo viacnásobnej poznámke,
5. špeciálny prvok označujúci množinu viacnásobnej poznámky.

Edit Note

Edit

Information

Original sentence

Other main cities are Košice, Banská Bystrica, Žilina, Trenčín, Nitra, Prešov, and Trnava.

1.

Reorder note

And Parser

Notes

Note
Other cities are Žilina.
Other cities are Trenčín.
Other cities are Nitra.
Other cities are Banská Prešov.
Other cities are Trnava.
Other cities are Banská Prešov.

Words in note

Other cities are [{AND}] 5.

3.

Unused words in sentence

main Banská Bystrica Žilina Trenčín Nitra Prešov and Trnava Košice

4.

Apply

Cancel

Obr. 32: Grafické rozhranie - Editovacie okno viacnásobných poznámok

B Zoznam vzťahov závislostí

V nasledujúcej tabuľke sú zobrazené skratky vzťahov závislostí slov vo vete ako sa používajú v programe, s celým názvom, vysvetlením, príkladom vety a použitím vzhľadom na príkladovú vetu.

Skratka	Názov	Vysvetlenie	Príklad	Použitie
nsubj	Nominal subject	Menná fráza, ktorá je syntaktickým subjektom klauzuly.	Clinton defeated Dole.	nsubj(Clinton, defeated)
nsubjpass	Nominal subject passive	Menná fráza v pasívnom tvare, ktorá je syntaktickým subjektom klauzuly.	Dole was defeated by Clinton.	nsubjpass(Dole, defeated)
dobj	Direct object	Sloveso v mennej fráze označujúce entitu, nad ktorou sa koná akcia.	She gave me a raise.	dobj(gave, raise)
nummod	Numeric modifier	Číselný modifikátor podstatného mena.	Sam spent forty dollars.	nummod(forty, dollars)
nmod	Nominal modifier	Podstatné meno alebo menná fráza slúžiaca ako doplnok.	The Chair's office.	nmod(Chair, office)
amod	Adjectival modifier	Prídavné meno ako modifikátor podstatného mena.	Sam eats red meat.	amod(red, meat)
neg	Negation modifier	Negácia.	Bill is not a scientist	neg(not, scientist)
compound	Compound	Zloženie slov, ktoré spolu majú význam.	I have four thousand sheep.	compound(four, thousand)
aux	Auxiliary	Vedľajšie sloveso klauzuly.	Regan has died.	aux(has, died)
cop	Copula	Vzťah medzi sponovým slovesom (to be) a jeho doplnkom.	Bill is honest.	cop(is, honest)
conj	Conjunct	Spojenie rovnocenných slov.	Bill is big and honest.	conj(big, honest)
cc	Coordinating conjunction	Vzťah medzi spojkou a slovom, patricím k nej.	Bill is big and honest.	cc(big, and)
dep	Unspecified dependency	Nešpecifikovaná závislosť.		
root	Root	Koreň vety. V skutočnosti veta žiadne také slovo neobsahuje.	{ROOT} I love French fries.	root(ROOT, love)

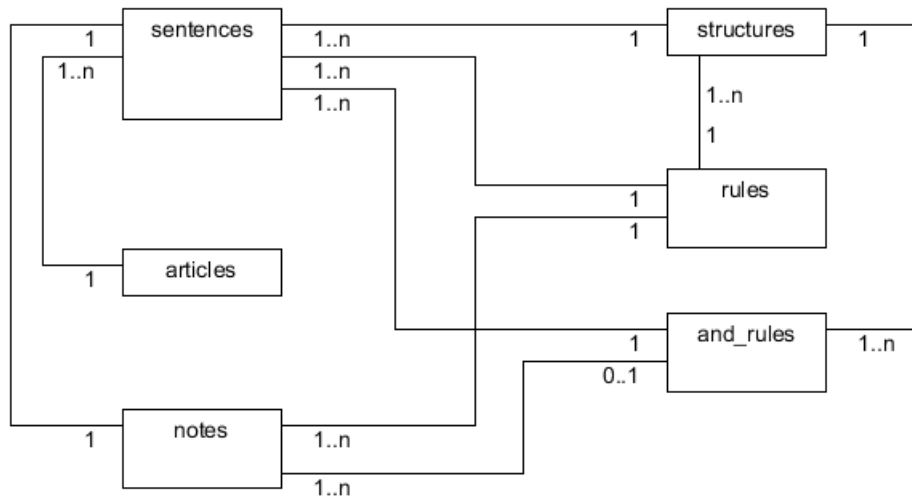
Obr. 33: Zoznam závislostí

C Legenda diagramov kolekcií

V priloženej tabuľke je legenda pre diagramy zobrazujúce štruktúru dát ukladaných v kolekciách v databáze.

Pole	Popis
_id	Generovaná identifikačná hodnota záznamu.
rule_ref_id	Odkaz do kolekcie <i>Rules</i> .
and_rule_ref_id	Odkaz do kolekcie <i>And Rules</i> .
article_ref_id	Odkaz do kolekcie <i>Articles</i> .
structure_ref_id	Odkaz do kolekcie <i>Structures</i> .
note_ref_id	Odkaz do kolekcie <i>Notes</i> .
text	Spracovaný text.
structure_data	Zoznam dát o štruktúre.
relation_name	Názov vzťahu závislosti.
dependencies	Zoznam závislostí.
governor	Nadradený token.
dependent	Podradený token.
position	Pozícia závislosti v štruktúre.
comparison_type	Interná hodnota – typ porovania závislostí.
token_type	Interná hodnota – typ zodpovedajúceho tokenu informácie.
POS	Značka slového druhu.
index	Pozícia tokenu vo vete.
ner	Typ názvoslovnej entity.
lemma	Lema tvar slova.
sentence_terminators	Pozície ukončení viet v poznámke.
set_position	Pozícia množiny viacnásobnej poznámky.

Obr. 34: Legenda diagramov kolekcií



Obr. 35: Databázový model

Pole	Typ
_id	ObjectId
text	String
created_at	DateTime
updated_at	DateTime

Obr. 36: Model kolekcie articles

Pole	Typ
_id	ObjectId
text	String
rule_ref_id	ObjectId
and_rule_ref_id	ObjectId
created_at	DateTime
updated_at	DateTime

Obr. 37: Model kolekcie notes

Pole	Typ
_id	ObjectId
text	String
article_ref_id	ObjectId
structure_ref_id	ObjectId
rule_ref_id	ObjectId
and_rule_ref_id	ObjectId
note_ref_id	ObjectId
created_at	DateTime
updated_at	DateTime

Obr. 38: Model kolekcje sentences

Pole	Typ																																						
_id	ObjectId																																						
structure_data	Array of Documents <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>relation_name</td><td>String</td></tr> <tr> <td>dependencies</td><td>Array of Documents <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>governor</td><td>Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table> </td></tr> <tr> <td>dependent</td><td>Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table> </td></tr> <tr> <td>position</td><td>Integer</td></tr> <tr> <td>comparison_type</td><td>Integer</td></tr> <tr> <td>token_type</td><td>Integer</td></tr> </table> </td></tr> </table>	Pole	Typ	relation_name	String	dependencies	Array of Documents <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>governor</td><td>Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table> </td></tr> <tr> <td>dependent</td><td>Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table> </td></tr> <tr> <td>position</td><td>Integer</td></tr> <tr> <td>comparison_type</td><td>Integer</td></tr> <tr> <td>token_type</td><td>Integer</td></tr> </table>	Pole	Typ	governor	Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table>	Pole	Typ	POS	String	index	Integer	ner	String	lemma	String	dependent	Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table>	Pole	Typ	POS	String	index	Integer	ner	String	lemma	String	position	Integer	comparison_type	Integer	token_type	Integer
Pole	Typ																																						
relation_name	String																																						
dependencies	Array of Documents <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>governor</td><td>Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table> </td></tr> <tr> <td>dependent</td><td>Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table> </td></tr> <tr> <td>position</td><td>Integer</td></tr> <tr> <td>comparison_type</td><td>Integer</td></tr> <tr> <td>token_type</td><td>Integer</td></tr> </table>	Pole	Typ	governor	Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table>	Pole	Typ	POS	String	index	Integer	ner	String	lemma	String	dependent	Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table>	Pole	Typ	POS	String	index	Integer	ner	String	lemma	String	position	Integer	comparison_type	Integer	token_type	Integer						
Pole	Typ																																						
governor	Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table>	Pole	Typ	POS	String	index	Integer	ner	String	lemma	String																												
Pole	Typ																																						
POS	String																																						
index	Integer																																						
ner	String																																						
lemma	String																																						
dependent	Document <table> <tr> <th>Pole</th><th>Typ</th></tr> <tr> <td>POS</td><td>String</td></tr> <tr> <td>index</td><td>Integer</td></tr> <tr> <td>ner</td><td>String</td></tr> <tr> <td>lemma</td><td>String</td></tr> </table>	Pole	Typ	POS	String	index	Integer	ner	String	lemma	String																												
Pole	Typ																																						
POS	String																																						
index	Integer																																						
ner	String																																						
lemma	String																																						
position	Integer																																						
comparison_type	Integer																																						
token_type	Integer																																						
created_at	DateTime																																						
updated_at	DateTime																																						

Obr. 39: Model kolekcje structures

Pole	Typ
_id	ObjectId
sentence_terminators	Array of Integers
structure_ref_id	ObjectId
created_at	DateTime
updated_at	DateTime

Obr. 40: Model kolekcje rules

Pole	Typ
_id	ObjectId
sentence_terminators	Array of Integers
set_position	Integer
structure_ref_id	ObjectId
created_at	DateTime
updated_at	DateTime

Obr. 41: Model kolekcje and rules

D Ukážka článkov použitých v experimente

Slovensko - „Slovakia (Slovak: Slovensko) (Official name The Slovak Republic, Slovenská republika) is a country with no access to the ocean in Central Europe. It is bordered by Austria in the southwest, Hungary in the south, Ukraine in the east, Poland in the north and Czech Republic in the northwest. Its capital city is Bratislava. Other main cities are Košice, Banská Bystrica, Žilina, Trenčín, Nitra, Prešov, and Trnava. Slovakia is a member of the European Union.”

Česko - „Czech Republic (Czech: Česká republika) is a country in Central Europe, sometimes also known as Czechia (Czech: Česko). The capital and the biggest city is Prague. The currency is the Czech Crown (koruna česká - CZK). 1€ is about 27 CZK. The president of the Czech Republic is Miloš Zeman. The Czech Republic's population is about 10.5 million. The local language is Czech language. The Czech language is a Slavic language. It is related to languages like Slovak and Polish. In 1993 the Czech Ministry of Foreign Affairs announced that the name Czechia be used for the country outside of formal official documents. This has not caught on in English usage. Czech Republic has no sea; its neighbour countries are Germany, Austria, Slovakia, and Poland.”

Maďarsko - „Hungary is a country in Central Europe. Its capital city is Budapest. Hungary is slightly bigger than its western neighbour Austria and has about 10 million inhabitants. Other countries that border Hungary are Slovakia, Ukraine, Romania, Serbia, Croatia and Slovenia. Hungary's official language is the Hungarian language. It has been a member of the European Union (EU) since 2004. In Hungarian the country is called Magyarország (Hungary) or Magyar Köztársaság (Hungarian Republic). This is named after the Magyar tribes who came to Hungary in the late 9th century.”

Pol'sko - „Poland is a country in Eastern Europe. It is next to Germany to the west (along Oder and Lusatian Neisse), the Czech Republic and Slovakia to the south, Ukraine and Belarus to the east, and the Baltic Sea, Lithuania, and Russia to the north. The total land area of Poland is about 312,679 km² (120,728 mi²). This

makes Poland the 77th largest country in the world with over 38.5 million people. Most Polish people live in large cities, including the capital, Warsaw (Polish: Warszawa), Łódź, Cracow (Polish: Kraków), the second capital of Poland (first was Gniezno), Szczecin, Gdańsk, Wrocław and Poznań. The word "Poland" was written officially for the first time in 966. In 1569, Poland formed a strong union with Lithuania called the Polish-Lithuanian Commonwealth. At some point in its history, it was the largest state in Europe and became very influential. Much of the territory that now makes up Central European states used to belong to the Commonwealth. Eventually, after entering a somewhat sudden yet steady decline, the Commonwealth collapsed in 1795. Poland regained its independence in 1918 after World War I. In 1921, Poland defeated Soviet Russia in the Polish-Soviet War that started in 1919."

Nemecko - „The Federal Republic of Germany, also called Germany (German: Bundesrepublik Deutschland or just Deutschland), is a country in Central Europe. The country's full name is sometimes shortened to the FRG (or the BRD, in German). To the north of Germany are the North Sea, the Baltic Sea, and the country of Denmark. To the east of Germany are the countries of Poland and the Czech Republic. To the south of Germany are the countries of Austria and Switzerland. To the west of Germany are the countries of France, Luxembourg, Belgium, and the Netherlands. The total area of Germany is 137,847 square miles. The large majority of Germany has warm summers and cool or cold winters. In June 2013, Germany had a population of 80.6 million people. After the United States, Germany is the second most popular country for migration in the world. Before it was called Germany, it was called Germania. In the years A.D. 900 until 1806, Germany was part of the Holy Roman Empire. From 1949 to 1990, Germany was made up of two countries called the Federal Republic of Germany (inf. West Germany) and the German Democratic Republic (inf. East Germany). During this time, the capital city of Berlin was divided into a west and an east part. On 13 August 1961, East Germany started building the Berlin Wall between the two parts of Berlin. West Germany was one of the countries that started the European Union."

Taliansko - „Italy is a country in Europe and a member of the European Union.

Its official name is Repubblica Italiana. Italy is a democratic republic and is a founding member of the European Union. Italy is also a member of the G8, as it has the 8th largest Gross Domestic Product in the world. Its President is Sergio Mattarella and its Prime Minister is Matteo Renzi. Before 1861, it was made up of smaller kingdoms and city-states.”

Francúzsko - „France (French: France), officially the French Republic (French: République française), is a country in Western Europe. Its capital city is Paris. It is a member of the European Union. It is known for its culture, its many monuments and structures, and places such as the Louvre, the Eiffel Tower, the Arc de Triomphe, Giverny, Mont Saint Michel, Versailles, and Notre Dame de Paris. France is divided into 22 régions that are further subdivided départements. The country has been one of the great powers since the end of the 17th century. In the 18th and 19th centuries, it built a big colonial empire across West Africa and Southeast Asia. Nowadays, this does not exist. It is the most visited country in the world. About 82 million foreign tourists visit it every year. France is a founding member of the European Union. It has the largest land area of any member. France is also a founding member of the United Nations, and a member of the G8 and NATO. It is one of the five permanent members of the United Nations Security Council. It has nuclear weapons, including active warheads, and also has nuclear power plants. Some well-known cities in France include Paris, Lyon, Marseille, Bordeaux, Lille, Toulouse, Nice, Strasbourg, Rennes and Nantes.”

Spojené kráľovstvo - „The United Kingdom of Great Britain and Northern Ireland, called the United Kingdom, GB or UK, is a sovereign state in Western Europe. It unites England, Northern Ireland, Scotland and Wales as one Kingdom.[10] It is a member of the European Union, the United Nations, the Commonwealth, NATO and the G8. It has the sixth largest economy in the world. Around 65 million people live in the UK. Most people in the UK speak English. There are five native languages other than English. They are Welsh in Wales, Gaelic and Scots in Scotland and Northern Ireland, Irish in Northern Ireland, and Cornish in Cornwall. Between the 17th and mid 20th-centuries Britain was an important world power. It became a colonial empire that controlled large areas of Africa, Asia, North America and

Oceania. Today this empire does not exist, although Britain keeps links with most countries of its former empire. Some well-known cities in the UK are London, Edinburgh, Cardiff, Belfast, Manchester, Bristol, Liverpool, Birmingham, York and Glasgow.”

Španielsko - „Spain is a country in Southern Europe. It is in the Iberian Peninsula near Portugal and Gibraltar. France and the country of Andorra are on its northeast side, where the Pyrenees mountains are. The people of Spain are called Spaniards. Most people there speak Spanish (in Spanish, ”Castellano”, from Castilla, or Español”) but there are other languages in different parts of the country. They are Catalan, Basque, and Galician, Leonese, Aragonese, Aranese Occitan and even Portuguese. The religion of most of the people in Spain is Roman Catholic. Since 1975, Spain has had a king who only does what the constitution allows him to. For example, the king can declare a war, but only if the Government asks him to do so. The parliament is called Las Cortes Generales, and has two bodies: Ël Congreso”(The Congress) and Ël Senado”(The Senate) and it is chosen by the Spanish people by voting. This kind of government is called a constitutional monarchy. The King of Spain is Felipe VI. The Prime minister is Mariano Rajoy. The government and the king’s palace are in Madrid, the capital of Spain. Spain has more than five hundred thousand square kilometres of land. It is smaller than France, but it is bigger than Sweden or Germany. Almost fifty million people live in Spain. Spain has 17 parts called autonomous communities (this means that they can decide upon some affairs themselves). Each part has its own government.”

Ukrajina - „Ukraine is a country in Eastern Europe. Russia is to the north-east of Ukraine, Belarus is to the Northwest, Poland and Slovakia are to the West, Hungary, Romania, Moldova and self-proclaimed Transnistria are to the South West and the Black Sea is to the Southwest. Ukraine is a republic. The capital of Ukraine is Kiev. It was a part of the Soviet Union from 1922 until 1991.”

Rakúsko - „Austria (German: Österreich; officially called Republic of Austria), is a country in Central Europe. Around Austria there are Germany, Czech Republic, Slovakia, Hungary, Slovenia, Italy, Switzerland, and Liechtenstein. Currently, the

chancellor is Werner Faymann. Austria has been a member-state of the EU since 1995. The people in Austria speak German, a few also Hungarian, Slovenian and Croatian. The capital of Austria is Vienna (Wien). Austria is more than a thousand years old. Its history can be followed to the ninth century. At that time the first people moved to the land now known as Austria. The name Östarrichiis first written in an official document from 996. Since then this word has developed into the Modern German word Österreich, which literally means East Kingdom. Austria is a democratic state and has nine federal states (German: 'Bundesländer'): Vorarlberg, the Tyrol, Salzburg, Carinthia, Styria, Upper Austria, Lower Austria, Vienna and Burgenland. It is a neutral state, that means it does not take part in wars with other countries. Austria has been in the United Nations since 1955 and in the European Union since 1995.”

Švédsko - „Sweden (Swedish: Sverige) is a Nordic country in the part of Europe called Scandinavia. Its neighbors are Finland and Norway. Sweden is also connected to Denmark in the south by a bridge. It is a developed country. It is famous for its welfare state. People who live in Sweden are called Swedes. Sweden's capital city is Stockholm. Sweden is a constitutional monarchy because it has a king, Carl XVI Gustaf, but he does not have any real power. Sweden is a parliamentary state meaning that the government is elected by the parliament which is appointed by the people. The country is democratically ruled by a government headed by an elected prime minister. Stefan Löfven was elected Prime Minister in September 2014. He took office in October 2014. The population of Sweden is almost 9.5 million people. Sweden has an official majority language, (called svenska in Swedish). Sweden has five official minority languages, Finnish, Yiddish, Sami, Meänkieli and Romani. Sweden became a member of the European Union in 1995. It is not a member of the Eurozone. Sweden has not begun to use the euro as currency. This is because the people have voted against this. The currency remains the Swedish krona (Swedish crown). Sweden has 25 provinces (landskap). They are found in three different regions: Norrland in the North, Svealand, the central region, and Götaland in the South.”

E IIT.SRC článok

Interactive System for Creation of Notes

Martin NEMČEK*

*Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
xnemcekm@stuba.sk*

Abstract. We are overwhelmed by information from various topics. The challenge in education is to create notes which covers important subset of information. There are known methods to extract information from text. In this article we propose a system to extract the notes from text which are important for educational purpose, so it should create personalized notes for students. We use mainly syntactic text analysis. Notes are created by help of part-of-speech tags and dependencies between words in sentences. The outcome will be an interactive system for creating notes based on learned rules from user.

1 Introduction

Computers are not able to understand information in natural language. In our proposed system the notes are created from sentences by extracting relevant information from them. We use syntactic analysis of sentences and extract relations and dependencies between words from these sentences. The final result of our proposed method are personalized notes. The user will be able to modify the automatically created notes. The system will then learn new rules for sentence to note transformation from these changes and takes them into account for the next time.

2 Our proposed system

A rule consists mainly from two parts - *list of data of original sentence* and *list of data of note*. Each entry in *list of data of original sentence* and *list of data of note* contains these parts: relation name and list of grouped dependencies with the same relation name. Each dependency contains a governor token, a dependent token and its position considering all dependencies. The governor and dependent token consists of Part-Of-Speech (POS) tag and index of word in sentence to which is the token connected. Index of the token is bounded with a position of its word in sentence.

Dependencies from the second list are applied to sentence to create a new note. The rule may order to create a compound note from a sentence. The compound note is composited of some simple sentences. The positions in sentence on which the note should be split into smaller sentences are

* Bachelor study programme in field: Informatics

Supervisor: Miroslav Blšták, Institute of Informatics and Software Engineering, Faculty of Informatics and Information Technologies STU in Bratislava

kept within the rule.

When processing a sentence an applicable rule has to be looked up in database before creation of the note. Dependencies of rule and dependencies of the sentence being processed has to correspond to each other. Evaluation is based upon two conditions. The sentence that is being processed has to have the exact amount of entries in list of data of original sentence while these entries contain exactly the same relation names as the rule's relation names.

The applicable rule is found if these two conditions are met. However, the conditions can cause a situation that more than one rule is found. In this case we have to calculate the match probability of this sentence and the original sentence obtained from the rule. The rule with the highest probability of the match is applied.

Calculating the match consists of several steps. First, the POS tags match of governor and dependent tokens is calculated separately. Indices of governor and dependent tokens are calculated also separately. These first steps determine if the sentence contains arbitrary dependency with same value of POS tag or index. In followed step is determined a half-match of dependencies. Half-match of dependency is match of POS tag and index at the same time at governor or dependent token of dependency. We calculate matches of POS tags and index of governor or dependent token for every dependency. Finally, in the last step we calculate the number of absolute-matched dependencies. Absolute-match dependencies is the total match of POS tags and indices in governor and dependent tokens. Every step has assigned a rating. If a condition in the step is evaluated as true, the rating of the step is added to the final result. The final result is a percentage value of the match. The rating is based on importance of the step in calculating a precise match, while depending on the number of steps and dependencies, so the final result cannot exceed a limit of 100%. A pseudo code for an algorithm calculating the match is outlined in Algorithm 1 and specific example is shown on Figure 1.

Algorithm 1 Calculating match

```

1: procedure CALCULATEMATCH(sentence, originalDependencies)
2:   oneCompareTypeRating  $\leftarrow$  calculate percentage rating of one comparison
3:   for all originalDependencies do
4:     if count(sentence, dependency) = count(originalDependencies, dependency) then
5:       match  $\leftarrow$  match + oneCompareTypeRating
6:       counter  $\leftarrow$  counter + count(originalDependencies, dependency)
7:   oneCompareTypeRating  $\leftarrow$  oneCompareTypeRating / counter
8:   for all originalDependency do
9:     for all dependency do
10:      for all comparison do
11:        if applyComparison(sentence, comparison, dependency) then
12:          match  $\leftarrow$  match + oneCompareTypeRating
13:   return match

```

If rule look up does not find any applicable rule, it means that the system have not processed the same or similar sentence yet. A manual rules of parser are used in this case. The output of the parser is a note. A new rule is created based on the note. Dependencies of original sentences are taken and used to create a *list of data of original sentence*. This list is then assigned to the rule. Dependencies of note are used to create a *list of data of note* which is then also assigned to the rule. The sentence ends are determined depending on how many sentences the note contains. POS tags and indices of tokens are stated by the corresponding words of the original sentence and the newly created note.

By the principle of rule look up, the sentence being processed has to contain dependencies from the *list of data of original sentence* and also dependencies from the *list of data of note*.

The process of applying a rule has several steps. For each dependency in the list of data of original sentence, the respective dependency is looked up in sentence that is being processed. The word corresponding with dependent token from the looked up dependency is taken and added to the note on its index position. In case of dependency relation *nominal subject* the word corresponding with governor is also added. After processing all dependencies the last minor changes are done such as capitalization of the first letter of the note, splitting note into more sentences if rule defined so. Algorithm 2 shows pseudo code of the process of applying rule on sentence.

Algorithm 2 Applying rule

```

1: procedure APPLYRULE(sentence, rule)
2:   note ← new Note
3:   for all ruleDependencies do
4:     dependency ← findDependency(sentence, ruleDependency)
5:     if isFound(dependency) then
6:       add(note, getDependent(dependency))
7:       if isNominalSubject(relation(dependency)) then
8:         add(note, getGovernor(dependency))
9:   splitToSentences(note, sentencesEnds(rule))
   return note
  
```

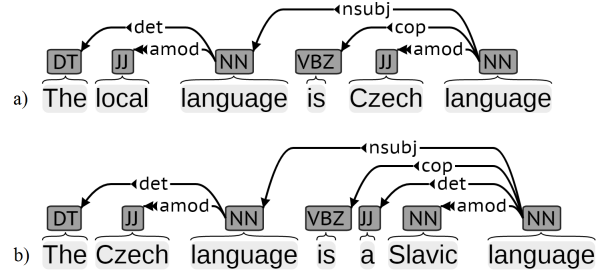


Figure 1. Example sentences

Lets consider example from Figure 1. We have rules for two sentences and we are processing the first one. In this situation, there are at least two rules which are applicable for the sentence *a*. Assume that we are calculating match with the sentence *b*. We iterate over all dependencies of processed sentence *a*. The first dependency is *det* with the governor token NN (noun) and index 3 and the dependent token DT (determiner) and index 1. First, we find out, if the sentence *b* contains any dependency with tokens NN or DT and index equals to 1 or 3. This is the separate calculation of POS tags and indices. Then, we try to find in the sentence *b* any dependency, which has dependent or governor token tag of type NN and index equal to 3 or tag of type DT and index equal to 1. This is only the half-match step. As the last step, we check if sentence *b* contains dependency, where the governor token is NN and index is equal to 3 and the dependent token is DT and its index is 1. If any of these step were matched, the rating of that particular step is added to the final result and iteration continues with following dependency until all dependencies were iterated over.

Acknowledgement: This work was partially supported by the Scientific Grant Agency of Slovak Republic, grant No. VG 1/0646/15.