Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

FIIT-5220-64334

Bc. Jozef Gajdoš

Identifikácia parafrázovania v textových dokumentoch

Diplomová práca

Študijný program: Softvérové inžinierstvo
Študijný odbor: 9.2.5 Softvérové inžinierstvo
Miesto vypracovania: Ústav informatiky a softvérového inžinierstva, FIIT STU Bratislava
Vedúci práce: Ing. Tomáš Kučečka

máj 2014

S T U
F I I T
SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
Fakulta informatiky a informačných technológií

Zadanie diplomovej práce

Meno študenta: **Bc. Jozef Gajdoš**

Študijný program: Softvérové inžinierstvo

Študijný odbor: Softvérové inžinierstvo

Názov práce: **Identifikácia parafrázovania v textových dokumentoch**

Samostatnou výskumnou a vývojovou činnosťou v rámci predmetov Diplomový projekt I, II, III vypracujte diplomovú prácu na tému, vyjadrenú vyššie uvedeným názvom tak, aby ste dosiahli tieto ciele:

Všeobecný cieľ:

Vypracovaním diplomovej práce preukážte, ako ste si osvojili metódy a postupy riešenia relatívne rozsiahlych projektov, schopnosť samostatne a tvorivo riešiť zložité úlohy aj výskumného charakteru v súlade so súčasnými metódami a postupmi študovaného odboru využívanými v príslušnej oblasti a schopnosť samostatne, tvorivo a kriticky pristupovať k analýze možných riešení a k tvorbe modelov.

Špecifický cieľ:

Vytvorte riešenie, zodpovedúce návrhu textu zadania, ktorý je prílohou tohto zadania. Návrh bližšie opisuje tému vyjadrenú názvom. Tento opis je záväzný, má však rámcový charakter, aby vznikol dostatočný priestor pre Vašu tvorivosť.

Riaďte sa pokynmi Vášho vedúceho.

Pokiaľ v priebehu riešenia, opierajúc sa o hlbšie poznanie súčasného stavu v príslušnej oblasti alebo o priebežné výsledky Vášho riešenia alebo o iné závažné skutočnosti, dospejete spoločne s Vaším vedúcim k presvedčeniu, že niečo v texte zadania a/alebo v názve by sa malo zmeniť, navrhnite zmenu. Zmena je spravidla možná len pri dosiahnutí kontrolného bodu.

Miesto vypracovania: Ústav informatiky a softvérového inžinierstva FIIT STU v Bratislave

Vedúci práce: **Ing. Tomáš Kučečka**

Termíny odovzdania:

podľa harmonogramu štúdia platného pre semester, v ktorom máte príslušný predmet (Diplomový projekt I, II, III) absolvovať podľa Vášho študijného plánu

Predmety odovzdania:

V každom predmete dokument podľa pokynov na www.fiit.stuba.sk v časti:
home > Informácie o > štúdiu > organizácia štúdia > diplomový projekt

V Bratislave dňa 18. 2. 2013

prof. Ing. Pavol Návrat, PhD.
riaditeľ Ústavu informatiky a softvérového inžinierstva

STU FIIT

SLOVENSKÁ TECHNICKÁ
UNIVERZITA V BRATISLAVE
FAKULTA INFORMATIKY
A INFORMAČNÝCH TECHNOLÓGIÍ

Návrh zadania diplomovej práce

Finálna verzia do diplomovej práce [1]

Študent:

Meno, priezvisko, tituly:	Jozef Gajdoš, Bc.
Študijný program:	Softvérové inžinierstvo
Kontakt:	xgajdos@is.stuba.sk

Výskumník:

Meno, priezvisko, tituly:	Tomáš Kučečka, Ing.

Projekt:

Názov:	Identifikácia parafrázovania v textových dokumentoch
Názov v angličtine:	Identification of paraphrasing in text documents
Miesto vypracovania:	Ústav informatiky a softvérového inžinierstva, FIIT STU, Bratislava
Oblasť problematiky:	Analýza textu, Podobnosť textov

Text návrhu zadania[2]

V dnešnej dobe je plagiátorstvo rozšírený spoločenský problém, a to najmä v akademickom prostredí, preto sa v poslednej dobe čoraz viac kladie dôraz na jeho prevenciu a odhaľovanie. Problémov pri identifikovaní plagiátorstva v textových dokumentov je hneď niekoľko. K tým hlavným patrí fakt, že nepoznáme originálny text, z ktorého bol plagiát vytvorený a detekcia parafrázovania. Parafrázovanie je samo o sebe ťažko identifikovateľné, keďže ide o vyjadrenie rovnakej myšlienky, len inými slovami, a teda závisí od sémantiky. Parafrázovanie textu môže byť na rôznych úrovniach, pričom čím vyššia úroveň, tým je jeho odhalenie obtiažnejšie.

K existujúcim prístupom k identifikácii parafrázovania na nižšej úrovni patrí napríklad metóda n-gramy, ktorá je jednou z najčastejšie používaných metód pri identifikovaní plagiátorstva. Ďalšie známe používané prístupy sú založené na metóde najdlhších spoločných podsekvencií, analýze príponových stromov a podobne.

Analyzujte existujúce metódy odhaľovania podobnosti medzi textovými dokumentmi napísanými v slovenskom jazyku pre potreby identifikácie parafrázovania na vyššej úrovni. Navrhnite a implementujte vlastnú metódu pre identifikovanie parafrázovania na vyššej úrovni alebo rozšírte existujúci prístup. Experimentuje s navrhnutou metódou nad korpusom dokumentov napísaných v slovenskom jazyku a dosiahnuté výsledky overte porovnaním s inými existujúcimi systémami na odhaľovanie plagiátorstva v slovenskom jazyku.

---

[1] Vytlačiť obojstranne na jeden list papiera

[2] 150-200 slov (1200-1700 znakov), ktoré opisujú výskumný problém v kontexte súčasného stavu vrátane motivácie a smerov riešenia

Literatúra[3]

- Lintean, M. C., Rus, V. (2010):Paraphrase Identification Using Weighted Dependencies an Word Semantics. Informatica (Slovenia), vol. 34, Slovenian Society Informatika, no. 1 pp.19–28, 2010.
- Kozareva, Z.,Montoyo, A. (2006):Paraphrase Identification on the basis of Supervise Machine Learning Techniques. In:Proceedings of the 5th International Conference o Natural Language Processing (FinTAL 2006), WSEAS, pp. 524-533, 2006.

Vyššie je uvedený návrh diplomového projektu, ktorý vypracoval(a) Bc. Jozef Gajd konzultoval(a) a osvojil(a) si ho Ing. Tomáš Kučečka a súhlasí, že bude takýto projekt vies prípade, že bude pridelený tomuto študentovi.

V Bratislave dňa 12.1.2013

Podpis študenta

Podpis výskumníka

Vyjadrenie garanta predmetov Diplomový projekt I, II, III

Návrh zadania schválený: áno / ~~nie~~[4]

Dňa: 1.2.2013

Podpis garanta predmetov

[3] 2 vedecké zdroje, každý v samostatnej rubrike a s údajmi zodpovedajúcimi bibliografickým odkazom pc normy STN ISO 690, ktoré sa viažu k téme zadania a preukazujú výskumnú povahu problému a j aktuálnosť (uveďte všetky potrebné údaje na identifikáciu zdroja, pričom uprednostnite vedecké príspevl časopisoch a medzinárodných konferenciách)

ANOTÁCIA

Slovenská technická univerzita v Bratislave
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ
Študijný program: Softvérové inžinierstvo

Autor: Bc. Jozef Gajdoš
Diplomový projekt: Identifikácia parafrázovania v textových dokumentoch
Vedenie diplomového projektu: Ing. Tomáš Kučečka
máj 2014

Cieľom tejto diplomovej práce je oboznámiť čitateľa s metódami identifikácie parafrázovania a predspracovania textov v prirodzenom jazyku, navrhnúť a implementovať metódu použiteľnú na identifikáciu parafrázovania v slovenskom jazyku. Práca ponúka prehľad možností predspracovania textov napísaných v slovenčine, ako aj metód identifikácie parafrázovania v anglickom jazyku, pretože z týchto metód identifikácie parafrázovania vychádzame.

Hlavným prínosom našej práce je návrh a implementácia metódy identifikácie parafrázovania pre slovenský jazyk, využívajúcej SVM (algoritmy podporných vektorov), spolu s vhodnými metódami predspracovania. Ďalším prínosom našej práce je návrh a implementácia knižnice pre morfologické značkovanie viet v slovenskom jazyku.

Náš implementovaný systém dosahuje úspešnosť až 93% pri identifikácii parafrázovania, ktorú sme dosiahli pomocou metódy predspracovania nazvanej *Kombinovaná metóda*, a metódou konverzie nazvanou *Jednoduchá reprezentácia*. V tejto práci sa taktiež venujeme aj iným metódam predspracovania a informačnému prekryvu.

ANOTATION

Slovak University of Technology Bratislava
FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES
Degree Curse: SOFTWARE ENGINEERING

Author: Bc. Jozef Gajdoš
Diploma project: Identification of paraphrasing in text documents
Supervisor: Ing. Tomáš Kučečka
2014, May

The goal of this thesis is to present to the reader various methods that deal with paraphrasing identification and preprocessing of natural text. Next goal is to design and implement own method for paraphrasing identification in Slovak language. This work gives an overview of existing methods for preprocessing in Slovak language and also discusses several approaches that identify paraphrasing in English language.
The main result of our work is a created method for identifying paraphrasing in the Slovak language using SVM (support vector machines) together with appropriate preprocessing methods. Another important output of this work is created library for morphological tagging of sentences written in Slovak language.
Our implemented system achieves accuracy 93% in identification of paraphrasing, this was achieved by using a pre-processing method called *Combined method* and conversion method called *Simple representation.* In this work we also deal with other methods of text pre-processing and information overlap.

Čestné prehlásenie

Prehlasujem, že diplomovú prácu som vypracoval samostatne s využitím iba uvedených informačných prameňov.

V Bratislave 10.5.2014

Bc. Jozef Gajdoš

Poďakovanie

Ďakujem vedúcemu diplomového projektu Ing. Tomášovi Kučečkovi za jeho pomoc, odborné vedenie, cenné rady a pripomienky pri vypracovávaní mojej diplomovej práce. Takisto ďakujem aj ľudom, ktorí mi pomohli vytvoriť a zozbierať parafrázované vety.

Bc. Jozef Gajdoš

Obsah

Zoznam skratiek a slovníkových pojmov

Zoznam použitých skratiek:

Skratka	Vysvetlenie
ATN	rozšírené prechodové siete
r-mark	Ručne morfologicky anotovaný korpus
idf	*Inverse document frequency* – prevrátená početnosť slova vo všetkých dokumentoch
tf	*Term Frequency*- početnosť slova v dokumente
SAV	Slovenská akadémia vied
POCO	Objekt .NET frameworku – *Plain Old CLR Object*

Vysvetlenie použitých pojmov:

Pojem	Vysvetlenie
afix	predpona alebo prípona
aktant	jednotka významovej roviny vety, pomenováva účastníka slovesného deja
flektívny, flexívny	lingvisticky ohybný
particient	prvok významovej štruktúry, ktorý vetne odráža zobrazované situácie
prefix	predpona
sufix	prípona
lexikálna jednotka	komplexná, ďalej analyzovatelná jazyková mikroštuktúra s množstvom jazykových aj mimojazykových vzťahov

Zoznam obrázkov

Zoznam tabuliek

Úvod

V dnešnej dobre sa obrovské množstvo znalostí a prác uchováva v elektronickej podobe a je napísaných v prirodzenom jazyku. Práve oblasť analýzy textu sa zaoberá spracovaním a analýzou týchto informácii v textovej podobe. Medzi otvorené problémy analýzy textu patrí detekcia parafrázovania na vyššej úrovni. Z hľadiska problematiky spracovania textu ide teda o problém odhaľovania vzorov v texte a určovaniu podobnosti medzi textami porovnávaním týchto vzorov.

Problém identifikácie parafrázovania je potrebné riešiť, pretože analýza textu je oblasť, v ktorej neustále prebieha masívny výskum. Nejestvuje uspokojivé riešenie identifikácie parafrázovania na vyššej úrovni. Takisto nejestvuje ani uspokojivé riešenie pre identifikáciu parafrázovania v slovenskom jazyku.

Identifikovanie parafrázovania môže pomôcť pri:

- detekcii plagiátorstva v prirodzenom jazyku,
- zhlukovaní dokumentov,
- dolovaní dát a vyhľadávaní informácií,
- sumarizácii poznatkov a hľadaní vzorov v prirodzenom jazyku,
- hľadaní článkov s rovnakým obsahom.

Zaujímavým použitím identifikácie parafrázovania je inteligentný systém, ktorý kladie otázky a používateľ na ne odpovedá v prirodzenom jazyku. Ak je odpoveď parafrázou vzorovej odpovede, systém ju považuje za správnu.

Odhaľovanie plagiátorstva v prirodzenom jazyku je často zložité a v niektorých prípadoch až nemožné. Veľkým problémom je napríklad dostupnosť samotného zdroja, z ktorého je plagiát vytvorený, a samotný problém detekcie parafrázovania na vyššej úrovni. V určitých prípadoch je ťažké aj pre samotného človeka posúdiť, či daný dokument vznikol alebo nevznikol parafrázovaním iného diela.

Jedným z foriem maskovania plagiátorstva je práve parafrázovanie na vyššej úrovni, v slovenskom jazyku aj jednoduché parafrázovanie dokáže oklamať nástroje na detekciu plagiátorstva. Preto je dôležitá problematika identifikácia parafrázovania.

Dokument je rozdelený na niekoľko kapitol.

Kapitola 1 - Parafrázovanie – venuje sa definíciám parafrázovania.

Kapitola 2 - Štruktúra slovenského jazyka – kapitola sa venuje našej problémovej oblasti – slovenskému jazyku. Uvádza jeho základné gramatické a syntaktické kategórie. Tieto poznatky sú dôležité z hľadiska procesu identifikácie parafrázovania v slovenčine.

Kapitola 3 - Existujúce prístupy predspracovania textov v slovenskom jazyku – venuje sa nástrojom a súčasným možnostiam predspracovania textov v slovenskom jazyku pre účely identifikácie parafrázovania. Zameriavame sa najmä na morfologickú analýzu viet.

Kapitola 4 - Metódy identifikácie parafrázovania – v tejto kapitole sa venujeme rôznym metódam identifikácie parafrázovania. Zameriavame sa najmä na metódy identifikácie parafrázovania využívajúce algoritmy strojového učenia. V závere kapitoly porovnávame a hodnotíme prínosy jednotlivých prístupov.

Kapitola 5 - Špecifikácia požiadaviek – v tejto kapitole sú predstavené požiadavky na aplikáciu, ktorá implementuje metódu identifikácie parafrázovania pre slovenský jazyk.

Kapitola 6 - Návrh identifikácie parafrázovania na vyššej úrovni – predstavuje náš návrh metódy identifikácie parafrázovania textových dokumentov napísaných v slovenskom jazyku.

Kapitola 7 - Implementácia – venuje sa implementácii prototypu, knižniciam a rámcom pri tom využitých. Opisuje detaily metód pre morfologické značkovanie a identifikáciu parafrázovania.

Kapitola 8 - Overenie riešenia – venuje sa testovacím dátam, spôsobu validácie a vyhodnotenia metód pre morfologické značkovanie a identifikáciu parafrázovania.

Kapitola 9 - Zhrnutie – ponúka celkový pohľad na doteraz dosiahnuté výsledky identifikácie parafrázovania.

Bibliografia – zoznam použitej literatúry.

Prílohy – obsahuje prílohy k práci, napríklad technickú dokumentáciu s diagramami k implementácii systému na identifikáciu parafrázovania.

1 Parafrázovanie

„Parafráza je upravená alebo voľne interpretovaná myšlienka autora zdrojovej myšlienky, pričom samotná podstata myšlienky sa v procese parafrázovania nemení.“[Piaček, a iní, 1999]

Iná definícia parafrázovania je :

„Parafrázu ľubovoľného segmentu jazyka *L* (slova, frázy, vety, súvetia, textu) definujeme ako každý taký segment jazyka *L'*, že *L* a *L'* sú sémanticky relevantné. Sémanticky relevantné môžu byť v určitom kontexte ľubovoľné dva segmenty jazyka, ktorých významy majú neprázdny prienik.“ [Páleš, 1994]

Parafrázovanie v slovenčine môže vzniknúť napríklad:

- zmenou poradia slov,
- zmenou poradia viet v odstavci,
- zmenou slovesa na spodstatnené meno,
- vkladaním slov do viet,
- použitím synoným,
- zovšeobecnením vety,
- konkretizáciou vety.

Slovenčina je bohatý jazyk, ktorý umožňuje vyjadrenie tej istej myšlienky viacerými spôsobmi. Spôsobov, ako parafrázovať text v slovenskom jazyku, je teda veľmi veľa, pričom tieto jednotlivé prístupy, ako aj ich hromadné použitie definujú úroveň parafrázovania, ktorá bola použitá. Platí, že čím vyššia úroveň, tým je jeho odhalenie náročnejšie.

Parafrázovanie na vyššej úrovni

Za parafrázovanie na vyššej úrovni považujeme úpravu vety, ktorá môže vzniknúť:

- zmenou poradia slovosledu,
- vkladaním alebo odstraňovaním slov z vety,
- použitím synoným.

2 Štruktúra slovenského jazyka

Slovenčina patrí do skupiny indoeurópskych jazykov a konkrétne do západoslovanských jazykov spolu s češtinou, poľštinou a lotyšskou srbčinou. Zaraďuje sa k flektívnym jazykom. Pre jej základné porozumenie a komunikáciu je potrebné zvládnuť asi tisíc až tritisíc slov. Krátky slovník slovenského jazyka má približne 60 tisíc slov tvoriacich jadro slovnej zásoby slovenského jazyka, zatiaľ čo napríklad v angličtine základná slovná zásoba obsahuje okolo 2 500 slov. Oproti angličtine je slovenčina omnoho rozsiahlejší jazyk.

2.1.1 Vzťahy medzi významami slov

V slovenskom jazyku môžeme vidieť tieto vzťahy medzi významami slov:

Synonymá (rovnoznačné slová) sú slová, ktoré majú rôznu formu, ale rovnaký alebo podobný význam, patria do toho istého slovného druhu. Napríklad: utekať – bežať, nahnevaný – nazlostený – rozčúlený. Za synonymá sa nepovažujú slangové slová, nespisovné a nárečové slová. Ďalej sa delia podľa štýlového zafarbenia na:

- neutrálne (matka, mama, mať),
- hovorové (mati, mamina, mamuľka),
- básnické (mamuška, matička),
- odborné (matka dieťaťa, rodička).

Homonymá (rovnozvučné slová) – slová, ktoré rovnako znejú, ale majú rozličné významy. Ich význam závisí od kontextu, v ktorom sú použité. Napríklad: pravý – skutočný, nachádzajúci sa vpravo.

Antonymá (opozitá, slová s opačným významom) – dvojice slov, ktoré majú protikladný význam, vyjadrujú vzájomný opak činnosti, stavu alebo vlastnosti. Napríklad: hovoriť – mlčať.

2.1.2 Tvaroslovie a gramatické kategórie

Tvaroslovie (morfológia) je označenie pre opis tvarových prostriedkov jazykovej skladby, v rámci gramatiky sa zaoberá slovnými druhmi, ich ohýbaním a odvodzovaním slov pomocou afixov.

Slovné druhy delíme podľa plnovýznamovosti na:

- plnovýznamové,
- neplnovýznamové (považujeme ich za stop slová).

Podľa ohybnosti ich delíme na:

- ohybné (substantíva, adjektíva, pronomiá, numeráliá, verbá),
- neohybné (adverbiá, prepozície, konjukcie, partikuly, interjekcie).

Ohybné slovné druhy a adverbiá majú vetnočlenskú platnosť. Ostatné neohybné slovné druhy nemajú vetnočlenskú platnosť.

Substantívum (podstatné meno) je plnovýznamový, ohybný slovný druh, označuje samostatne existujúce alebo chápané veci a javy. Podstatné meno je hlavným slovným druhom, ktorý vo vete najčastejšie vyjadruje podmet a predmet. Medzi jeho gramatické kategórie patria:

- rod – mužský, ženský, stredný,
- číslo – singulár (jednotné číslo), plurál (množné číslo),
- pád – nominatív, genitív, datív, akuzatív, lokál, inštrumentál.

Medzi ďalšie delenia patria:

- vzor,
    - chlap, hrdina, dub, stroj (pre mužský rod),
    - žena, ulica, dlaň, kosť (pre ženský rod),
    - mesto, srdce, vysvedčenie, dievča (pre stredný rod),
- delenie podľa konkrétnosti,
    - konkrétne (hmotné),
    - abstraktné (nehmotné),
- delenie podľa všeobecnosti,
    - všeobecné podstatné mená,
    - vlastné podstatné mená (predstavujú napríklad názvy miest a mená osôb),
- počítateľnosť,
    - počítateľné,
    - hromadné,
    - pomnožné.

Základný tvar podstatných mien je nominatív jednotného čísla.

Adjektívum (prídavné meno) je plnovýznamový, ohybný slovný druh, ktorý pomenúva statické príznaky veci. Prídavné mená majú rovnaké gramatické kategórie ako podstatné mená. Ich základný tvar je nominatív jednotného čísla. Líšia sa v ďalšom delení:

- Druh,
    - akostné (dajú sa stupňovať),
    - vzťahové (nie všetky sa dajú stupňovať),
    - privlastňovacie (zakončené na –ov, -in) ,
- stupeň (pri stupňovacích prídavných menách),
    - 1.,
    - 2. (zakončené na –ší, -ejší),
    - 3. (predpona naj-, a prípona –ší, -ejší),
- vzor,
    - pekný,
    - cudzí,
    - páví,
    - matkin (otcov).

Pronomen (zámeno) odkazuje na mená, naznačuje osobu, zviera, vec alebo vlastnosť, nepomenúva ich priamo, ale odkazuje na ne v texte. Gramatické kategórie má rovnaké ako podstatné meno a základný tvar je nominatív jednotného čísla. Zámená sa delia na druhy:

- osobné – základné, privlastňovacie,
- zvratné,
- ukazovacie,
- vzťažné (opytovacie zámená v súvetiach),
- opytovacie,
- neurčité,
- vymedzovacie.

Numerále (číslovka) je neohybný aj ohybný, plnovýznamový slovný druh, ktorý vyjadruje množstvo, poradie, počet alebo násobok osôb, zvierat, vecí, dejov a vlastností. Ohybné číslovky majú rovnaké gramatické kategórie a základný tvar ako podstatné mená. Číslovky sa delia na:

- základné,
- skupinové,
- radové,
- druhové,
- násobné.

Verbum (sloveso) je plnovýznamový, ohybný slovný druh, pomenúva činnosť, dej alebo stav. Medzi gramatické kategórie slovies patria:

- osoba – 1., 2., 3.,
- číslo – singulár, plurál,
- čas – minulý, prítomný, budúci,
- spôsob – oznamovací, rozkazovací, podmieňovací,
- vid – nedokonavý, dokonavý, obojvidový,
- slovesný rod – činný, trpný,
- zvratnosť – zvratné sloveso, nezvratné sloveso.

Základný tvar slovies je 1. osoba jednotného čísla.

Adverbium (príslovka) je neohybný, plnovýznamový slovný druh, vyjadrujúci rozličné okolnosti deja. Príslovky sa delia na:

- miesta,
- času,
- spôsobu,
- príčiny.

Prepozícia (predložka) je neohybný slovný druh, ktoré vyjadrujú vzťah medzi slovami. Používajú sa iba v spojení s podstatnými menami, zámenami, číslovkami v určitom páde

a s nadradeným slovesom. Predložky nadobúdajú iný tvar iba vokalizáciou napr.: k – ku, z – zo.

Konjukcia (spojka) je neohybný slovný druh, slúžiaci na spájanie slov a viet, vyjadruje syntaktický vzťah medzi nimi. Spojky rozdeľujeme na:

- priraďovacie – zlučovacie, stupňovacie, odporovacie, vylučovacie,
- podraďovacie,
- spájajúce výrazy.

Partikula (častica) je neohybný slovný druh, ktorý vyjadruje osobitý postoj hovoriaceho k výpovedi. Častice často stoja na začiatku vety a môžu vyjadrovať aj významové odtiene. Vo funkcii častice sa môžu vyskytovať aj iné slovné druhy.

Interjekcia (citoslovce) je neohybný slovný druh, ktorý vyjadruje city, emócie hovoriaceho alebo napodobňujú zvuky. Citoslovcia delíme na:

- vlastné – prvotné, druhotné (utvorené zo slovies alebo podstatných mien),
- zvukomalebné (onomatopoje).

2.1.3 Skladba - syntax

Skladba sa zaoberá spájaním jednotlivých slov (vetných členov) do viet a viet do súvislého textu. Najväčšou jednotkou je text a najmenšou je vetný člen.

V syntaxe určujeme tieto konštrukcie:

- syntagramatické (vetné sklady - syntagramy),
- vetné (druhy viet),
- polovetné,
- súvetné (súvetia).

Vetné sklady sú významové spojenia dvoch plnovýznamových slov, z ktorých je často jedno riadiace a druhé riadené. Sklady delíme na:

- prisudzovací (medzi podmetom a prísudkom),
- určovací (medzi nadradeným a podradeným vetným členom),
- priraďovací (medzi rovnocennými vetnými členmi).

Medzi slovami, tvoriacimi vetný sklad, určujeme gramatické prostriedky:

- zhoda
    - medzi predmetom a prísudkom,
    - medzi nadradeným vetným členom a zhodným prívlastkom, zhodujú sa v rode, čísle a páde,
- väzba – medzi nadradeným vetným členom a predmetom, nadradený vetný člen býva podstatné meno, prídavné meno, zámeno alebo číslovka,

- primkýnanie – medzi nadradeným podstatným menom a nezhodným prívlastkom, medzi prísudkom a prísudkovým určením.

Vety v slovenskom jazyku klasifikujeme podľa troch kategórií: zloženia, obsahu a členitosti. Podľa zloženia vety delíme na:

- jednoduchá veta,
  - holá,
  - rozvitá,
  - s viacnásobným vetným členom,
- súvetie,
  - jednoduché,
    - priraďovacie súvetie (zlučovacie, stupňovacie, odporovacie, vylučovacie),
    - podraďovacie súvetie (s vedľajšou vetou: prísudkovou, podmetovou, predmetovou, príslovkovou, prívlastkovou, doplnkovou) ,
  - zložené (vyjadruje viacej myšlienok).

Vety podľa obsahu klasifikujeme:

- oznamovacia,
- opytovacia,
- rozkazovacia,
- zvolacia,
- želacia veta.

Vety podľa členitosti delíme na:

- jednočlenná (môže byť slovesná alebo neslovesná),
- dvojčlenná (môže byť úplná alebo neúplná).

Vetný člen je základná stavebná jednotka vety. Medzi vetnými členmi sú syntagmatické vzťahy. Základné vetné členy sú podmet a prísudok, ostatné sú rozvíjajúce vetné členy.

Prísudok vyjadruje dej, činnosť alebo stav podmetu. Je vyjadrený slovesom.

Podmet vyjadruje nositeľa činnosti alebo deju určeného prísudkom. Podmetom je najčastejšie podstatné meno alebo zámeno v nominatíve, občas v genitíve.

Predmet je vetný člen, ktorý závisí na slovese (zriedkavo aj od prídavného mena). Predmet sa vyjadruje podstatným menom, zámenom, niekedy slovesom v neurčitku. Predmet je vyjadrený všetkými pádmi, okrem nominatívu.

Prívlastok rozvíja vo vete podstatné meno, zámeno alebo iný prívlastok, bez ohľadu na jeho syntaktickú funkciu. Prívlastok býva vyjadrený prídavným menom, zámenom, číslovkou, neurčitkom slovesa alebo príslovkou.

Prísudkové určenie rozvíja vetné členy, vyjadruje, za akých okolností sa uskutočňuje dej prísudku. Prísudkové určenia delíme na miesta, času, spôsobu a príčiny.

Doplnok sa spája s prísudkom a súčasne s predmetom alebo podmetom vety. Doplnok rozvíja oba vetné členy súčasne.

2.1.4 Modálnosť vety

Modálna stavba vety patrí k jej sémantickému opisu. Modálnosť je spôsob vyjadrovania postojov hovoriaceho k vypovedanej skutočnosti. Modálnosť rozdeľujeme:

- postojová modálnosť (vyjadruje komunikatívny záver hovoriaceho),
    - oznamovací spôsob (veta sa končí bodkou),
    - rozkazovací spôsob (veta sa končí výkričníkom a často obsahuje pobádacie častice),
    - opytovací spôsob (veta sa končí otáznikom, v doplňovacích vetách sa vyskytujú opytovacie zámená, zisťovacie opytovacie častice a inverzia podmetu a prísudku),
- istotná modálnosť (hovoriaci vyjadruje istotu pravdivosti svojej výpovede, vyjadruje sa časticami významovo medzi *áno* a *nie – možno, asi, pravdepodobne*, ...)
- voluntatívna modálnosť (vyjadruje možnosť, potrebu, schopnosť, nevyhnutnosť, vyjadruje sa modálnymi plnovýznamovými slovesami (*musieť, chcieť,...*) alebo modálnymi príslovkami (*treba, netreba, možno,...*))

2.1.5 Nejednoznačnosť slovenčiny

Slovenčina trpí výraznou nejednoznačnosťou, ktorá komplikuje strojovú analýzu prirodzeného jazyka. Ak výsledok analýzy (určenie gramatických kategórii a syntagramatických vzťahov) nie je jednoznačný, pri ďalšom spracovaní musíme pracovať so všetkými možnými alternatívami. Nezávislé alternatívy sa medzi sebou násobia, pričom pamäťová a časová zložitosť analýzy narastá exponenciálne. Podľa [Páleš, 1994] v slovenskom jazyku vznikajú tieto nejednoznačnosti:

- Tvarová homonymita – jeden tvar slova prislúcha viacerým slovným druhom alebo viacerým gramatickým kategóriám.
- Lexikálna homonymita – jedno slovo označuje viacero výrazov.
- Polysémia – jedno slovo nadobúda viacero významov v závislosti od kontextu, v ktorom je použitý (napr.: „*padnúť do jamy*“, „*padnúť za vlasť*“).
- Syntaktická viacznačnosť – jedna veta umožňuje viac spôsobov syntaktického členenia.
- Kontextová nejednoznačnosť – slová vo vete môžu odkazovať na frázy, ktoré ich v kontexte predchádzajú, nasledujú alebo ukazujú na objekty z vonkajšieho sveta (často ide o zámená).
- Metaforickosť.
- Metonymickosť.
- Pragmatická nejednoznačnosť – nejednoznačnosti, ktoré sa dajú vyriešiť iba aplikovaním znalostí o reálnom svete.

3 Existujúce prístupy predspracovania textov v slovenskom jazyku

Pre úspešnú identifikáciu parafrázovania je najskôr potrebné vstupný text rozdeliť na jednotlivé vety a vykonať jeho predspracovanie do vhodnej formy. V tejto kapitole sa zaoberáme metódami a nástrojmi pre počítačové spracovanie prirodzeného jazyka. Ponúkame prehľad existujúcich riešení pre slovenský jazyk.

Medzi základné metódy predspracovania textov v počítačovej lingvistike patrí lematizácia, steming a morfologické značkovanie (*pos-taging*). Lematizácia je určenie základného tvaru slova (*lemy*), steming je určenie koreňa slova (*stem*) a morfologické značkovanie (*pos-taging*) je priraďovanie značiek slovu, ktoré sú zaužívané v morfologickej analýze a vlastne vyjadrujú jeho gramatické kategórie. Slovenčina patrí medzi jazyky s bohatou morfológiou, to znamená, že tvar slova sa mení podľa významu. Pre stematizáciu slov v angličtine je bežne používaný Porterov algoritmus, no ten nie je pre slovenčinu použiteľný.

3.1 Jednoduché metódy predspracovania

Pre identifikáciu parafrázovania na nižšej úrovni nie je potrebné vykonávať morfologické značkovanie, ale stačí použiť jednoduché metódy predspracovania, ktoré sa používajú pri detekcii plagiátorstva, ako napríklad:

- Odstraňovanie stop-slov – z kontrolovaného textu sa odstránia bezvýznamové slova ako napríklad spojky, častice alebo citoslovcia.
- Lematizácia – prevedenie slov do základného tvaru. Existujúce riešenia pre lematizáciu sú uvedené v rámci tejto kapitoly.
- Nahrádzanie synoným – slová, ku ktorým je nájdené v slovníku synonymum, sú nahradené vždy tým istým synonymom zo skupiny, napríklad sa vyberie vždy to, ktoré je z nich prvé v abecede.

3.2 SAPFO

SAPFO[Páleš, 1994] je parafrázovač slovenčiny, ktorý bol vyvíjaný RNDr. Emil Pálešom v prologu. Cieľom tohto nástroja je vygenerovať všetky parafrázy k danej vete, ktoré dávajú v slovenskom jazyku zmysel. Ide o komplexný nástroj, ktorý zahŕňa fonológiu, morfológiu, syntax, sémantiku a štatistické metódy na analýzu aj syntézu prirodzeného jazyka.

SAPFO obsahuje údajovú a procedurálnu časť. Údajovú časť tvoria slovníky: morfologický, lexikálno-sémantický, slovotvorný, syntakticko-väzobný, valenčný, synonymický, frazeologický, štylistický a konotačný slovník. Procedurálnu časť tvoria implementácie algoritmov morfologickej, morfematickej, syntaktickej, sémantickej a kontextovej analýzy.

Morfologický slovník obsahuje ku každej ohybnej lexikálnej jednotke gramatické kategórie podľa jednotlivých slovných druhov. Gramatické kategórie sú v mnohom podrobnejšie ako nami uvedené v kapitole Tvaroslovie a gramatické kategórie. Napríklad SAPFO používa 132 vzorov pre slovesné podstatné mená, pretože zahŕňa aj vzory pre dvojtvary podstatných mien, 24 jednoznačne definovaných slovesných vzorov, 7 skloňovacích vzorov a 6 stupňovacích vzorov pre prídavné mená. Všetky tieto vzory sú uvedené v [Páleš, 1994].

Lexikálno-sémantický slovník obsahuje ku každej uvedenej lexikálnej jednotke súbor sémantických príznakov, ktorý každú lexikálnu jednotku zaraďuje do určitých sémantických tried. Sémantické triedy majú hierarchickú štruktúru a uplatňuje sa pri nich dedičnosť, napríklad ak je lexikálna jednotka zaradená do sémantickej triedy *osoba*, tak je súčasne zaradená aj v sémantickej triede *životné*. Samotné sémantické triedy sú často neostré a preto býva lexikálna jednotka zaradená do viacerých tried súčasne. Pre každý slovný druh má SAPFO osobitný lexikálno-sémantický slovník s rôznymi sémantickými triedami.

Slovník syntaktických väzieb obsahuje informácie o väzobných prostriedkoch slovenského jazyka. Používa sa pri syntaktickej analýze aj syntéze. Obsahuje gramatické pády predložiek, pádové väzby substantív a dvojice parataktických spojok (*sotva-už*, *raz-a zase*).

Slovník slovotvorných hniezd zaraďuje lexiku do slovotvorných hniezd. Obsahuje informácie o vzťahoch odvodenosti medzi jednotlivými slovami. V niektorých prípadoch nie je zrejmé, od ktorého slova bolo dané slovo odvodené, SAPFO daný problém rieši tak, aby dané slovo bolo čo najviac v súlade s pravidlami slovenskej gramatiky. Pretože deriváty slova mu nemusia významovo zodpovedať, sú slovotvorné hniezda rozdelené do sémantických kategórií, napríklad slovo *srdce* ma niekoľko slovotvorných hniezd, a to s významami: telesný orgán, schopnosť cítiť, predmet srdcového tvaru, centrálne položené miesto a srdce zvonu. Takýmto rozdelením slovník pomáha aj pri odstraňovaní nejednoznačností v syntaktickej analýze.

Synonymický a lexikálno-paradigmický slovník je databázou paradigmických vzťahov v lexike. Tie sú zachytené formou lexikálnych parametrov, čo sú systematicky opakujúce sa vzťahy v lexike. Obsahuje napríklad vzťahy: *zlodej - kradnúť, vrah – zabiť*,...

Frazeologický slovník obsahuje informácie, ktoré využíva frazeologický modul programu SAPFO na analýzu a syntézu parafrázovania pomocou frazeologických zvratov. Snaží sa fráze priradiť jedno slovo, ktoré by ju vystihovalo. Napríklad: „*kosť a koža*“ – „*chud-ý/á/é*“.

Slovník asociatívnych väzieb akonotácií je slovník, ktorý obsahuje často opakujúce sa asociácie. O asociáciách je možné uvažovať aj medzi ľubovoľnými slovnými druhmi. SAPFO ich využíva na rekonštrukciu informácie, ktorá bola stratená počas syntézy novej parafrázy. Obsahuje napríklad dvojice: *chlieb - jesť, víno – piť*. Asociatívny slovník môže vzniknúť aj frekvenčnou analýzou korpusu textov.

Tieto slovníky využíva procesná časť programu SAPFO na analýzu viet a syntézu parafráz. Procedurálna časť sa skladá z niekoľko spolupracujúcich modulov.

Morfologický modul zabezpečuje morfologickú analýzu a syntézu. Morfologická analýza znamená k danému slovu určiť všetky jeho gramatické kategórie. Morfologická analýza angličtiny sa rieši slovníkovou metódou, ale pri flexívnych jazykoch, ako je slovenčina, by bol nárast veľkosti slovníka neúnosný. Preto sa používa kombinovaná slovníkovo-algoritmická analýza. SAPFO sa pokúsi vykonať stematizáciu a následne podľa afixov pôvodného slova určí gramatické kategórie. Pokúsi sa odtrhnúť:

- 0 až 5 znakov predpony,

- 0 až 4 znakov prípony,
- 0 až 3 znaky ako relačnú príponu.

Výsledky po odrhnutí afixov porovnáva so slovníkom, po nájdení koreňa z odtrhnutých afixov odvodí gramatické kategórie. Algoritmus morfologickej analýzy je implementovaný ako nedeterministický stavový automat.

Morfologická syntéza znamená vytvorenie daného tvaru slova na základe gramatických kategórií.

Syntaktický modul zabezpečuje syntaktickú analýzu a syntézu. Syntaktická analýza identifikuje vetné členy, ich skladbu do fráz (syntagiem) a prideľuje im syntaktické roly (vyjadrené vetným členom). Jej výsledkom je syntaktický (derivačný) strom, ktorý vyjadruje vzťahy medzi jednotlivými vetnými členmi. Gramatika slovenského jazyka sa nachádza medzi bezkontextovou a kontextovou formálnou gramatikou.

Na prvé experimenty so syntaktickou analýzou v programe SAPFO bola použitá bezkontextová gramatika, pretože pomocou zásobníkových automatov dokáže spracovávať jazyk v polynomiálnom čase. Pri syntaktickej analýze je možné postupovať zdola-nahor (začína sa počiatočným symbolom pre vetu a prepisuje ľavé strany pravidiel pravými, až kým nedostane samé terminálne symboly) a zhora-nadol (syntaktická analýza sa začína terminálmi a snaží sa dostať počiatočný symbol, prepisovacie pravidlá sa používajú opačným smerom). Syntaktická analýza zdola-nahor má schopnosti predikcie, pretože nikdy neuvažuje o gramatických kategóriách na pozíciách, kde nemôžu stáť. No trvá jej nejaký čas, kým sa vôbec dostane k terminálom. Syntaktická analýza zhora-nadol zas nekonštruuje žiadnu frázu dvakrát, no uvažuje aj o frázach, z ktorých nemôže vzniknúť syntakticky správna veta.

Ďalšie experimenty boli zamerané na syntaktickú analýzu pomocou rozšírených prechodových sietí. Prechodová sieť (ang. *Transition Network* - TN) je graf, ktorého hrany sú označené slovnodruhovými a gramatickými kategóriami. Viacnásobné vetné členy vyjadruje slučkou, fakultatívne členy prázdnymi hranami, dokáže vyjadriť nepovinnosť a opakovanie. Rozšírená neurónová sieť (ATN) „*je syntaktický analyzátor pracujúci na princípoch ‚zľava doprava' a ‚zhora nadol'. Pozostáva z orientovaného grafu, zoznamu podmienok a zoznamu akcií. Vrcholy grafu zodpovedajú stavom automatu, do ktorých sa analyzátor môže dostať. Ku každej hrane je priradená podmienka, pri ktorej možno cez hranu prejsť, a akcia, ktorú treba pritom vykonať.*"[Páleš, 1994] Rozšírená prechodová sieť zo vstupu číta slová a zároveň postupuje v sieti medzi uzlami cez hrany a súčasne vytvára syntaktický (derivačný) strom. Ak je z daného stavu, reprezentovaného aktuálnym uzlom, viac prechodov, vyberie si jeden a ostatné uloží na zásobník. Ak sa ocitne v stave, z ktorého jej podmienky nedovoľujú traverz po žiadnej s výstupných hrán, tak sa zo zásobníka vyberie stav a ATN ním pokračuje. Ak je zásobník prázdny a hlava automatu sa nemôže pohnúť ďalej pozdĺž hrán, tak veta nepatrí do daného jazyka. Ak sa hlava dostane na koniec prechodovej siete aj na koniec analyzovanej vety, tak veta patrí do daného jazyka.

Výsledkom experimentov so systémom SAPFO boli rozšírené prechodové siete pre slovenčinu [Páleš, 1990]. Sieť pozostáva z ôsmich podsietí, ktoré navzájom komunikujú a môžu sa rekurzívne volať.

Modul sémantickej analýzy sa zaoberá analýzou významu vety. Hlavným nositeľom významu vety je prísudok, ktorý je považovaný za centrum a hlavného člena vety. Okolo neho sú združené particienty, ktoré sa zúčastňujú na deji alebo význame vety. Aj particient môže byť štruktúrovaný podobným spôsobom. Každý particient vystupuje k hlavnému členu vety (alebo frázy) v určitom sémantickom páde. Sémantická analýza znamená vo vete identifikovať particienty a ich sémantické pády. Ako particient môže vystupovať aj celá vedľajšia veta.

SAPFO pracuje s množinou 66-tich sémantických pádov. Každý vzťah z reálneho sveta má zobrazenie v jednom z týchto abstraktných sémantických pádov. Tieto pády vznikli na základe prác C. J. Fillmora (1968), Winograda (1983),E. Tibenskej (1988) a experimentov so systémom SAPFO.

Veľkým problémom sémantickej analýzy sú združené pomenovania (napr.: *združené pomenovanie*, *vysoká škola*, *vlčí mak*), pretože sa nedajú identifikovať formálne, iba slovníkom. SAPFO ich má vymenované v lexikálno-sémantickom slovníku. Nepodliehajú regulárnej syntaxi a sú chápané ako samostatné lexémy, do syntaktickej analýzy vstupujú ako združené pomenovania. No analyzátor nie je úspešný v prípadoch, keď ide zdanlivo o združené pomenovanie, ale v skutočnosti ním nie je. Napríklad veta „*Chodím do tej vysokej školy.“*.

Ďalším problémom sú ustálené pomenovania – frázy (*mlčať ako hrob, z ruky do úst, ...*). Tie sa nespracovávajú ako samostatné lexémy, ale pomocou frazeologického slovníka sa ich SAPFO pokúsi nahradiť jedným významovým slovom („*vo dne v noci*“ → *neustále*). Ak ide o vetné frázy, SAPFO sa ich ani nepokúša spracovať.

Sémantický modul dopĺňa informácie aj o modálnosti vety. SAPFO sa snaží využiť interpunkčné znamienka a častice opísané v kapitole 2.1.4, navyše istotnú modálnosť vyjadruje číslom v intervale *<0,1 >*(*rozhodne = 1.00, zaiste = 0.90, zrejme = 0.75, snáď = 0.30,horkýže = 0.05*). Ale niekedy uvedené znaky modálnosti nestačia na určenie modálnosti. Napríklad SAPFO nedokáže rozpoznať rečnícke otázky, gnómickú osobu, autorský plurál, plurál majestaticus a familiárny plurál. Takisto nerozpozná ani tretiu osobu vo funkcii druhej osoby („*Slečna sa na mňa hnevá?“*), gnómický čas, historický prézens, prézens v zastúpení futúra, budúci čas vo funkcii odhadu, gnómický imperatív a minulý čas ako zosilený imperatív.

Integrovaná syntakticko-sémantická analýza v systéme SAPFO slúži na skoré odstránenie nejednoznačností počas analýzy. Táto myšlienka vychádza z toho, že ľudská myseľ vníma jazyk ako celok a tak ho aj spracováva. Integrovaný syntakticko-sémantický analyzátor sa nezastavuje pri prideľovaní syntaktických rolí, nevytvára syntaktický strom, ale prideľuje sémantickú rolu každej podfráze ihneď, ako na ňu narazí. Syntaktický analyzátor hneď spolupracuje so sémantickým modulom. Keď syntaktický analyzátor identifikuje nejakú

podfrázu (napríklad združené pomenovanie), vyvolá sémantický modul, aby identifikáciou sémantickej roly potvrdil jej sémantickú prípustnosť. Ak sémantický modul zistí, že podfráza je sémanticky nezmyselná, upozorní syntaktický modul, že pracovať s touto podfrázou je neperspektívne. Vylučovaním nezmyslených syntagiem SAPFO výrazne znižuje nejednoznačnosti v analýze vety a tým zvyšuje aj jej efektivitu.

RNDr. Emil Páleš nazýva valenčným poľom (podobne ako v chémii) vetného člena tú časť vety, v ktorej sa nachádzajú aktanty sémanticky podradené tomuto vetnému členu. Valenčné polia SAPF-u umožňujú priraďovať sémantické pády okamžite. Pre identifikáciu valenčných polí treba poznať centrálny vetný člen, čo pri LR-syntaktickej analýze nie je možné, lebo veta sa spracováva postupne zľava a podradené vetné členy sa často vyskytujú skôr ako nadradené. Identifikácia valenčných polí prebieha nasledovnými algoritmami:

Algoritmus na identifikáciu centrálneho slovesa:

1. Nájdi prvé nemodálne, nefrázové určité sloveso alebo neurčitok, ignorujúc všetko medzi dvoma čiarkami.

Algoritmus 1 – Identifikácia centrálneho slovesa

Algoritmus na identifikáciu centrálneho substantíva:

1. Nájdi najľavejšie podstatné meno.
2. Ak za ním nasleduje podstatné meno v rovnakom páde, druhé meno je hlavné, prvé je prístavok.
3. Ak nasleduje slovesné príčastie, musia sa zhodovať v rode, čísle a páde, inak pokračuj v hľadaní.

Algoritmus 2 – Identifikácia centrálneho substantíva

Kontextová analýza zisťuje, ako sa frázy použité vo vete vzťahujú k objektom reálneho sveta. Prideľuje frázam reálne objekty vonkajšieho sveta, respektíve zgrupuje frázy, ktoré majú vzťah ku tomu istému objektu reálneho sveta.

Prepojenie analýz prináša výrazný synergický efekt a zníženie nejednoznačností. Pri analýze systému SAPFO sme sa zamerali hlavne na jeho analytickú časť. Časť pre syntézu parafráz sme vynechali. SAPFO je veľmi zložitý softvérový systém s množstvom algoritmov a bohatými slovníkmi. Myslíme si, že nie všetky atribúty vety, ktoré analyzuje, sú potrebné pre identifikáciu parafrázovania, pretože mnoho z nich sa využíva hlavne pri syntéze nových parafráz.

3.3 Hunspell

Hunspell [Hunspell, 2011] je nástroj na kontrolu pravopisu (*spellchecker)*, steming a morfologickú analýzu v morfologicky zložitých jazykoch. Vychádza z projektu s otvoreným kódom (*opensourse) MySpell* a pôvodne bol vyvíjaný pre maďarský jazyk, ale postupne bol rozšírený pre mnohé iné jazyky, medzi inými aj slovenčinu. V súčasnosti Hunspell používa na kontrolu pravopisu mnoho projektov s otvoreným kódom, napríklad: *OpenOffice*.

Hunspell funguje na slovníkovom princípe, pre slovenský jazyk obsahuje okolo 260 000 slov, no z toho sú priradené gramatické kategórie okolo 66 000 slovám. Dokáže však vykonávať steming aj lematizáciu algoritmicky pre slová, ktorých základný tvar je obsiahnutý v slovníku.

3.4 Tvaroslovník

Tvaroslovník je nástroj na jednoduchú lematizáciu a slovný rozbor slovenčiny, vyvíjaný v rámci projektu NAZOU [1], ktorý rieši problémy spracovania informačných zdrojov v slovenskom jazyku. Jeho pilotnou aplikáciou je vyhľadávanie pracovných ponúk na internete. Tie sa vyhľadávajú, indexujú, ďalej spracovávajú vo forme ontologických metadát a nakoniec prezentujú používateľovi. Práve tu je potreba algoritmického stemingu, pretože názvy firiem, mená a priezviská môžu byť pred indexovaním vyskloňované.

Tvaroslovník je pravidlový lematizér, ohýba slová podľa šablóny. Jeho veľkou výhodou je možnosť lematizácie aj novovytvorených slov alebo vlastných podstatných mien, ktoré by sa pri slovníkovej lematizácii nenachádzali v príslušnom slovníku. Jeho nevýhodou je možnosť nepresností, spôsobených výberom zlej šablóny.

3.5 Slovenský WordNet

Slovenský WordNet [SAV, 2014] je slovník často sa vyskytujúcich slovenských slov, ktoré sú usporiadané podľa sémantických vzťahov. Vyvíja ho *Jazykovedný ústav Ľ. Štúra Slovenskej akadémie vied*. V súčasnosti obsahuje okolo 2 500 synsetov, no stále ide len o pilotnú verziu, ktorá je vytvorená preložením anglického WordNetu.

3.6 Skryté Markovovské modely

Skryté Markovovské modely sú modely pre modelovanie stochastických procesov. Ide o model založený na pravdepodobnostnom konečnom automate. Skrytý Markovovský model sa skladá z množiny stavov, množiny výstupných symbolov, množiny pravdepodobnostných prechodov z jedného stavu do druhého a pravdepodobnosťou generovania výstupných symbolov v jednotlivých stavoch. Prechody medzi stavmi sú často vyjadrené prechodovou maticou. Pravdepodobnosť nasledujúceho stavu závisí len na aktuálnom stave a nezávisí od sekvencie predchádzajúcich stavov. Skrytý Markovovský model si možno predstaviť ako čiernu skrinku, ktorá generuje postupnosť výstupných symbolov (Markovov reťazec) a postupnosť stavov, cez ktoré prechádza, sú pre pozorovateľa skryté.

Problém zistenia najpravdepodobnejších morfologických značiek sa dá previesť na problém hľadania najpravdepodobnejšej postupnosti prechodov medzi stavmi, ktoré predstavujú jednotlivé morfologické značky a generujú jednotlivé slová vety. Množinu stavov predstavujú jednotlivé morfologické značky, abecedu predstavujú slová spracovávaného textu a pravdepodobnosti prechodov sa získajú z korpusu doplneného o morfologické značky.

Hľadania najpravdepodobnejších prechodov medzi stavmi rieši Viterbiho algoritmus.

[1] http://nazou.fiit.stuba.sk/

```
function viterbi(O,S,π,A,B)
 t = lenght(O);
 foreach i in S do
  T1[i,1] = π[i]*B[i,O[1]];
  T2[i,1] = 0;
 end
 for i=2 to t do
  foreach j in S do
   T1[j,i] = max {T1[k,i-1]*A[k,j]*B[j,O[i]] foreach k in S};
   T2[j,i] = arg max {T1[k,i-1]*A[k,j]*B[j,O[i]] foreach k in S};
  end
 end
 Z[t] = arg max {T1[k,t] foreach k in S};
 X[t] = S[Z[t]];
 for i=t downto 2 do
  Z[i-1] = T2[Z[i],i];
  X[i-1] = S[Z[i-1]];
 end
 return X;
end
```

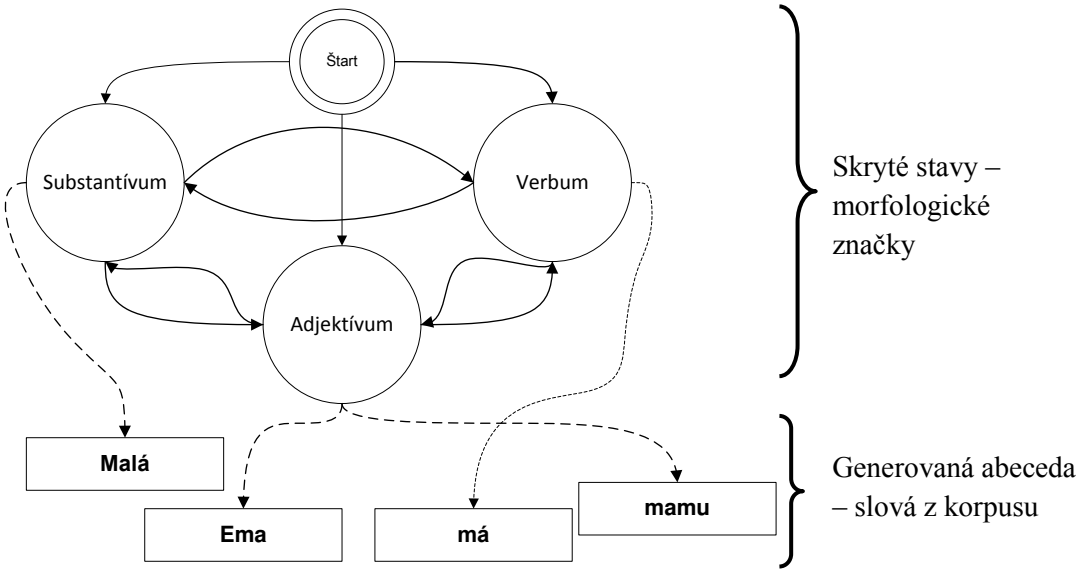
Algoritmus 3 - Pseudokód pre Viterbiho algoritmus

Vstupom do Viterbiho algoritmu je:

- O – vstupný Markovov reťazec, v našom prípade postupnosť slov vo vete,
- S – množina všetkých stavov – morfologické značky,
- π – množina pravdepodobností začiatočných stavov, kde $\pi[i]$ je pravdepodobnosť začiatočného stavu $i \in S$,
- A – matica pravdepodobností prechodov, kde $A[i,j]$ je pravdepodobnosť prechodu zo stavu i do stavu j, pričom $i, j \in S$, je získaná z korpusu,
- B – matica pravdepodobností generovania výstupného symbolu u v stave i.

Výstupom je postupnosť najpravdepodobnejších stavov.

Zaujímavým pokusom je projekt s názvom *Štatistický Part-Of-Speech tagger* od Ing. Martina Jačalu, využívajúci skryté Markovovské modely na morfologické značkovanie slov vo vetách. Tento konkrétny morfologický analyzátor pracuje iba so slovnými druhmi. Pri dostatočne veľkej trénovanej vzorke sa úspešnosť tejto metódy pohybuje okolo 87%.

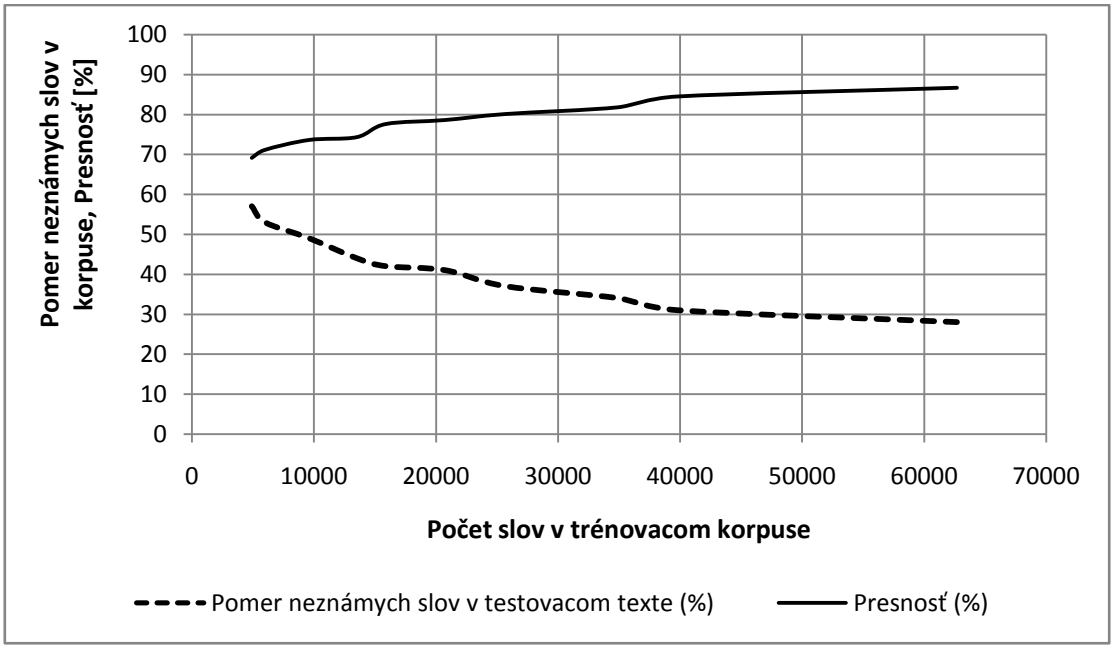


Obrázok 3.1 - Princíp určovania slovných druhov pomocou skrytých Markovovských modelov

Postup realizácie, ktorý zvolil Ing. Martin Jačal, začína vytvorením lexikónu z trénovacieho korpusu, ktorý obsahuje všetky unikátne kombinácie slovo – morfologická značka. V prípade, že jedno slovo má viacej morfologických značiek, sa pre každú morfologickú značku vypočíta pravdepodobnosť jej výskytu. Lexikón ďalej obsahuje pravdepodobnosti prechodov medzi morfologickými značkami. Samotné priraďovanie morfologických značiek je realizované pomocou Viterbiho algoritmu.

Pre vytvorenie skrytého Markovovského modelu bola použitá časť *Slovenského národného korpusu*, ktorá má doplnené v texte morfologické značky – *Ručne morfologicky anotovaný korpus* (*r-mark)*. Problém tohto korpusu sú chyby v morfologickej analýze a niekedy aj nekonzistentnosti v jej označovaní.

Výsledky *Štatistického Part-Of-Speech taggera* sú na nasledujúcom grafe (Obrázok 3.2). Graf vyjadruje presnosť morfologického značkovania v závislosti od počtu slov v trénovacom korpuse a pomeru neznámych slov, v testovacom texte (neznáme slová sa nenachádzali v trénovacom korpuse).



Obrázok 3.2 - Úspešnosť nástroja Štatistický Part-Of-Speech tagger

4 Metódy identifikácie parafrázovania

V tejto kapitole sa venujeme rôznym metódam identifikácie parafrázovania. Opisujeme jednoduchšie, ktoré sa často využívajú pri detekcii plagiátorstva, no zamerali sme sa najmä na metódy využívajúce strojové učenie, pretože ich považujeme za najperspektívnejšie pre identifikáciu parafrázovania v slovenskom jazyku. V závere kapitoly uvádzame prínosy jednotlivých metód.

4.1 n-gramy, skip-gramy

Na identifikáciu parafrázovania na nižšej a čiastočne aj strednej úrovni sa dajú použiť metódy, ktoré sa používajú na detekciu plagiátorstva, spolu s predspracovaním textov uvedených v kapitole 3.1 a to najmä nahrádzaním synoným. Je možné použiť kombináciu nasledujúcich čiastkových metód:

- *n-gramy* – n-gram je postupnosť *n* za sebou idúcich slov (alebo znakov, no v prípade identifikácie parafrázovania uvažujeme iba o slovách), pričom nasledujúci n-gram je posunutý oproti predchádzajúcemu, tento parameter posunu nazývame offset a susedné n-gramy sa môžu prekrývať, pri kontrole plagiátorstva sa kontrolujú medzi sebou jednotlivé n-gramy [Češka, 2007]. V detekcii plagiátorstva sa najčastejšie používajú 3-gramy s offsetom 1.
- *skip-gramy* – v mnohom sa podobajú na n-gramy, no slová tvoriace skip-gram nenasledujú za sebou, ale majú medzi sebou jedno alebo viaceré slová.
- *LCS* – najdlhšia spoločná podpostupnosť (*longest common subsequence*), v dvoch reťazcoch hľadá najdlhšiu spoločnú podpostupnosť slov (alebo znakov). Okrem počtu zhodných slov označí aj vzájomné prekrytie. LCS býva najčastejšie implementované pomocou dynamického programovania alebo Hyrsbergovho algoritmu. Rekurzívna formalizácia algoritmu je:

$$c[i,j] = \begin{cases} 0 & , \quad ak\ i = 0 \vee j = 0 \\ c[i-1, j-1] + 1, & ak\ i,j > 0 \wedge x[i] = y[j] \\ \max\{c[i-1,j], c[i,j-1]\}, & inak \end{cases}$$

  kde x a y sú vstupné reťazce a c je pomocné dvojrozmerné pole.

Tieto metódy využíva aj nástroj PlaDeS [Chudá, 2011], ktorý je určený na detekciu plagiátorstva v slovenskom jazyku.

4.2 Identifikácia parafrázovania pomocou váhovaných závislostí slovnej sémantiky

Tento prístup, opísaný pre anglický jazyk v [Lintean, 2009], deteguje parafrázovania pomocou podobností a odlišností medzi dvoma vetami. Podobnosť a odlišnosť sa určuje na základe sémantických a syntaktických vzťahov v rámci viet aj medzi nimi navzájom. Vychádza z myšlienky, že ak sú si vety podobné a majú veľa spoločných syntaktických a sémantických vzťahov, tak sú si významovo príbuzné a od istej hodnoty takejto podobnosti sa jedná o parafrázovanie.

Špecifikom uvedeného prístupu, oproti obdobným, je priraďovanie váh dvojiciam slov, medzi ktorými nie je závislosť. Táto metóda získava informácie o slovnej sémantike z WordNet-u, čo je lexikálna databáza opisujúca sémantické vzťahy najfrekventovanejších slov. Čím bližšie sú slová v rámci WordNet-u, tým sú si sémanticky podobnejšie.

Autor uvádza, že ak sú vety parafrázované, mali by mať veľa spoločných syntaktických vzťahov a málo rozdielnych slov. Na druhej strane existujú vety, ktoré sú takmer identické a to lexikálne aj syntakticky, no napriek tomu nie sú parafrázy.

Pre identifikáciu parafrázovania medzi dvoma vetami S_1 a S_2 sa používa nasledujúca postupnosť krokov:

1. Vykoná sa analýza vety pomocou Stanford-syntaktického analyzátora. Ten analyzuje čistý text vety a vráti zoznam vzťahov medzi jednotlivými slovami. Tieto vzťahy medzi slovami v rámci vety sa prevádzajú do stromovej reprezentácie. Podmet sa pokladá za hlavný vetný člen vo vete a preto je v tomto strome vzťahov koreňom.
2. Pre vety S_1 a S_2 sa identifikujú podobnosti $sim(S_1, S_2)$ a odlišnosti $diss(S_1, S_2)$, na základe slovnej sémantiky a lexiky slov. Podobnosť viet $sim(S_1, S_2)$sa počíta iteratívne pre slová d_1 z vety S_1 a slová d_2 z vety S_2.

$$sim(S_1, S_2) = \sum_{d_1 \in S_1} \max_{d_2 \in S_2} \left(d2dSim(d_1, d_2) \right)$$

   Kde $d2dSim(d_1, d_2)$ je funkcia mapujúca sémantickú podobnosť slov d_1 a d_2 z informácií získaných pomocou WordNet-u, syntaktických vzťahov vypočítaných v predchádzajúcom kroku a empiricky nájdených váh. Pri výpočte odlišnosti sa pre všetky nenamapované slová (vráti ich funkcia $unmap(S)$) z oboch viet sčítajú váhy (funkcia $weight(d)$ vráti váhu pre slovo). Tieto váhy sa vypočítajú pomocou hĺbky slova v strome závislostí a empiricky určených hodnôt pre morfologické kategórie slova d.

$$diss(S_1, S_2) = \sum_{\substack{d \in \{unmap(S_1), \\ unmap(S_2)\}}} weight(d)$$

3. V poslednom kroku sa pomer podobnosti a rozdielnosti viet porovnáva s prahovou hodnotou T.

$$\frac{sim(S_1, S_2)}{diss(S_1, S_2)} > T$$

   Ak je $diss(S_1, S_2) = 0$, tak sú vety S_1 a S_2 identické, lebo neobsahujú žiadne rozdielne slovo. Ak je pomer podobnosti a rozdielnosti väčší ako hodnota T, tak sa tieto dve vety považujú za parafrázy. Prahová hodnota T je určená empiricky z korpusu dokumentov.

Experimenty s týmto systémom boli uskutočnené na Microsoft korpuse a výsledky autor porovnával s obdobnými systémami na identifikáciu parafrázovania. Systém dokázal identifikovať parafrázovanie s presnosťou 71,3% a návratnosťou 89,8%, presnosť ostatných porovnávaných systémov sa pohybovala medzi 70% až 72%.

Nevýhodou opísaného prístupu je závislosť prahovej hodnoty *T* na doménovej oblasti a použitom korpuse. To znamená, že ak sa hodnota *T* vhodne určí pre jeden korpus, identifikácia parafrázovania na inom korpuse s touto hodnotou môže dávať zlé výsledky.

4.3 Identifikácia parafrázovania pomocou strojového učenia

V tejto podkapitole sa venujeme použitiu algoritmov strojového učenia pri identifikácii parafrázovania. Rozoberáme v nej aj použité algoritmy, najmä Algoritmy podporných vektorov (SVM) a ich spôsobu použitia a dosahovaných výsledkov.

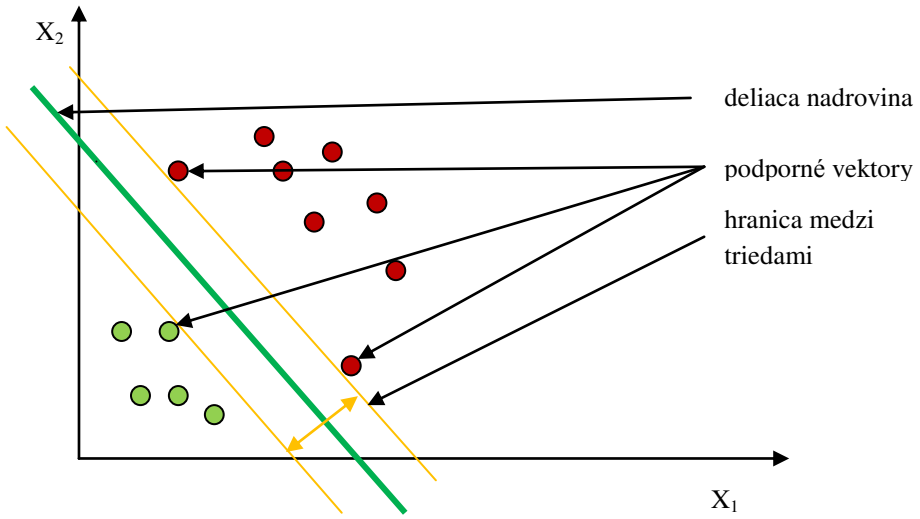
4.3.1 Algoritmy podporných vektorov (SVM)

Algoritmy podporných vektorov (*support vector machines*) patria k relatívne novým metódam strojového učenia s učiteľom. SVM vychádza z myšlienky kombinácie výhod rýchleho nájdenia deliacej nadroviny pri klasifikácii lineárne separovateľných problémov a schopnosťou pracovať s nelineárne separovateľnými problémami. SVM je binárny klasifikátor, to znamená, že vstupné dáta (*n*-rozmerné vektory) dokáže klasifikovať do dvoch tried.

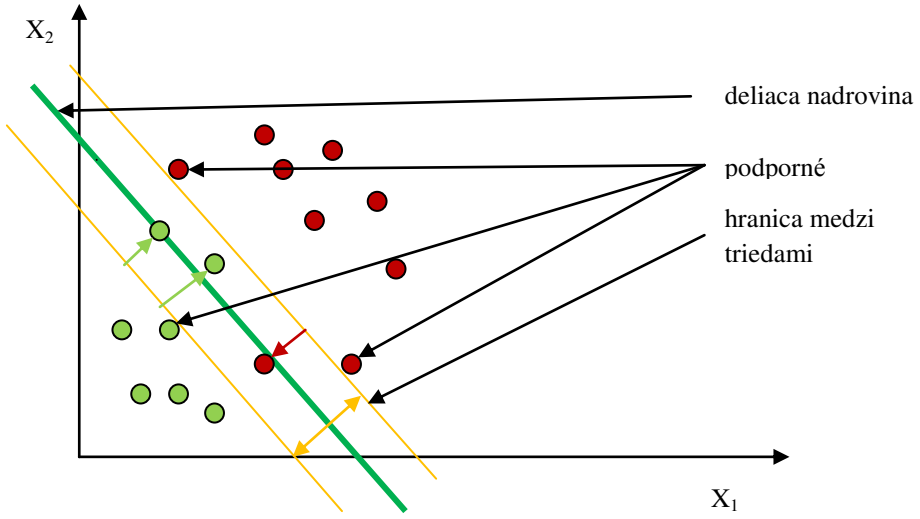
Jedným zo základných princípov SVM je prevedenie pôvodného priestoru do priestoru s oveľa vyššou dimenziou, kde sa už dajú rozdeliť triedy pomocou deliacej nadroviny. Prevod vstupných hodnôt do vyšších dimenzií zabezpečuje kernel funkcia.

SVM sa pri učení snaží nájsť optimálnu deliacu nadrovinu, ktorá maximalizuje hranice medzi triedami. Ak ju nedokáže nájsť, prevedie pôvodný problém v n-rozmernom priestore na omnoho vyšší, v ktorom je možné lineárne separovať triedy nadrovinou. Pri mapovaní do priestoru s dostatočne vysokým počtom dimenzií je vždy možné nájsť nadrovinu, ktorá oddeľuje triedy problému. Lineárna nadrovina sa hľadá pomocou metódy kvadratického programovania.

Nie všetky problémy sú aj vo vysoko dimenziálnom priestore jednoducho separovateľné, preto SVM dovoľuje rozdeliť nadrovinou aj nelineárne separovateľné problémy tým, že sa zavedú tolerančné premenné. Tie umožnia, aby sa niektoré vstupné vzory nachádzali za klasifikačnými hranicami, a tieto vzory sa penalizujú. Takéto SVM sa nazývajú *Soft-Margin SVM*. Oproti *Hard-Margin SVM*, ktoré neumožňujú takéto zjednodušenie vstupného problému, nájdu vždy riešenie a sú robustnejšie voči zašumeným vstupom.



Obrázok 4.1 - Princíp fungovania SVM s tvrdými hranicami (hard-margin SVM)



Obrázok 4.2 - Princíp fungovania SVM s mäkkými hranicami (soft-margin SVM)

Proces mapovania pôvodného priestoru do vyššej dimenzie využíva kernel funkciu. Najznámejšie kernel funkcie sú:

- Lineárny kernel: $K(x_i, x_j) = x_i^T x_j + c$
- Polynomiálny kernel: $K(x_i, x_j) = (x_i^T x_j + 1)^p$
- Gaussov (RBF) kernel: $K(x_i, x_j) = e^{(-\frac{|x_i - x_j|^2}{2\sigma^2})}$
- Sigmodiálny kernel: $K(x_i, x_j) = than(\beta_0 x_i^T x_j + \beta_1)$

SVM majú využitie napríklad pri rozpoznávaní obrazu, efektívne pracujú aj pri úlohách s veľkým počtom parametrov. V praxi sa úspešne používajú napríklad na filtrovanie textových dokumentov, detekciu spamu v mailoch a podobne.

4.3.2 k-najbližších susedov

Metóda k-najbližších susedov je metóda strojového učenia pre klasifikáciu. Vzorky trénovacej množiny sú *n*-rozmerné vektory. Keď chce klasifikátor určiť cieľový atribút neznámej vzorky (vektora), hľadá *k* vzoriek z trénovacej množiny, ktoré sú najbližšie k neznámej vzorke (vektoru). Na určenie vzdialenosti neznámej vzorky (vektora) *X* od vzorky *Y* sa používajú rôzne miery, napríklad Euklidovská alebo Manhattanovská.

Výsledná trieda, do ktorej patrí vzorka *X*, sa určí ako najpočetnejšia trieda z *k* najbližších susedov.

Táto metóda má niekoľko nevýhod, napríklad pri procese klasifikácie musí mať k dispozícii všetky inštancie trénovacích vzoriek, všetky atribúty vzoriek majú rovnakú váhu, čo môže viesť k chybným klasifikáciám. Ďalším problémom je takzvaná kliatba rozmerov.

4.3.3 Maximálna entropia

Hlavnou myšlienkou maximálnej entropie je nájsť také podmienené rozdelenie pravdepodobnosti, ktoré má za daných podmienok maximálnu entropiu. Ide o to nájsť čo najjednoduchší opis toho, čo už poznáme, ten je najbližšie k rovnomernému rozdeleniu, teda má čo najvyššiu entropiu.

Model maximálnej entropie je často používaný v počítačovej lingvistike, pretože si vystačí aj s malým množstvom dát. Dá sa použiť napríklad aj na dopĺňanie morfologických značiek na základe okolitého textu.

4.3.4 Použitie strojového učenia na identifikáciu parafrázovania

V článku [Kozerva, 2006],ktorý sa venuje identifikácii parafrázovania v anglickom jazyku, sa uvádza, že SVM sa dá použiť na identifikáciu parafrázovania na nižšej úrovni, pričom vstupom do tohto procesu je čistý text. Jadrom tohto článku boli experimenty s predspracovaním viet, použitím strojového učenia (okrem SVM použili aj k–najbližších susedov a maximálnu entropiu) a vplyv týchto metód na presnosť identifikácie parafrázovania.

V experimentoch porovnávali vety z Microsoft korpusu, pre učenie bolo vybraných 4 076 vetných párov, z toho bolo 2 753 parafrázovaných. Na textovanie bolo použitých 1 726 iných vetných párov, z toho 1 147 bolo parafrázovaných.

Vstupom do algoritmov strojového učenia boli dvojice viet a k nim nasledujúce vlastnosti:

- počet zhodných slov vo vetách k celkovému počtu slov,
- prekryv slov vo vetách nájdený 4-skip-gramami,
- prekryv slov vo vetách nájdený algoritmom LCS (najdlhšia spoločná podpostupnosť), pretože v angličtine má pozícia vo vete syntaktický význam,
- sémantická podobnosť medzi slovami,
- veta, v ktorej sa nachádza vlastné podstatné meno, dostala príznak 1, ak sa v nej nenachádza, dostala príznak 0.

Pri všetkých experimentoch boli dosiahnuté najlepšie výsledky pomocou lineárneho kernelu. Pri metóde k-najbližších susedov sa používa normalizovaná Manhattanská vzdialenosť:

$$max_i = \max\{y_i | y_i \in \forall Y\}$$

$$min_i = \min\{y_i | y_i \in \forall Y\}$$

$$d(X,Y) = \sum_{i=0}^{n} \left| \frac{x_i - y_i}{max_i - min_i} \right|$$

Experiment č.1 skúmal vplyv prekryvových atribútov na identifikáciu parafrázovania. Z výsledkov vyplýva, že prekryv (skip-gramy a LCS) je vhodný atribút, pretože má nízku výpočtovú zložitosť a spolu s SVM zvýšil presnosť identifikácie parafrázovania. Lexikálna podobnosť slov znížila presnosť identifikácie parafrázovania.

Experiment č.2 zahrnul aj kombinácie lexikálnych a sémantických informácií. Pri SVM došlo k zlepšeniu presnosti iba o jedno percento.

Experiment č.3 predstavoval spojenie troch metód strojového učenia: SVM, k–najbližších susedov a maximálnej entropii. Vstupom bolo pre ne prekryv viet aj lexikálne a sémantické informácie, výsledok identifikácie parafrázovania bol určený hlasovaním. Preto sa v tabuľkách nižšie (Tabuľka 4.1 a Tabuľka 4.2) pre experiment číslo 3 uvádza iba jedna hodnota. Takýmto spôsobom sa výrazne zvýšila presnosť.

Výsledky uvádzajúce presnosť a návratnosť identifikácie parafrázovania sú uvedené v nasledujúcich tabuľkách.

Tabuľka 4.1 - Tabuľka vyjadrujúca presnosť metód strojového učenia pri jednotlivých experimentoch

Č. experimentu	SVM	Maximálna entropia	K – najbližších susedov
1. – prekrytie slov	69.86%	68.29%	63.36%
1. –podobnosť slov	66.50%	66.49%	67.81%
2.	70.43%	66.44%	64.68%
3.	76.64%		

Tabuľka 4.2 - Tabuľka vyjadrujúca návratnosť metód strojového učenia pri jednotlivých experimentoch

Č. experimentu	SVM	Maximálna entropia	K – najbližších susedov
1. – prekrytie slov	70.66%	59.53%	71.58%
1. – podobnosť slov	66.49%	68.20%	66.43%
2.	74.12%	70.50%	71.13%
3.	68.76%		

SVM na identifikáciu parafrázovania sa používa aj pri strojovom preklade. V článku [Brockett, a iní, 2005] je tiež uvádzané, že SVM má pri identifikácii parafrázovania najlepšie výsledky. Testovacie dáta predstavovalo 10 000 viet, ktoré boli ručne kategorizované.

Vstup do SVM predstavovali samotné vety a:

- reťazcová podobnosť,
- počet zhodných slov,
- levenshteinova vzdialenosť medzi slovami,
- stemy slov,
- označenie sémanticky podobných slov (synoným).

Pre identifikáciu parafrázovania bol takisto použitý lineárny kernel ($constant = 10^{-3}$, *complexity = 0,5*). Výsledky ukázali, že prekrytie viet a označovanie sémanticky podobných slov má výrazný vplyv na presnosť identifikácie parafrázovania.

V článku [Chitra, 2010] sa autor zaoberá použitím SVM a vplyvom lexikálnych a sémantických informácií na presnosť identifikácie parafrázovania. Na trénovanie aj testovanie bol použitý anotovaný Microsoft korpus.

Vstupom do SVM boli dvojice viet a k nim nasledujúce lexikálne a sémantické vlastnosti:

- Prekryv viet určený pomocou skip-gramou. Pre vety S_1 a S_2 sa určí hodnota prekryvu ako:

$$overlapS_1 = \frac{skip\_gram(S_1, S_2)}{\left(\frac{n}{skip\_gram(S_1, S_2)}\right)}$$

$$overlapS_2 = \frac{skip\_gram(S_1, S_2)}{\left(\frac{m}{skip\_gram(S_1, S_2)}\right)}$$

  kde n je počet slov vo vete S_1, m je počet slov vo vete S_2, funkcia$skip_gram(S_1, S_2)$ vyjadruje počet prekrytých skip-gramou vo vetách.
- Prekryv, ktorý je určený pomocou najdlhšej spoločnej podpostupnosti (LCS) sa vyčísli ako pomer počtu slov nájdených algoritmom LCS k počtu slov vo vete S_1 alebo S_2.
- Sémantická podobnosť slovies a podstatných mien vo vetách. Pre dvojicu slov w_1 a w_2, jedno slovo je sloveso a druhé podstatné meno, sa určuje sémantická podobnosť:

$$simm_{Lin}(w_1, w_2) = 2 \times \frac{hierarchy(w_1, w_2)}{ic(w_1) + ic(w_2)}$$

$$ic(w) = \log P(w)$$

  kde funkcia $ic(w)$ určuje informačný obsah slova w, $P(w)$ je pravdepodobnosť výskytu slova získaná zo systému WordNet. Funkcia $hierarchy(w_1, w_2)$ hovorí o sémantickej hierarchii medzi slovamiw_1 a w_2. Pre dvojicu viet sa celková podobnosť vypočíta ako priemer podobností $simm_{Lin}(w_i, w_j)$ medzi všetkými pármi podstatné meno – sloveso z oboch viet.
- Podobnosť číselných atribútov vo vetách. Z viet sa extrahujú číselné atribúty vyjadrené číslovkami aj číslicami a následne sa porovnávajú ich hodnoty. Ak sa hodnoty číselných atribútov považujú za sémanticky zhodné, dvojici viet sa priradí atribút s hodnotou 1, inak 0. Napríklad slovo „*tridsať*“ sa považuje za sémanticky zhodné s číslicou „*30*“, ale takisto aj s frázou „*väčší ako 29*“.

- Prítomnosť vlastných podstatných mien. Ak veta obsahuje názov, tak sa jej priradí atribút 1, inak 0.

Pri experimentoch bola použitá lineárna a Gaussova kernelová funkcia. Autor konštatuje, že pridanie lexikálnych a sémantických informácii zvýšilo presnosť identifikácie parafrázovania. Výsledná presnosť pri uvedených lexikálnych informáciách dosahovala až 70%.

4.4 Vyhodnotenie metód identifikácie parafrázovania

V tejto podkapitole si zhrnieme opísané prístupy k identifikácii parafrázovania. Všetky uvedené metódy boli vytvorené pre anglický jazyk a textované na Microsoft korpuse. Tento korpus bol zvolený kvôli svojej veľkosti a kvôli tomu, že má k sebe priradené vety, ktoré sú navzájom parafrázy.

Všetky metódy využívali lexikálny alebo sémantický prekryv. Lexikálny prekryv bol nájdený pomocou algoritmu LCS, skip-gramov na úrovni slov. Sémantický prekryv bol určený pomocou WordNet-u alebo značením slov, ktoré sú synonymá. Určenie prekryvu výrazne zlepší výsledky identifikácie parafrázovania pri metódach strojového učenia.

Určovanie lexikálnej vzdialenosti medzi slovami (napr. pomocou levenshteinovej vzdialenosti) presnosť identifikácie parafrázovania dokonca zhoršilo. Podľa nás je to tým, že aj lexikálne blízke slová môžu mať diametrálne rozdielny význam.

Určovanie morfologických kategórií v prípade [Kozerva, 2006] zvýšilo presnosť identifikácie parafrázovania pre angličtinu iba o jedno percento. Sme toho názoru, že pre slovenčinu môže určenie morfologických kategórii výraznejšie zlepšiť výsledky identifikácie parafrázovania, pretože slovenčina je flexívny a morfologicky bohatý jazyk. Vychádzame z toho, že ak sú vety parafrázované, mali by mať veľa spoločných syntaktických vzťahov a málo rozdielnych slov.

Všetky opísané prístupy mali presnosť identifikácie parafrázovania okolo 70%. Na presnosť má výrazný vplyv predspracovanie. Spomedzi algoritmov strojového učenia sú najviac používané algoritmy podporných vektorov (SVM), ktoré sú na tento účel vhodné, vďaka svojim vlastnostiam.

Myslíme si, že identifikácia parafrázovania pomocou váhovania (kapitola 4.2) nie je vhodná pre všeobecné použitie. Táto metóda používa iba jeden prahový atribút, ktorý je špecifický pre „trénovací“ korpus a pri identifikácií parafráz v textoch z inej domény klesá úspešnosť. Na tento fakt upozorňuje aj samotný autor opísanej metódy [Lintean, 2009] a aj [Chitra, 2010]. Preto považujeme použitie algoritmov strojového učenia a najmä SVM za vhodnejšie.

5 Špecifikácia požiadaviek

Hlavnou požiadavkou na našu aplikáciu je identifikácia parafrázovania na vyššej úrovni v slovenskom jazyku. Z analýzy a nutnosti experimentovania so samotnou metódou vyplynuli nasledujúce požiadavky.

Požiadavky na metódu sme určili nasledovne:

- Metóda má identifikovať aspoň parafrázovanie na vyššej úrovni, ktoré sme si definovali ako kombináciu:
  - zmeny slovosledu,
  - vkladaním alebo odstraňovaním slov z vety,
  - použitím synoným.
- Na predspracovanie viet budú použité existujúce prístupy založené na skrytých Markovovských modeloch, LCS a metóde n-gramy.
- Pri identifikácii parafrázovania bude použitý prístup využívajúci algoritmy podporných vektorov (SVM).

Požiadavky na aplikáciu sme učili nasledovne:

- Aplikácia bude vedieť spracovať dokumenty vo formátoch:
  - čistý text (*.txt*),
  - dokumenty Microsoft Office Word *(.doc, .docx*),
  - opendocument (*.odt*),
  - rich text format (*.rtf*)
  - formát PDF (*.pdf*).
- Aplikácia bude umožňovať pridávať dokumenty do zvoleného korpusu a odoberať dokumenty zo zvoleného korpusu.
- Uchovávanie metadát – dokumenty budú spolu s ich metadátami a predspracovanými vetami uložené v databáze.
- Konfigurovateľnosť aplikácie – v aplikácii bude možné nakonfigurovať:
  - postupnosť krokov predspracovania viet a ich parametre,
  - kernel funkciu,
  - parametre trénovania algoritmov podporných vektorov.
- Export výsledkov – aplikácia bude exportovať výsledky identifikácie parafrázovania do formátu CSV.

6 Návrh identifikácie parafrázovania na vyššej úrovni

Náš návrh riešenia pre identifikáciu parafrázovania na vyššej úrovni vychádza z analýzy existujúcich prístupov k identifikácii parafrázovania a výsledkov týchto prístupov. Keďže sa zaoberáme len slovenským jazykom, musíme pri tomto návrhu zohľadniť špecifiká tohto jazyka a súčasné možnosti jeho predspracovania.

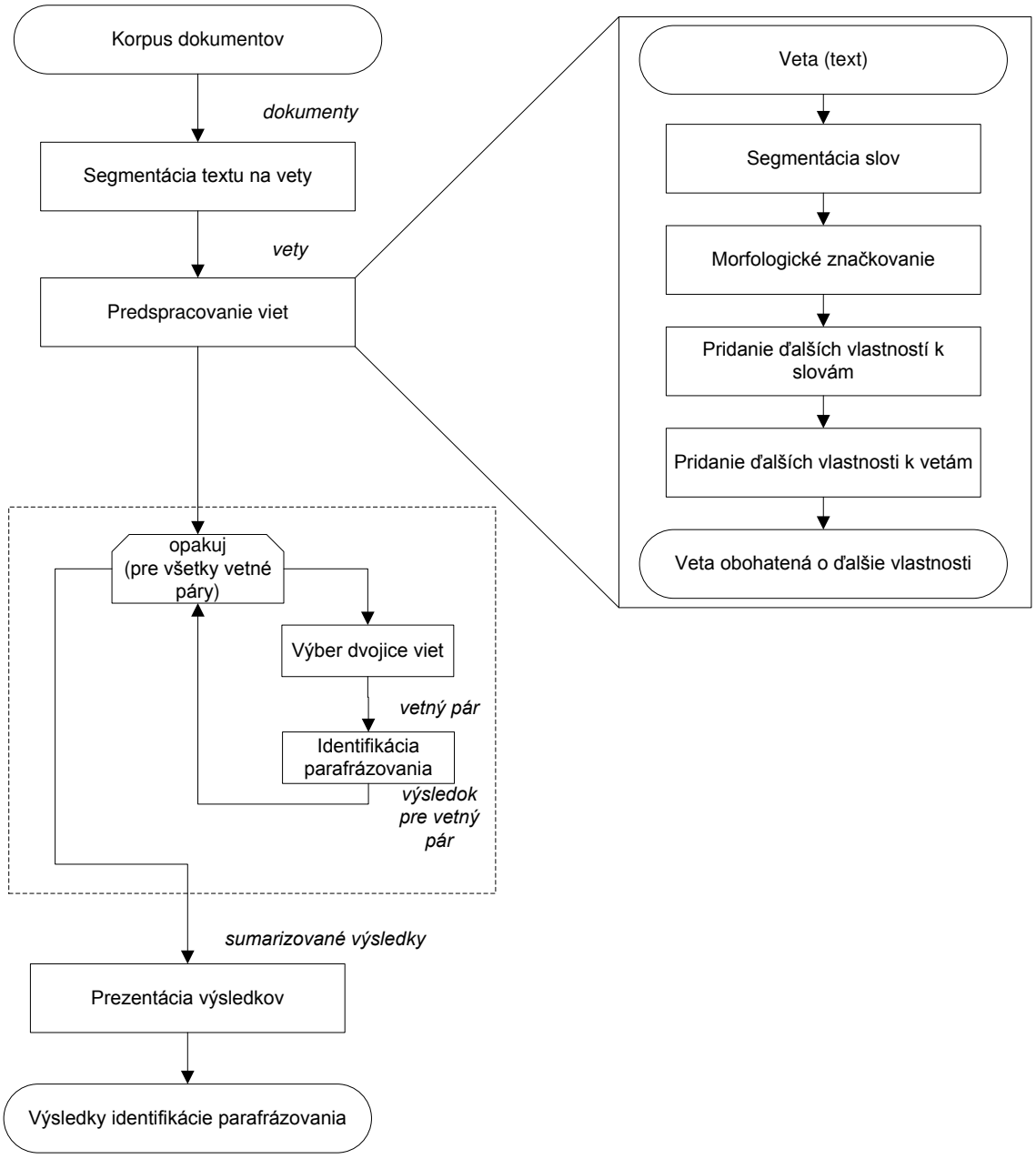
Pri návrhu výsledného riešenia bolo taktiež potrebné zohľadniť aj vlastnosti, ktoré priamo nesúvisia s hľadaním parafrázovania v texte. Výsledný systém by totiž mal byť použitý na realizovanie experimentov s rôznymi druhmi predspracovania textov na lexikálnej, syntaktickej a sémantickej úrovni, takisto aj s parametrami pre algoritmy podporných vektorov (SVM). Preto by výsledný systém mal byť v čo možno najväčšej miere flexibilný z tohto hľadiska.

Navrhovaný systém bude spracovávať dokumenty nasledujúcim postupom (Obrázok 6.1):

1. *Načítavanie dokumentov* – zo súborov dokumentov, ktoré môžu byť v rôznom formáte (*.docx, .pdf, .rtf, .txt, .odt*), sa získa čistý text. V tejto fáze treba previesť texty do jednotného kódovania.
2. *Segmentácia textu na vety* – text získaný z dokumentu sa rozdelí na vety. Následne sa tie úseky textu, ktoré nepokladáme za vety, odstránia (napríklad sú tieto úseky príliš krátke). Príklad takýchto úsekov sú nadpisy a popisy obrázkov.
3. *Predspracovanie viet* – v tomto kroku sa vety predspracujú. Tento postup predspracovania je podrobne opísaný nižšie.
4. *Výber dvojice viet* – v tomto kroku sa vyberajú dvojice viet zo všetkých dokumentov, ktoré vstupujú do procesu identifikácie parafrázovania.
5. *Identifikácia parafrázovania* – v tomto kroku sa identifikuje parafrázovanie nad dvojicou viet. Podrobnejšie bude tento bod vysvetlený neskôr.
6. *Prezentácia a export výsledkov* – v tomto kroku sa zobrazia výsledky používateľovi alebo vyexportujú pre účely ich spracovania iným systémom.

Predspracovanie viet pre identifikáciu parafrázovania pozostáva z nasledujúcich krokov:

1. *Segmentácia slov* – vety sa rozdelia na jednotlivé slová.
2. *Morfologické značkovanie* – slovám vo vetách sa priradia morfologické značky. Pre samotné prideľovanie morfologických značiek budú použité skryté Markovovské modely a Viterbiho algoritmus (kapitoly 3.4 a 3.6). Skryté Markovovské modely obohatíme o tri symboly výstupnej abecedy – o bodku, otáznik a výkričník na konci vety, pretože určujú postojovú modálnosť vety a tým aj čiastočne jej skladbu.
3. *Priradenie vlastností k slovám* – pre každé slovo určíme jeho vlastnosti, napríklad lemu slova, entropiu a podobne.
4. *Priradenie vlastností k vetám* – pre vetu sa určia ďalšie vlastnosti, napríklad valenčné polia, či obsahuje vlastné podstatné mená, modálnosť vety a podobne. Tieto atribúty a atribúty získané v predchádzajúcom kroku budú slúžiť pri experimentovaní.

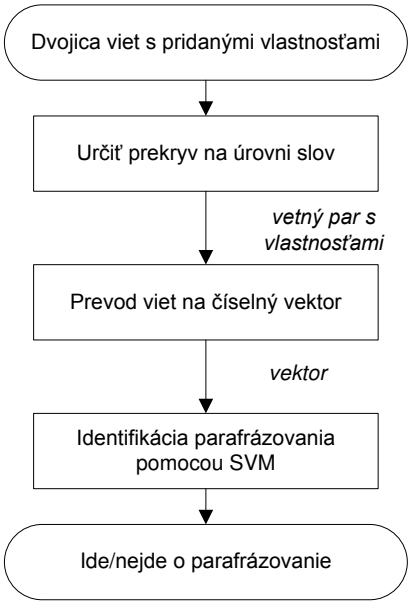


Obrázok 6.1 - Navrhovaný postup spracovania dokumentov a predspracovania viet za účelom identifikácia parafrázovania na vyššej úrovni

Identifikácia parafrázovania pre dvojicu predspracovaných viet bude prebiehať v nasledujúcich krokoch (Obrázok 6.2):

1. *Určí sa prekryv medzi každými dvoma vetami na úrovni slov* – zistí sa, ktoré slová sa vo vetách prekrývajú lexikálne alebo syntakticky. Prekryv sa určí pomocou algoritmu LCS a skip-gramov.
2. *Reprezentácia viet vektormi* – vety aj s ich vlastnosťami budú reprezentované vektorom obsahujúcim reálne čísla, kde každá hodnota vektoru reprezentuje inú vlastnosť. Navrhli sme niekoľko alternatív konverzie dvojice viet na vektor, ktoré sú bližšie opísané v kapitole 6.1.

3. *Identifikácia parafrázovania prostredníctvom SVM* – použitím SVM sa vyhodnotí parafrázovanie medzi dvoma vetami. Vstupom pre SVM je číselný vektor vytvorený v predchádzajúcom kroku a výstupom je booleovská hodnota, či je jedna veta parafrázou tej druhej.



Obrázok 6.2 - Identifikácia parafrázovania pre dvojicu viet

6.1 Konverzia na vektor

Navrhli sme niekoľko alternatív konverzie dvojice viet na vektor, ktorý by bol vstupom pre SVM:

- Základná reprezentácia,
- Obohatený vektorový model dokumentu,
- Jednoduchá reprezentácia.

V nasledujúcich častiach tieto reprezentácia bližšie opíšeme.

Základná reprezentácia

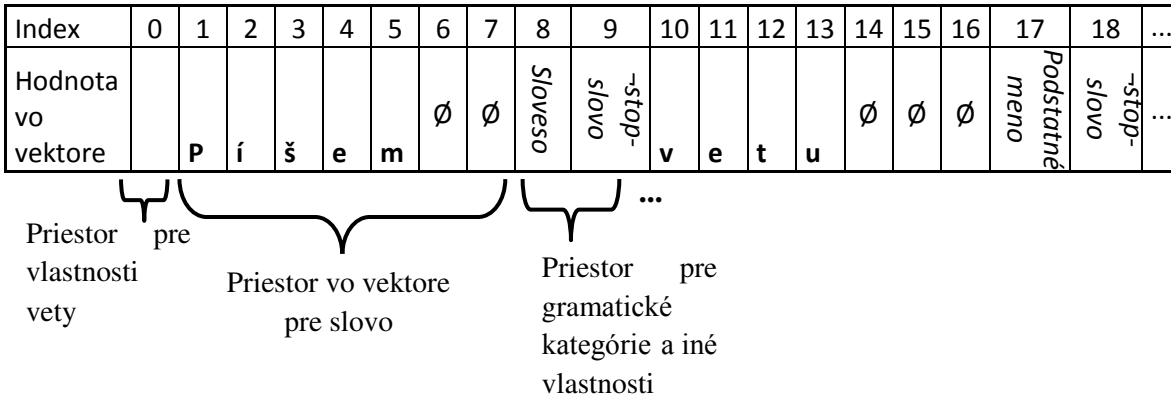
Tento spôsob reprezentácie dvojice viet bol použitý v pôvodných zdrojoch [Chitra, 2010],[Kozerva, 2006] zaoberajúcich sa identifikáciou parafrázovania v anglickom jazyku. Blokové znázornenie sa nachádza na obrázku nižšie (Obrázok 6.3).

Prekryvové vlastnosti	Vlastnosti 1. vety	Slová 1. vety s ich vlastnosťami	Vlastnosti 2. vety	Slová 2. vety s ich vlastnosťami

Obrázok 6.3 - Príklad logického usporiadanie hodnôt v číselnom vektore pre SVM

V prvej časti vektora sa nachádzajú prekryvové vlastnosti a zvyšná časť sa rozdelí na dve rovnaké polovice .Do prvej polovice sa vloží prvá veta, do druhej sa vloží druhá veta. Jednotlivé slová sa spolu so svojimi vlastnosťami vkladajú po znakoch za sebou tak, ako sú v pôvodných vetách. Pre každé slovo je vyhradený pevný počet položiek vo vektore. Ak má

slovo menej písmen, ako je týchto položiek, zvyšné sa vyplnia nulami, ak má slovo viac písmen, ako je týchto položiek, zvyšné sa zahodia. Podobne, ak má veta menej slov, ako má vyhradené vo vektore, tak sa zvyšným položkám vektora priradí nula. Výstupom tohto kroku je vždy rovnako dlhý vektor, ktorého ukážka pre vetu začínajúcu na „*Píšem vetu ...*“, je znázornená na obrázku nižšie (Obrázok 6.4).



Obrázok 6.4 - Príklad logického usporiadanie hodnôt v číselnom vektore pre SVM

Obohatený vektorový model dokumentu

Tento spôsob reprezentácie dvojice viet vychádza z vektorového modelu dokumentu VSM (*angl. Vector space model*). Samotná vektorová reprezentácia sa skladá zo štyroch častí: prekryvové vlastnosti, vlastnosti pre prvú a druhú vetu a agregovaná *tf-idf* reprezentácia viet. Je ho možné vidieť na obrázku nižšie (Obrázok 6.5).

Prekryvové vlastnosti	Vlastnosti 1. vety	Vlastnosti 2. vety	Agregovaná tf-idf reprezentácia

Obrázok 6.5 - Obohatený vektorový model dokumentu

Druhá časť vektoru vznikne spojením *tf-idf* dvoch vektorov, ktoré vznikli z jednotlivých viet, pomocou agregačnej funkcie.

$$v_i = \omega\left(tf_{1,i}, tf_{2,i}\right).idf_i$$

kde v_i je hodnota agregovaného *tf-idf* pre *i*-ty term, ω je agregačná funkcia, $tf_{1,i}$ je frekvencia *i*-teho termu patriaceho prvej vete a idf_i je inverzná frekvencia termu patriaca *i*-tému termu. My sme navrhli nasledujúce agregačné funkcie, ktoré vyjadrujú hodnotu prekryvu *i*-teho termu vo vetách vetného páru (ak sa term nenachádza ani v jednej vete, výsledok funkcie je 0, ak sa nachádza aspoň v jednej, výsledkom funkcie je malá hodnota, ak sa term nachádza v oboch, výsledkom funkcie je vysoká hodnota).

$$\omega\left(tf_{1,i}, tf_{2,i}\right) = \beta.\min\left(tf_{1,i}, tf_{2,i}\right) + (1-\beta).\max\left(tf_{1,i}, tf_{2,i}\right)$$

$$\omega(tf_{1,i}, tf_{2,i}) = \begin{cases} 1 & ak\ tf_{1,i} > 0 \wedge tf_{2,i} > 0 \\ 0{,}3 & ak\ tf_{1,i} > 0 \vee tf_{2,i} > 0 \\ 0 & inak \end{cases}$$

Jednoduchá reprezentácia

Tento spôsob reprezentácie dvojice viet sa skladá iba z prekryvových vlastností a vlastností určených pre obe vety. Je ho možné vidieť na obrázku nižšie (Obrázok 6.6).

Prekryvové vlastnosti	Vlastnosti 1. vety	Vlastnosti 2. vety

Obrázok 6.6 - Jednoduchá reprezentácia

6.2 Informačný prekryv

Navrhli sme niekoľko filtrov zohľadňujúcich informačný prekryv pri výpočte prekryvu viet. Snažíme sa priradiť vyššiu váhu slovám, ktoré nesú väčšiu časť informačnej hodnoty vety ako iné. Následne túto hodnotu zohľadňujeme pri výpočte LCS:

$$c[i,j]_{s1,s2} = \begin{cases} 0, & ak\ i = 0 \vee j = 0 \\ c_{s1,s2}[i-1, j-1] + w(s1_i), & ak\ i,j > 0 \wedge s1_i = s2_j \\ \max\{c[i-1,j]_{s1,s2}, c[i,j-1]_{s1,s2}\}, & inak \end{cases}$$

kde $s1$ a $s2$ sú porovnávané vety, c je pomocné dvojrozmerné pole a funkcia $w(s1_i)$ určí informačnú hodnotu (váhu) slova $s1_i$. Po výpočte LCS sa hodnota ešte normalizuje:

$$infOverlap(s1, s2) = \frac{c[lenght(s1), lenght(s2)]_{s1,s2}}{\min\{\sum_{s \in s1} w(s), \sum_{s \in s2} w(s)\}}$$

Podobne postupujeme pri výpočte informačného prekryvu pomocou n-gramov (vychádza z Jaccardovej podobnosti) a skip-gramov

$$infOverlap(s1, s2) = \frac{infGram(ngram(s1) \cap ngram(s2))}{infGram(ngram(s1) \cup ngram(s2))}$$

$$infGram(Q) = \sum_{q \in Q} \sum_{i=1}^{n} w(q_i)$$

kde funkcia $ngram(s1)$ vráti množinu n-gramov alebo skip-gramov z vety $s1$, funkcia $infGram(Q)$ počíta informačnú hodnotu množiny n-gramov na úrovni slov, funkcia $w(q_i)$ vráti informačnú hodnotu pre slovo q_i.

Informačnú hodnotu (váhu) $w(t)$ pre jednotlivé slová počítame rôznymi spôsobmi:

- *Inverzná frekvencia termov* (slov) – vyjadruje dôležitosť slova v rámci celého korpusu dokumentov S, informačnú hodnotu pomocou *idf* teda vypočítame ako:

$$w(t) = log \frac{N}{|\{s \in S | t \in s\}|}$$

kde N je počet dokumentov v korpuse S.

- *Inverzná frekvencia termov nad vetami* – na rozdiel od predchádzajúcej hodnoty váhy považujeme každú vetu v korpuse za samostatný dokument. Na rozdiel od klasického výpočtu *idf* neberieme do úvahy celé dokumenty, ale jednotlivé vety, pretože naša metóda identifikácie parafrázovania pracuje nad vetami. (Za normálnych okolností sa *idf* počíta nad korpusom dokumentov [Parlič, 2010], ale pre potreby identifikácie parafrázovania sme zvolili aj granularitu viet.)
- *Entropické váhovanie* – používa sa podobne ako *idf* pri latentnej sémantickej indexácii [Nemirovsky, a iní, 2008], samotnú váhu počítame ako:

$$w(t) = 1 - \sum_{j=1}^{N} \frac{p_{t,j} \log p_{t,j}}{\log N}$$

kde $p_{t,j}$ je pravdepodobnosť výskytu slova t v dokumente j. Túto pravdepodobnosť počítame podobne ako pri *idf*, je to pomer početnosti slova t v dokumente i k celkovému počtu výskytov slova t v korpuse:

$$p_{t,j} = \frac{\left|\{s \in s_j | s = t\}\right|}{\sum_{i=1}^{N} |\{s \in s_i | s = t\}|}$$

- *Entropické váhovanie nad vetami* – aj v tomto prípade považujeme každú vetu v korpuse za samostatný dokument. Pravdepodobnosť $p_{t,j}$ počítame ako v predchádzajúcom prípade, ale doplnili sme pre ňu ďalší vzťah. Ide o pravdepodobnosť výskytu slova *t* vo vete.

$$p_{t,j} = \begin{cases} \frac{1}{lenght(s_j)}, & ak\ s_j \in t \\ 0, & inak \end{cases}$$

Tento vzorec vyjadruje pravdepodobnosť, že náhodne zvolené slovo z vety s_j je slovo *t*.

- *Váhy pre slovné druhy* – vychádzame z myšlienky, že rôzne slovné druhy prinášajú do vety rôzne vysokú informačnú hodnotu. Slovesá a podstatné mená sú nositeľmi deja vety (podmet, prísudok, predmet), preto dostali najvyššiu váhu ako napríklad častice a citoslovcia. Hodnoty váh sme určili odhadom, na základe kapitoly 2 Štruktúra slovenského jazyka.

7 Implementácia

V tejto kapitole opisujeme výber programovacieho jazyka, rámcov a knižníc, ktoré boli použité pri implementácii prototypu a podrobnejšie opisujeme fungovanie jednotlivých častí prototypu.

Pri výbere programovacieho jazyka a platformy sme sa rozhodovali podľa niekoľkých kritérií. Na základe typu licencie, pod ktorou nám bol poskytnutý korpus *r-mark*, sme sa rozhodli pre klient-server architektúru s tenkým klientom. Zvolili sme komunikáciu pomocou webových služieb, pretože ide o štandardizované a robustné riešenie. Samotné riešenie sme sa rozhodli implementovať na platforme .NET 4.0 v jazyku C#, pretože poskytuje rámec na prácu s webovými službami (WCF rámec). Taktiež v tomto jazyku existujú knižnice na spracovanie textových dokumentov, ako aj knižnice implementujúce SVM. S .NET platformou máme celkovo aj bohatšie skúsenosti ako s inými platformami, napr. Java, ktorá tiež poskytuje obdobný ekosystém knižníc a rámcov ako .NET.

7.1 Prototyp

Pre overenie nami navrhovanej metódy identifikácie parafrázovania na vyššej úrovni a zistenia vplyvu predspracovania na presnosť metódy bolo nutné vytvoriť prototyp, s ktorým bude možné vykonávať experimenty nad našimi dátovými vzorkami. Tento prototyp bol rozdelený na niekoľko častí z pohľadu zabezpečovanej funkcionality. Časti, ktoré sú zaujímavé z pohľadu identifikácie parafrázovania, rozoberáme v nasledujúcich podkapitolách.

7.1.1 Predspracovanie dokumentov

Knižnica pre predspracovanie dokumentov implementuje konverziu rôznych typov dokumentov do spoločného interného formátu. Tento formát obsahuje vlastný text dokumentu a jeho metadáta:

- titulok,
- autor dokumentu,
- autor, ktorý vykonal poslednú zmenu,
- dátum a čas vytvorenia dokumentu,
- dátum a čas poslednej zmeny dokumentu.

Konverziana čistý text je podporovaná pre formáty *pdf*, *doc*, *docx*, *rtf, odt* a súbory s čistým textom. Diagram znázorňujúci časť knižnice pre konverziu dokumentov sa nachádza v prílohe (Obrázok P.3). Okrem tohto naša knižnica pre predspracovanie obsahuje funkcionalitu pre rýchle vyhľadávanie v texte ako KMP a AHO Corashickej algoritmus a triedu pre kúskovanie textu (*text chunking*), ktorá delí text na vety a vety na lexikálne jednotky (v našom prípade slová, čísla a interpunkcia). Súčasťou tejto knižnice sú aj triedy pre prácu s *tf-idf* (príloha Obrázok P.4).

7.1.2 Knižnica pre morfologické značkovanie

Knižnica pre morfologické značkovanie priraďuje slovám z vety ich slovný druh. Využíva skryté Markovove modely a Viterbiho algoritmus opísané v kapitole 3.6. Určujeme ňou desať slovných druhov (kapitola 2.1.2), príčastie (*participum*), reflexívum (*sa, si*), kondiciánolovú

mofrému (*by*), bodku, čiarku, otáznik, výkričník a iné (zlučuje všetky ostatné lexikálne jednotky ako zátvorky, citácie, ...).

Knižnica pozostáva z dvoch častí. Prvá slúži na vytvorenie prechodovej matice (v popise algoritmu matica A), množiny pravdepodobností počiatočných stavov (π) a lexikónu (slúžiaceho pre funkciu nahradzujúcu maticu B) z korpusu *r-mark* a ich uloženie do XML súboru.

Druhá časť priraďuje slovné druhy slovám vo vete pomocou Viterbiho algoritmu. Oproti pôvodnému riešeniu sme vo Viterbiho algoritme nahradili maticu B funkciou *tagWordPropability*, využívajúcej lexikón, vďaka ktorej sa nám podarilo zvýšiť presnosť určovania slovných druhov.

$$tagWordPropability(w,t) = \begin{cases} M_1[w,t], & ak\ \exists M_1[w,t] \\ M_2[suffix(w),t], & ak\ \exists M_2[suffix(w),t] \\ P(t), & inak \end{cases}$$

Funkcia pre slovo w a morfologickú značku (slovný druh) t počíta pravdepodobnosť, že slovo patrí do slovného druhu. M_1 je matica obsahujúca pravdepodobnosti, že slovo w patrí slovnému druhu t. M_2 je matica obsahujúca pravdepodobnosti, že ľubovoľné slovo s príponou $suffix(w)$ patrí slovnému druhu t. Funkcia $suffix(w)$ vracia príponu - posledné 3 písmená zo slova w a $P(t)$ vracia pravdepodobnosť výskytu slovného druhu t.

Matice M_1, M_2 a funkcia $P(t)$ sú súčasťou vytvoreného lexikónu. Matica M_1 nemôže pokryť pravdepodobnosti ku všetkým slovám v slovenčine, pretože ich neobsahuje korpus *r-mark*. Preto bola oproti pôvodnému návrhu pridaná matica M_2. Experimentálne sme určili najvhodnejšiu dĺžku použitých prípon na 3 písmená.

7.1.3 Dátový model

Pre dátovú vrstvu sme zvolili ORM rámec *Entity Framework*, diagram entít je na obrázku v prílohách (Obrázok P.9). Ako databázu sme použili *MS SQL Server 2008 R2*.

7.1.4 Identifikácia parafrázovania

Implementovaný proces identifikácie parafrázovania je zhodný s návrhom uvedeným v kapitole 6. Na pridávanie vlastností k vetám, slovám a dvojiciam viet sa používajú filtre. Filter môže meniť text a pridávať vlastnosti. Použili sme štyri druhy filtrov:

- *filter pre dokument* – nepridávajú vlastnosti dokumentom, ale menia text dokumentu (napr. filter pre odstraňovanie textu v zátvorkách),
- *filter pre vetu* – môže upraviť text vety a pridáva k nej vlastnosti, filter pre vetu sa môže aplikovať na príslušnú vetu:
  - pred delením textu vety na slová (napr. určenie modálnosti),
  - po delení na slová (napr. určenie relatívnej dĺžky vety),
- *filter pre slovo* – môže meniť slovo, ale aj pridávať vlastnosti k slovám (napr. označenie stop-slov, priradenie lemy slova),

- *filter pre vetný pár* – aplikuje sa až tesne pred konverziu vetného páru na číselný vektor, môže pridávať vlastné vlastnosti, takisto aj slovám a vetám (napr. určenie prekryvu pomocou LCS, 3-gramou).

Vlastnosť je dodatočná informácia k jazykovej jednotke, ktorá má svoj typ, názov a hodnotu. Každý druh filtra vytvára iné druhy vlastností. Zoznam vytvorených filtrov je v prílohe (Tabuľka P.2). Diagram tried pre filtre a vlastnosti je uvedený v prílohe (Obrázok P.6).

Implementovaná konverzia páru viet bola implementovaná podľa všetkých návrhov z kapitoly 6.1, ako osobitné triedy (diagram tried v prílohe Obrázok P.7). Na vyhodnotenie parafrázovania sme využili SVM z knižnice *Accord.NET*[2], táto knižnica obsahuje okrem možnosti trénovania a použitia SVM aj množstvo kernel funkcií a iných algoritmov strojového učenia.

7.1.5 Testovanie

Všetky dôležité implementácie algoritmov sú pokryté unit testami. Ako napríklad kúskovanie textu, *tf-idf*, skryté Markovove modely, n-gramy, LCS,... Testami sú pokryté aj triedy pre validáciu výsledkov. Použili sme možnosť vytvárať unit testy pomocou *Visual Studia*.

Ako súčasť implementácie boli vytvorené menšie aplikácie slúžiace na testovanie.

[2]Knižnica *Accord.NET* je dostupná na adrese https://code.google.com/p/accord/ .

8 Overenie riešenia

V tejto kapitole uvádzame, akým spôsobom sme overili a vyhodnotili identifikáciu parafrázovania na vyššej úrovni použitím nášho riešenia. Overenie a vyhodnotenie vykonávame nad dvoma časťami prototypu:

- nad knižnicou pre morfologické značkovanie, ktorá využíva skryté Markovovské modely,
- nad časťou pre identifikáciu parafrázovania, ktorá využíva SVM.

Obe tieto časti vyhodnocujeme pomocou krížovej validácie.

8.1 Dátová vzorka

Dátovú vzorku pre knižnicu na určovanie slovných druhov tvorí *Ručne morfologicky anotovaný korpus* (*r-mark)*, ktorý bol sprístupnený *Slovenskou akadémiou vied*. V tomto korpuse sú ku každej lexikálnej jednotke určené jej gramatické kategórie. Obsahuje 77 755 viet tvorených 1 199 224 lexikálnymi jednotkami.

Navrhovanú metódu identifikácie parafrázovania textových dokumentov je pre vyčíslenie presnosti nutné overovať nad dvojicami viet, u ktorým vieme, či sa z pohľadu človeka jedná o parafrázu. Podarilo sa nám vytvoriť korpus 1 947 vetných párov, ktorý obsahuje polovicu parafrázovaných vetných párov. Tento korpus vznikol parafrázovaním beletrie alebo zdrojov dostupných na internete. Po vytvorení bol skontrolovaný a preto obsahuje minimum gramatických chýb. Nazvali sme ho Korpus1947. Pre lepšiu predstavu o korpuse sme určili podobnosť parafrázovaných aj neparafrázovaných vetných párov pomocou 3-gramov a Jaccardovej podobnosti. Priemerná podobnosť parafrázovaných vetných párov je 8,55% a neparafrázovaných 0,90%.

Pre účely hľadania optimálneho nastavenia parametrov našej metódy pre identifikáciu parafrázovania sme z dátovej vzorky Korpus1947 náhodne vybrali 600 vetných párov. Túto skrátenú dátovú vzorku sme nazvali Korpus600.

Ako posledný sme vytvorili Korpus202, ktorý obsahuje 202 vetných párov. Na rozdiel od dátovej vzorky Korpus1947, každý vetný pár, ktorý neobsahoval parafrázy, bol vytvorený tak, aby neparafrázovaná veta obsahovala spoločné slová alebo sa týkal podobnej témy ako pôvodná veta. Jaccardova podobnosť 3-gramov pre Korpus202 je 7,42% pre parafrázované páry a 2,62% pre neparafrázované páry.

8.2 Spôsob vyhodnocovania výsledkov

Pri experimentoch sme vyhodnocovali výsledky identifikácie parafrázovania vo vetách a aj priraďovanie morfologických značiek (slovných druhov). Metódu identifikácie parafrázovania aj morfologické značkovanie vyhodnocujeme pomocou 3-krížovej validácie (*n-cross fold validation*). To znamená, že dátová vzorka sa rozdelí na 3 podmnožiny, pričom jedna slúži ako testovacia vzorka, ostatné podmnožiny slúžia ako trénovacia vzorka. Proces trénovania a validácie sa 3-krát opakuje, vždy s inou testovacou podmnožinou dát.

Pri identifikácii parafrázovania porovnávame, či naša metóda pre daný vetný pár na vstupe určila správne jeho zaradenie - či je parafrázovaný alebo nie.

Pri prideľovaní morfologických značiek priraďujeme lexikálnej jednotke jednu z 18-tich morfologických značiek a preto nás tu zaujíma celková presnosť, správnosť a návratnosť pre jednotlivé morfologické značky. Ako dátová vzorka slúži *r-mark*.

Metriky presnosť a úplnosť vychádzajú z konfiguračnej tabuľky pre triedu *c*:

Tabuľka 8.1 - Konfiguračná tabuľka

		Bola klasifikátorom predikovaná trieda *c*?	
		Áno	Nie
Je skutočná trieda klasifikovaného objektu *c*?	Áno	tp	fp
	Nie	tn	fn

Presnosť (*precision*) je definovaná ako podiel počtu objektov klasifikovaných správne do triedy *c* a počet objektov klasifikovaných do triedy *c*.

$$precision = \frac{tp}{tp + tn}$$

Návratnosť (*recall*) je definovaná ako podiel počtu správne klasifikovaných objektov do triedy *c* a celkového počtu objektov, ktoré patria do triedy *c*.

$$recall = \frac{tp}{tp + fp}$$

Správnosť (*accuracy*) je definovaná ako podiel správne klasifikovaných objektov patriacich aj nepatriacich do triedy *c* ku všetkým objektom.

$$accuracy = \frac{tp + fn}{tp + fp + tn + fn}$$

8.3 Dosiahnuté výsledky morfologického značkovania

Vyhodnotenie knižnice pre morfologické značkovanie založenej na skrytých Markovových modeloch prebiehalo pomocou vyššie uvedených metrík. Pomer lexikálnych jednotiek, ktorým bola určená správna morfologická značka ku všetkým je 95,35%. Výsledky presnosti a návratnosti sú uvedené v tabuľke nižšie (Tabuľka 8.2). Časť chybovosti pri výsledkoch, ktorá je najviac očividná pri interpunkčných znamienkach, je spôsobená chybami v samotnom korpuse.

Tabuľka 8.2 - Presnosť a návratnosť pre morfologické značky

Morfologická značka	Presnosť	Návratnosť
Podstatné meno	94,04%	97,66%
Prídavné meno	94,42%	92,00%
Zámeno	95,74%	98,41%
Číslovka	97,97%	91,16%
Sloveso	95,57%	97,84%
Príslovka	91,67%	85,57%
Spojka	92,39%	92,73%
Častica	82,95%	82,07%
Citoslovce	94,54%	56,05%
Príčastie	98,30%	54,04%
Reflexívum	98,34%	98,70%
Kondicionalová morféna	99,91%	99,88%
Bodka, Čiarka, Otáznik, Výkričník	99,81%	99,80%
Iné	85,15%	29,92%

8.4 Dosiahnuté výsledky v identifikácii parafrázovania

Pri identifikácii parafrázovania sme experimentovali s piatimi nastaveniami predspracovania, tromi typmi konverzie dvojice viet na vektor (kapitola 6.1) a šiestimi rôznymi kernelmi. Použili sme nasledujúce nastavenia predspracovania:

- *Základné* – vetám sa priradí ich modálnosť, slovám sa priradia morfologické kategórie a nahradia sa svojou lemou. Prekryv viet sa určuje nad lemami slov pomocou 3-gramov, 4-skip gramov a LCS.
- *Značkovanie* – vetám sa priradí ich modálnosť, slovám sa priradia morfologické kategórie a nahradia sa svojou lemou, a označí sa, či ide o stop-slovo. Prekryv viet sa určuje nad lemami slov pomocou 3-gramov, 4-skip gramov a LCS, ktoré označí zhodné slová.
- *Využitie synoným* - vetám sa priradí ich modálnosť, slovám sa priradia morfologické kategórie a nahradia sa svojou lemou, a či ide o stop-slovo. Prekryv viet sa určuje nad synonymami slov pomocou 3-gramov, 4-skip gramov a LCS.
- *Informačný prekryv* – je rozšírením nastavenia predspracovania pomenovaného *Základné* o filtre počítajúce informačný prekryv.
- *Kombinovaná metóda* – je to kombinácia využitia synoným spolu s informačným prekryvom počítaným vážením slovných druhov.
- *Replikácia experimentu* – tento spôsob predspracovania sme navrhli pre porovnanie našich výsledkov s experimentom z [Kozerva, 2006], ktorý je opísaný v kapitole 4.3.4.

8.4.1 Experimenty pre nastavenie parametrov

Ako prvé sme testovali šesť kernelových funkcií, aby sme zistili, ktoré sa pre identifikáciu parafrázovania nehodia. Najlepšie výsledky sme dosiahli s Gaussovým kernelom a lineárnym kernelom. Najhoršie výsledky sme dosiali so sférickým kernelom (presnosť iba 50%),

sigmodiálnym (presnosť okolo 65%) a polynomiálnym kernelom (presnosť sa pohybovala v rozmedzí od 53% do 85%, v závislosti od nastavenia kernelu).

V ďalšom experimente sme zisťovali najlepšie nastavenie parametrov, kombinácií spôsobu konverzie na vektor a nastavení predspracovania, pre identifikáciu parafrázovania. Najlepšie výsledky (v rámci hľadania nastavenia parametra *complexity*) pri rôznom nastavení predspracovania a konverzie sú uvedené v tabuľke nižšie (Tabuľka 8.3), tieto výsledky boli namerané na dátovej vzorke Korpus600. Najlepšie výsledky dosiahlo predspracovanie pomenované *Kombinovaná metóda* s *Obohateným vektorovým modelom dokumentu* a Gaussovým kernelom – 95,00%. Druhé najlepšie výsledky sme dosiahli pomocou predspracovania *Kombinovaná metóda* s *Jednoduchou reprezentáciou* a Gaussovým kernelom – 94,83%. Zopakovaný experiment z [Kozerva, 2006] (predspracovanie pomenované *Replikácia experimentu* so *Základnou* konverziou vektoru pre SVM) – 90,50% a to aj s lineárnym aj Gaussovým kernelom.

Tabuľka 8.3 - Najlepšie výsledky pre rôzne nastavenia parametrov

	Spôsob konverzie na vektor	**Základný**		**Jednoduchá reprezentácia**		**Obohatený vektorový model dokumentu**	
Nastavenie predspracovania		Gaussov kernel	Lineárny kernel	Gaussov kernel	Lineárny kernel	Gaussov kernel	Lineárny kernel
Základné	Complexity	2,5	0,2	2	2	1,5	0,4
	Správnosť	90,50%	89,17%	94,67%	94,33%	94,67%	94,33%
Značkovanie	Complexity	3,5	0,8	2	2,5	3	0,4
	Správnosť	91,33%	89,67%	94,67%	94,33%	94,83%	94,33%
Využitie synoným	Complexity	3,5	0,1	3	0,9	0,8	0,1
	Správnosť	89,67%	90,50%	94,67%	94,50%	94,83%	94,00%
Informačný prekryv	Complexity	2,5	0,1	3	2,5	3	0,7
	Správnosť	91,33%	89,17%	94,17%	94,67%	94,17%	94,33%
Kombinovaná metóda	Complexity	6,0	0,1	5,5	4,0	0,8	0,4
	Správnosť	90,33%	89,83%	94,83%	94,50%	95,00%	94,33%
Replikácia experimentu	Complexity	0,5	0,1	4,5	0,8	0,9	0,5
	Správnosť	90,50%	90,50%	94,50%	94,33%	94,17%	94,17%

8.4.2 Výkonnostné testy metód

Merali sme priemerný čas potrebný na spracovanie a vyhodnotenie vetného páru pomocou dátovej vzorky Korpus600. V nasledujúcej tabuľke (Tabuľka 8.4) je možné vidieť čas v milisekundách potrebných na spracovanie jedného vetného páru pre kombinácie nastavení predspracovania, konverzie na vektor a použitého kernelu.

Tabuľka 8.4 - Priemerný čas v ms, potrebný na spracovanie jedného vetného páru

	Základný		**Jednoduchá reprezentácia**		**Obohatený vektorový model dokumentu**	
	Gaussov kernel	Lineárny kernel	Gaussov kernel	Lineárny kernel	Gaussov kernel	Lineárny kernel
Základný	8,42	4,48	2,26	2,51	42,06	25,71
Značkovanie	7,3	5,04	3,28	2,75	36,59	27,4
Využitie synoným	8,46	5,25	2,75	3,18	47,65	36,31
Informačný prekryv	8,61	5,91	3,88	3,88	43,38	32,29
Kombinovaná metóda	8,04	5,08	3,01	2,83	35,62	26,02
Replikácia experimentu	4,68	2,54	0,56	0,78	38,63	24,82

8.4.3 Vzájomné porovnanie metód

Spravili sme niekoľko experimentov, aby sme detailnejšie porovnali naše najlepšie metódy s replikáciou pôvodného riešenia. V tabuľke nižšie (Tabuľka 8.5) uvádzame presnosť, návratnosť a správnosť pre zvolené metódy identifikácie parafrázovania namerané na dátovej vzorke Korpus1947. Všetky tieto metódy využívajú Gaussov kernel.

Tabuľka 8.5 - Detailné výsledky najlepších metód (s Gaussovým kernelom) pre Dataset A

Predspracovanie	**Konverzia**	**Jedná sa o parafrázu**	**Presnosť**	**Návratnosť**	**Správnosť**
Kombinovaná metóda	Obohatený vektorový model dokumentu	Áno	92,96%	93,16%	93,05%
		Nie	93,14%	92,94%	
Kombinovaná metóda	Jednoduchá reprezentácia	Áno	93,57%	93,37%	93,48%
		Nie	93,39%	93,58%	
Replikácia experimentu	Základný	Áno	90,31%	90,70%	90,49%
		Nie	90,66%	90,27%	

Výsledky nami vybraných metód na veľkej vzorke (Korpus1947 – 1 947 vetných párov) sú odlišné od výsledkov na dátovej vzorke Korpus600. Rozdiely medzi jednotlivými metódami sa zmenšili. No v tomto prípade sme najlepšie výsledky dosiahli pri *Kombinovanej metóde* a *jednoduchej reprezentácii* – 93,48%.

Po meraní sme ručne prezreli vetné páry, v ktorých nebolo správne identifikované parafrázovanie pre predspracovanie *Kombinovaná metóda*. Vo väčšine prípadov šlo o krátke vety, až 65% prípadov šlo o vety, ktoré majú menej ako 6 slov (za slovo sme počítali aj stop-slová), 9% viet obsahovalo vlastné podstatné mená.

S rovnakými nastaveniami systému sme vykonali experimenty s dátovou vzorkou Korpus202, ktorý sme vytvorili tak aby bolo veľmi ťažké odlíšiť parafrázované od neparafrázovaných vetných párov, ale súčasne aby vety v ňom dávali zmysel. Keďže je Korpus202 malý

(obsahuje iba 202 vetných párov), tak sme zvolili 202-krížovú validáciu [3], výsledky experimentov sú v tabuľke nižšie (Tabuľka 8.6).

Tabuľka 8.6 - Detailné výsledky najlepších metód (s Gaussovým kernelom) pre Dataset C

Predspracovanie	Konverzia	Jedná sa o parafrázu	Presnosť	Návratnosť	Správnosť
Kombinovaná metóda	Obohatený vektorový model dokumentu	Áno	85,71%	84,65%	84,64%
		Nie	83,65%	86,13%	
Kombinovaná metóda	Jednoduchá reprezentácia	Áno	88,17%	81,18%	85,14%
		Nie	82,56%	89,10%	
Replikácia experimentu	Základný	Áno	67,34%	65,31%	66,83%
		Nie	66,34%	68,31%	

Výsledky v tomto experimente sú celkovo nižšie ako v experimente s dátovou vzorkou Korpus1947. Dosiahli sme 85,14% správnosť pri *Kombinovanej metóde* a *Jednoduchej reprezentácii*. V tomto experimente rozdiely medzi našimi metódami a existujúcim riešením sa výrazne prehĺbili (*Kombinovaná metóda* 85,14%, *Replikácia experimentu* – 66,83%).

8.4.4 Porovnanie s existujúcim riešením

Pre adekvátne porovnanie výsledkov identifikácie parafrázovania prostredníctvom nami navrhnutého riešenia sme sa rozhodli zopakovať jeden z pôvodných experimentov identifikácie parafrázovania pomocou SVM. Pre možnosti predspracovania a dobré výsledky sme sa rozhodli zopakovať experiment z [Kozerva, 2006], ktorý je opísaný v kapitole 4.3.4 (Experiment č.2).

Predspracovanie, pomenované *Replikácia experimentu*, pridáva k vete, ktorá obsahuje vlastné podstatné meno, vlastnosť s hodnotou 1, ak vlastné podstatné meno neobsahuje, pridá vete vlastnosť s hodnotou 0. A pre prekryv vetného páru pridáva vlastnosti:

- počet zhodných slov vo vetách k celkovému počtu slov,
- prekryv slov vo vetách nájdený 4-skip-gramami,
- prekryv slov vo vetách nájdený algoritmom LCS (najdlhšia spoločná podpostupnosť),
- sémantická podobnosť medzi slovami (riešená pomocou synoným).

Použili sme *Základný* spôsob konverzie na vektor, ktorý je zhodný zo spôsobom konverzie uvedeným v [Kozerva, 2006].

Naše najlepšie riešenie dosiahlo lepšie výsledky aj na všetkých korpusoch. Na veľkej vzorke (Korpus1947) dosiahlo existujúce riešenie 90,49% správnosť a naše riešenie 93,05% a 93,48%. Taktiež malo naše riešenie lepšiu presnosť a návratnosť. Na dátovej vzorke Korpus600 dosiahlo existujúce riešenie 90,50% zatiaľ čo *Kombinovaná metóda* dosiahla 95% a 94,83% správnosť. Na dátovej vzorke Korpus202 dosiahlo existujúce riešenie len 66,83%, zatiaľ čo *Kombinovaná metóda* dosiahla 84,64% a 85,14%.

[3] Takáto validácia sa nazýva aj validácia vynechaním jedného (anglicky *leave one-out cross validation*).

8.4.5 Vyhodnotenie metód predspracovania a konverzie na vektor

Pri experimentoch sme prišli na to, že pre úspešnú identifikáciu parafrázovania v slovenčine má veľký vplyv určenie prekryvu na lexikálnej aj sémantickej úrovni (pri metóde konverzie na vektor *Jednoduchá reprezentácia* tvoria vektor len tieto hodnoty lexikálneho, sémantického prekryvu a vlastnosti vety). Takisto menej vlastností znamená vyššiu rýchlosť vyhodnotenia pomocou SVM a takisto aj menšie pamäťové nároky.

Naša navrhovaná metóda informačného prekryvu nepriniesla očakávané zlepšenie výsledkov, no v *Kombinovanej metóde* používame informačný prekryv počítaný pomocou slovných druhov, ktorý mierne zlepšil výsledky oproti využitiu synoným.

Ako najlepšia metóda (*Jednoduchá reprezentácia* a *Kombinovaná metóda*) sa ukázala metóda, pri ktorej sa vetám pridelí ich modálnosť, slovám sa priradia morfologické kategórie a nahradia sa lemou. Prekryv viet sa určuje nad lemami, synonymami slov pomocou 3-gramov, 4-skip-gramov a LCS, k tomu informačný prekryv počítaný pomocou váhovania slovných druhov.

9 Zhrnutie

V tejto práci sme navrhli metódu, pomocou ktorej dokážeme identifikovať parafrázovanie v slovenskom jazyku. Táto metóda vychádza z existujúcich metód použitých v anglickom jazyku, no je upravená pre špecifiká slovenského jazyka, ktorý na rozdiel od angličtiny patrí do flexívnych jazykov. Súčasťou tohto návrhu sú aj metódy predspracovania textu, viet a slov, ktoré sú nutné pre úspešnú identifikáciu parafrázovania. Parafrázovanie v texte identifikujeme prostredníctvom SVM (algoritmy podporných vektorov). Na predspracovanie textu používame nasledujúce metódy: 3-gramy, 4-skip gramy, LCS (najdlhšia spoločná podpostupnosť), určovanie jednoduchého prekryvu slov, značkovanie stop-slov, určovanie modálnosti a priraďovanie morfologických značiek lexikálnym jednotkám pomocou skrytých Markovovských modelov.

Výsledkom našej práce je prototyp, výsledky identifikácie parafrázovania pre jednotlivé metódy predspracovania a konverzie dvojice viet na vektor a anotovaný korpus vetných párov.

V rámci práce bola vytvorená aplikácia, ktorá pre dátovú vzorku pomocou krížovej validácie vyhodnotí presnosť identifikácie parafrázovania našej nami navrhnutej metódy pri rôznych nastaveniach vstupných parametrov a predspracovania. Súčasťou aplikácie je aj nami navrhnutá metóda morfologického značkovania, ktorú používame v procese predspracovania.

Druhý výstupom je systém na identifikáciu parafrázovania v dokumentoch, ktorý je implementovaný ako webová služba.

Nami navrhnutá knižnica pre morfologické značkovanie dokázala určiť správnu morfologickú značku pre 95,35% lexikálnych jednotiek, čo považujeme za výrazné zlepšenie oproti pôvodnému návrhu, ktorý dosahoval 87%.

Experimentálne sme určili najlepšie parametre pre nami navrhnutý model identifikácie parafrázovania a dosiahnuté výsledky s týmto modelom sme porovnali s existujúcim riešením. Na dátovej vzorke, ktorá obsahovala takmer 2 000 vetných párov, naše riešenie dosiahlo úspešnosť 93,48%. Túto hodnotu sme dosiahli pre nami navrhnutú metódu *Kombinovaná metóda*. Ako reprezentáciu sme použili nami navrhnutý model *Jednoduchá reprezentácia*, Gaussov kernel a parameter *complexity* s hodnotou 5.5. Existujúce riešenie na rovnakej vzorke dosiahlo úspešnosť 90,49%. Na dátovej vzorke, ktorá obsahovala 200 vetných párov, ale výrazne ťažších na negatívnu identifikáciu, sme dosiahli úspešnosť 85,14%, zatiaľ čo pôvodné riešenie len 66,83%.

9.1 Budúca práca

Náš systém na identifikáciu parafrázovania identifikuje parafrázovanie na úrovni viet, no pri reálnom parafrázovaní dlhších úsekov textu sa stáva, že dlhšia veta je rozdelená na viac menších, alebo naopak, krátke vety sú spájané do súvetia. Tento problém navrhujeme riešiť porovnávaním jednej vety (zo zdrojového dokumentu) k dvom alebo trom vetám, ktoré stoja za sebou. A to tak, že jednotlivé vety sa predspracujú jednotlivo, no prekryvové vlastnosti sa vypočítajú tak, ako keby porovnávané vety tvorili súvetie.

V našom prípade je proces identifikácie parafrázovania veľmi časovo náročný, pretože porovnávame dokumenty každý s každým a v rámci porovnávania dokumentu porovnávame každú vetu s každou. Preto ak by sme chceli identifikovať parafrázovanie vo veľmi veľkých korpusoch, je vhodné vytypovať dokumenty alebo časti dokumentov, v ktorých má význam identifikovať parafrázovanie. Napríklad pomocou metód zhlukovania (v dokumentoch, ktoré majú rovnakú tému, je parafrázovanie pravdepodobnejšie) alebo pomocou rýchlych a výpočtovo menej náročných metód identifikáciu podobnosti a plagiátorstva, ako napríklad 3-gramy alebo kosínusová podobnosť.

A v neposlednom rade je možné náš systém na identifikáciu parafrázovať optimalizovať, pretože bol navrhnutý na experimentovanie a použité metódy predspracovania je možné implementovať efektívnejšie.

Bibliografia

Accord.NET. Accord.NET Framework. [Online] [Dátum: 25. 11 2013.] http://accord-framework.net/.

Auder, Miloš. 2011. Rozpoznávanie slovných druhov. *SAV*. [Online] 2011. [Dátum: 3. 30 2013.] http://vi.ikt.ui.sav.sk/@api/deki/pages/1248/pdf.

Brockett, Chris a Dolan, William. 2005. *Support Vector Machines for Paraphrase Identification and Corpus Construction.* s.l. : Natural Language Processing Group Microsoft Research, 2005.

Češka, Zdeněk. 2007. *Využití n-gramù pro odhalování plagiátù.* Plzeň : Katedra informatiky a výpočetní techniky, Západočeská univerzita, 2007.

Hunspell. 2011. Hunspell. [Online] 2011. [Dátum: 30. 3 2013.] http://hunspell.sourceforge.net/.

Chitra, A. 2010. *Paraphrase Identification using Machine Learning Techniques.* s.l. : ACM, 2010. Zv. RECENT ADVANCES in NETWORKING, VLSI and SIGNAL PROCESSING. ISBN: 978-960-474-162-5.

Chudá, Daniela doc. Mgr. 2011. PlaDeS. [Online] 2011. [Dátum: 12. 3 2013.] http://www2.fiit.stuba.sk/~chuda/plagiarism/PlaDeS.html.

Kozerva, Zornitsa. 2006. *Paraphrase Identification on the Basis of Supervised Machine Learning Techniques.* s.l. : ACM, 2006.

Lintean, Michain. 2009. *Paraphrase Identification Using Weighted Dependencies and Word Semantics.Informatica.* s.l. : ACM, 2009.

Manning, Christopher D., Raghavan, Prabhakar a Schutze, Hinrich. 2008. *Introduction to Information retrieval.* New York : Cambridge University Press, 2008. ISBN: 978-0-521-86571-5.

Martin, Jačala. 2010. Štatistický POS tagger. [Online] SAV, 11. 12 2010. [Dátum: 1. 4 2013.] http://vi.ikt.ui.sav.sk/User:Martin.Jacala?view=home.

Microsoft. WCF Extensibility Guidance. [Online] Microsoft. [Dátum: 25. 8 2013.] http://msdn.microsoft.com/en-us/library/gg132853.aspx.

Nemirovsky, Danil a Dobrynin, Vladimir. 2008. *Word importance discrimination using context.* s.l. : TREC, 2008.

Oravec, Ján PhDr. DrSc. a Laca, Vincent. 1973. *Príručka slovenského pravopisu.* Bratislava : Slovenské pedagogické nakladateľstvo, 1973.

Páleš, Emil. 1990. *Co-operation of Syntax and Semantics in Flexive Languages.* s.l. : Journal of Experimental & Theoretical Artificial Intelligence, 1990.

Páleš, Emil. 1994. *SAPFO parafrázovač slovenčiny.* Bratislava : VEDA - Vydavateľstvo Slovenskej akadémie vied, 1994. ISBN 80-224-0109-9.

Parlič, Ján doc., Ing., PhD. 2010. *Dolovanie znalostí z textov.* Košice : Agentúra pre výskum a vývoj, FEI TUKE, 2010. ISBN 978-80-89284-62-7.

Piaček, Jozef a Kravčík, Miloš. 1999. FILIT. *Parafráza.* [Online] 1999. [Dátum: 24. 2 2013.] http://ii.fmph.uniba.sk/~filit/fvp/parafraza.html.

Ripka, Ivan prof. PhDr. DrSc., Imrochová, Mária PhD. a Skladaná, Jana PhDr. CSc. 2006. *Príručka slovenského pravopisu.* Bratislava : OTTOVO NAKLADATEĽSTVO, 2006. ISBN 80-969159-1-6.

Sangeetha, R. a Kalpana, B. 2010. *Optimizing the kernel selection for support vector machines using performance measures.* New York : ACM, 2010. ISBN: 978-1-4503-0194-7.

SAV, JuLS. 2014. Slovenský WordNet. *Slovenský národný korpus.* [Online] Jazykovedný ústav Ľ. Štúra Slovenskej akadémie vied, 25. 1 2014. [Dátum: 25. 1 2014.] http://korpus.juls.savba.sk/WordNet.html.

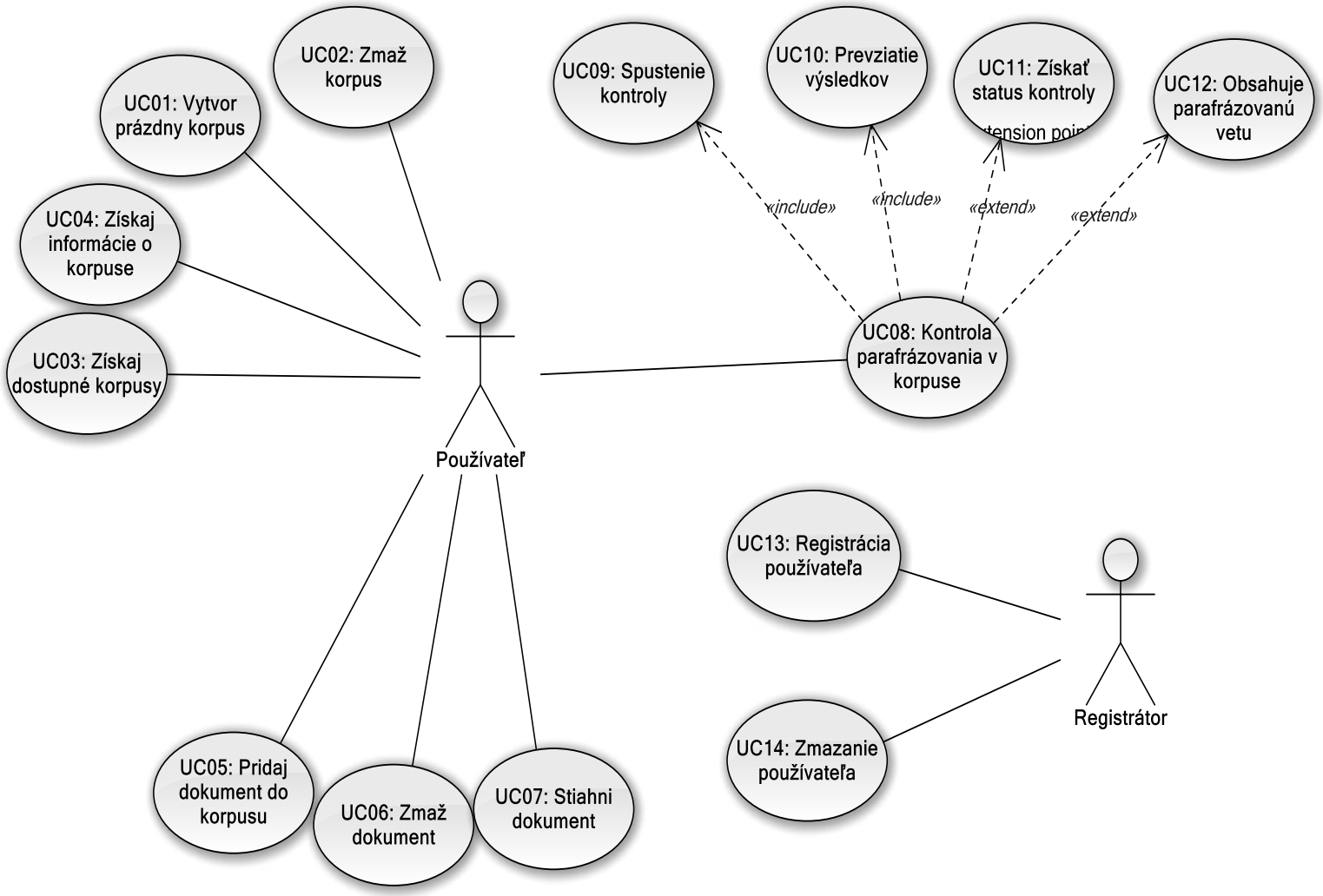
Taria, Hirothosi a Haruno, Masahiko. 1999. *Feature selection in SVM text categorization.* s.l. : ACM, AAAI, 1999. ISBN:0-262-51106-1.

Vapnik, V. N. 1995. *The Nature of Statistical Learning Theory.* New York : Springer,, 1995.

Prílohy

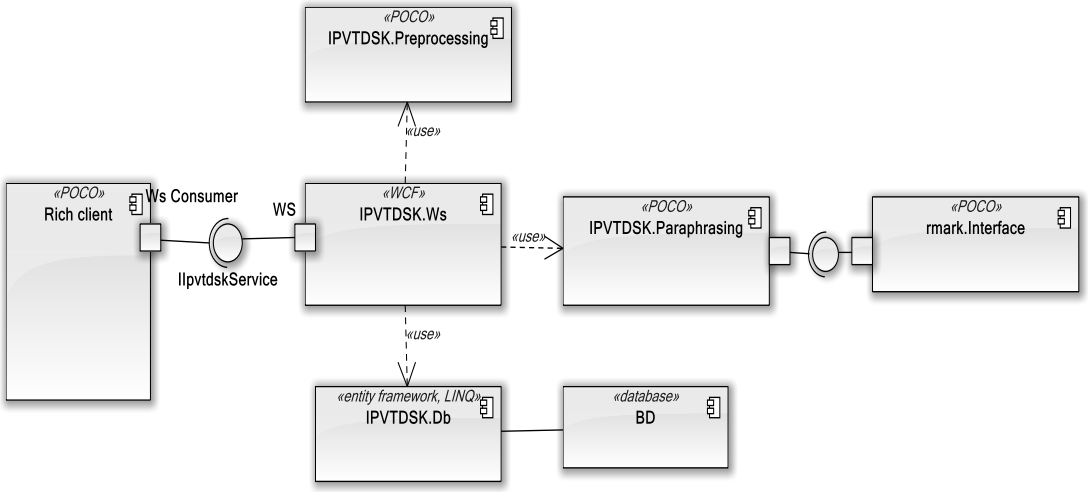
Príloha A – Technická dokumentácia k systému na identifikáciu parafrázovania

V tejto prílohe sú zobrazené UML diagramy k systému na identifikáciu parafrázovania v slovenskom jazyku.



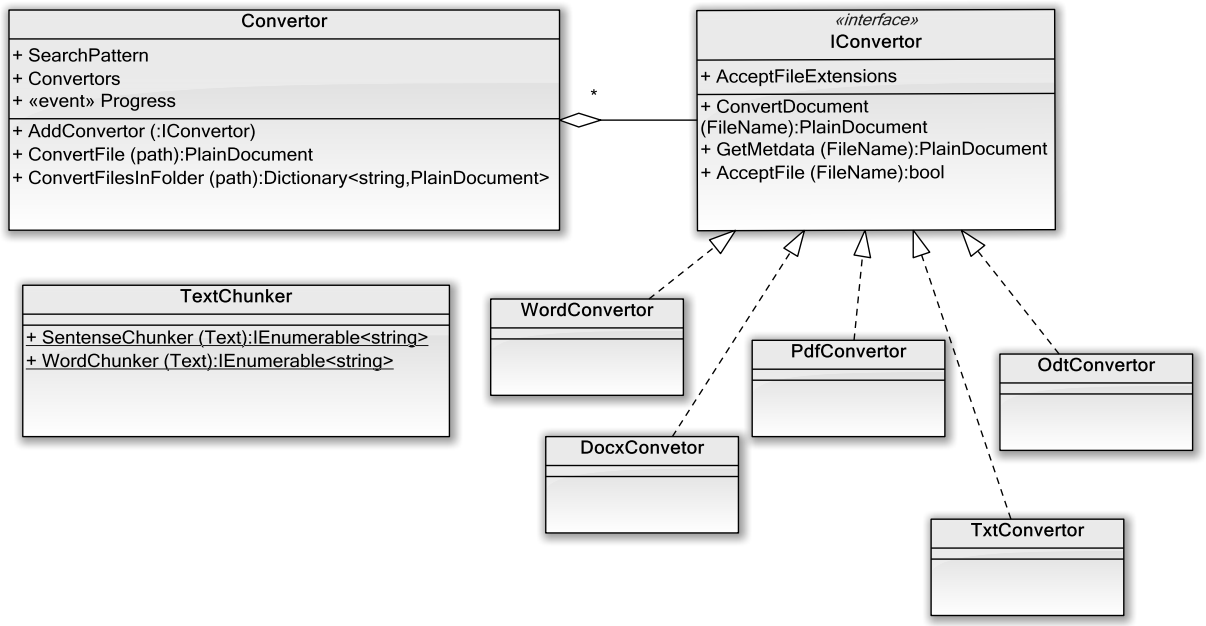
Obrázok P.1 - Diagram prípadov použitia pre webové služby

Náš diagram prípadov použitia má dva typy používateľov (role) *Registrátor* – spravuje používateľov (*UC13* – registrácia nového používateľa, *UC14* – zmazanie existujúceho používateľa) a *Používateľ* – má asociované všetky prípady použitia, ktoré súvisia s identifikáciou parafrázovania (*UC01* až *UC03* – spáva korpusov, *UC05* až *UC07* – správa dokumentov v korpuse, *UC08* až *UC12* – identifikácia parafrázovania v korpuse).



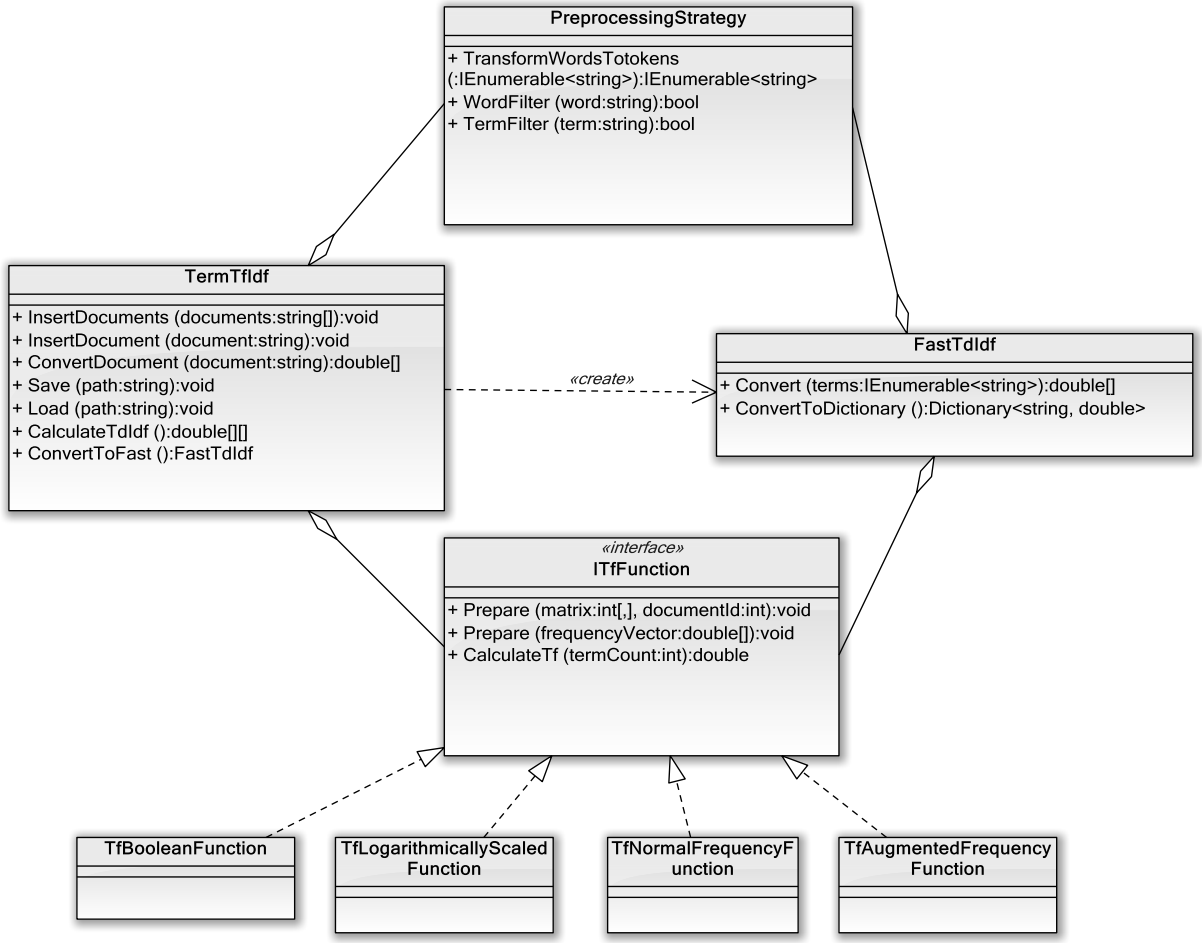
Obrázok P.2 - Diagram komponentov

Na diagrame komponentov sú znázornené jednotlivé komponenty systému. Komponent *Rich client* predstavuje tenkého klienta, ktorý konzumuje webovú službu z rozhraním *IIpvtdskService*. Komponent *IPVTDSK.Ws* je webová služba, ktorá tvorí vstupný bod do celého systému, je implementovaná prostredníctvom WCF rámca. *IPVTDSK.Preprocessing* je komponent starajúci sa o predspracovanie textu a konverziu textových dokumentov. Komponent *IPVTDSK.Db* slúži ako vrstva dátových služieb, cez ktorú aplikácia komunikuje s databázou. Tento komponent je implementovaný pomocou *Entity Frameworku*. *DB* je databáza, ktorú naša aplikácia využíva. Komponent *IPVTDSK.Paraphrasing* slúži na samotnú identifikáciu parafrázovania, obsahuje pre to potrebnú infraštruktúru a filtre, ktoré sa pri tom používajú. Komponent *rmark.Interface* je knižnica pre morfologické značkovanie, v diagrame je zvýraznenie oddelenie tohto komponentu od zvyšku systému.



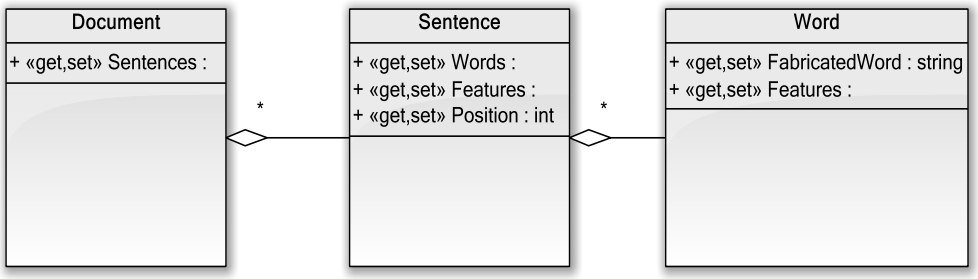
Obrázok P.3 - Diagram tried pre predspracovanie dokumentov

Diagram na obrázku vyššie (Obrázok P.3) zobrazuje triedy pre predspracovanie dokumentov – konvertor dokumentov (trieda *Convertor*) a triedy, ktoré používa na konverziu jednotlivých typov súborov (napríklad trieda *DocxConvertor*). Trieda *TextChunker* slúži na kúskovanie textu na vety a slová.



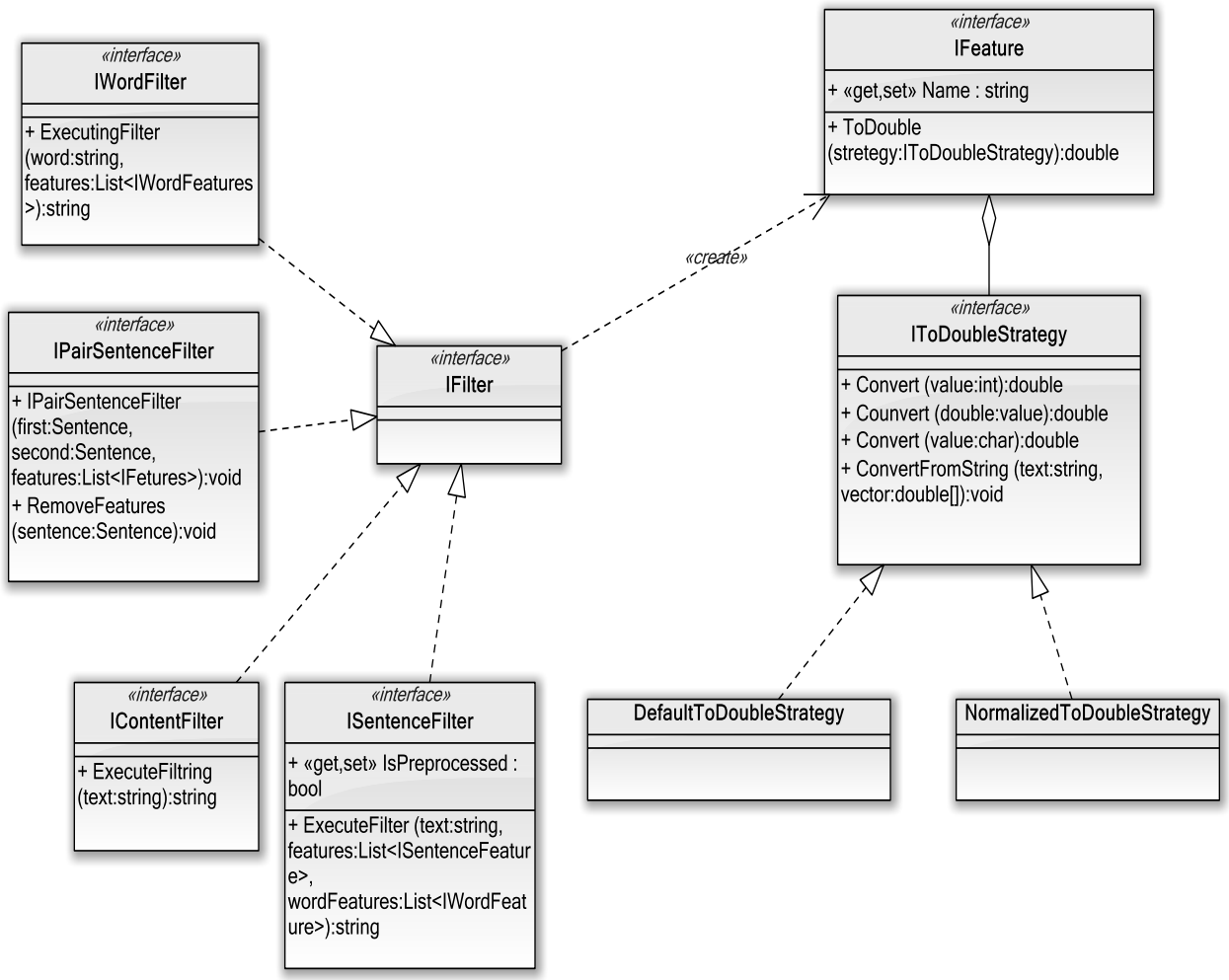
Obrázok P.4 - Diagram tried pre predspracovanie pomocou Tf-Idf

Diagram na obrázku vyššie zobrazuje triedy, ktoré sa používajú pri výpočte a použití *tf-idf* a *idf váh*. Trieda *TermTfIdf* slúži na výpočet *idf* a vytvára inštanciu *FastTdIdf*, ktorý počíta samotné *tf-idf*. Trieda *PreprocessingStrategy* určuje stratégiu predspracovania textu a extrakciu termov pre vytvorenie matice *idf* a výpočtu *tf-idf*. Rozhranie *ITfFunction* predstavuje stratégiu počítania *tf*.



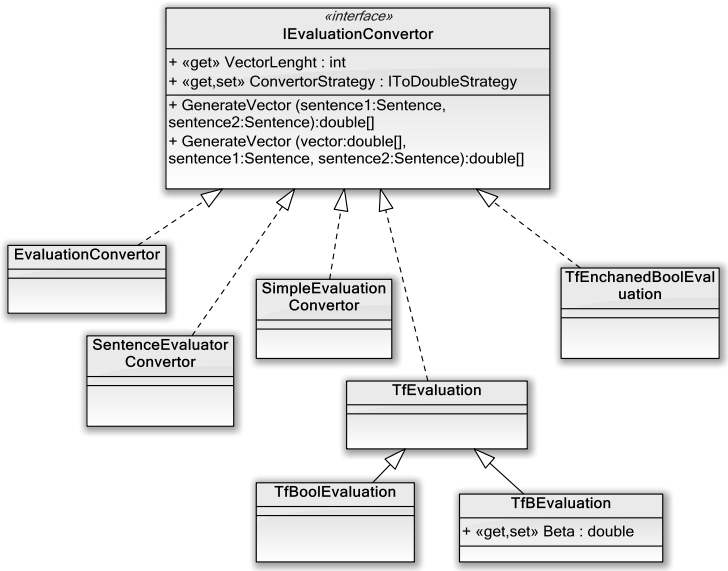
Obrázok P.5 - Diagram tried pre štruktúru dokumentov

Diagram zobrazuje štruktúru dokumentov, nachádzajú sa v ňom triedy *Document*, *Sentence* a *Word*, ktoré tvoria dokument.



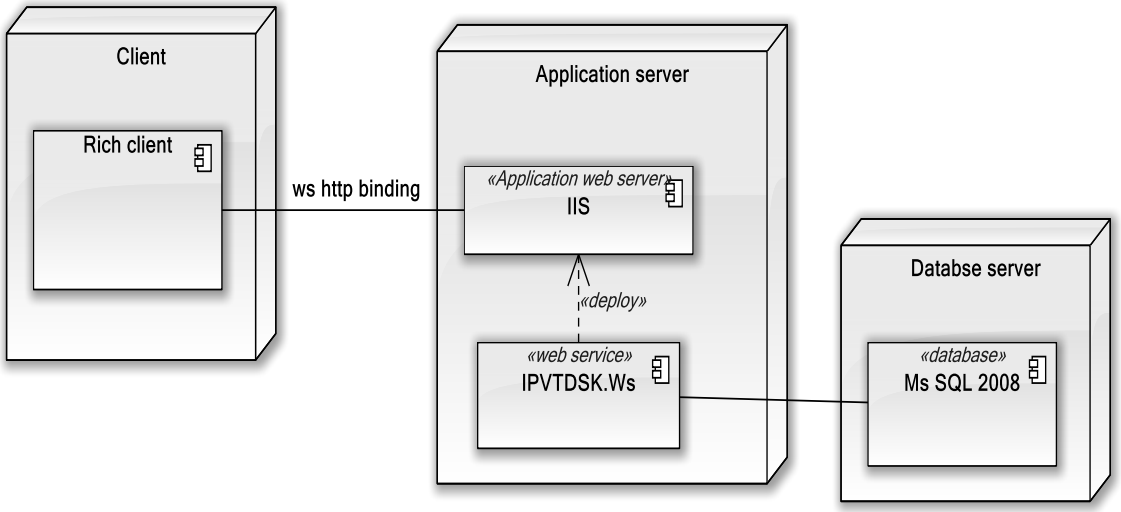
Obrázok P.6 - Diagram tried pre filtre a vlastnosti

Diagram zobrazuje štruktúru filtrov a vlastností. Od rozhrania *IFilter* dedia rozhrania pre filtre, ktoré sa aplikujú na dokument (*IContentFilter*), vetu (*ISentenceFilter*), slovo (*IWordFilter*) a vetný pár (*IPairSentenceFilter*). Rozhrania pre jednotlivé typy filtrov implementujú triedy konkrétnych filtrov. Každý filter môže lexikálnej jednotke pridávať vlastnosti, každá vlastnosť musí implementovať rozhranie *IFeature*.



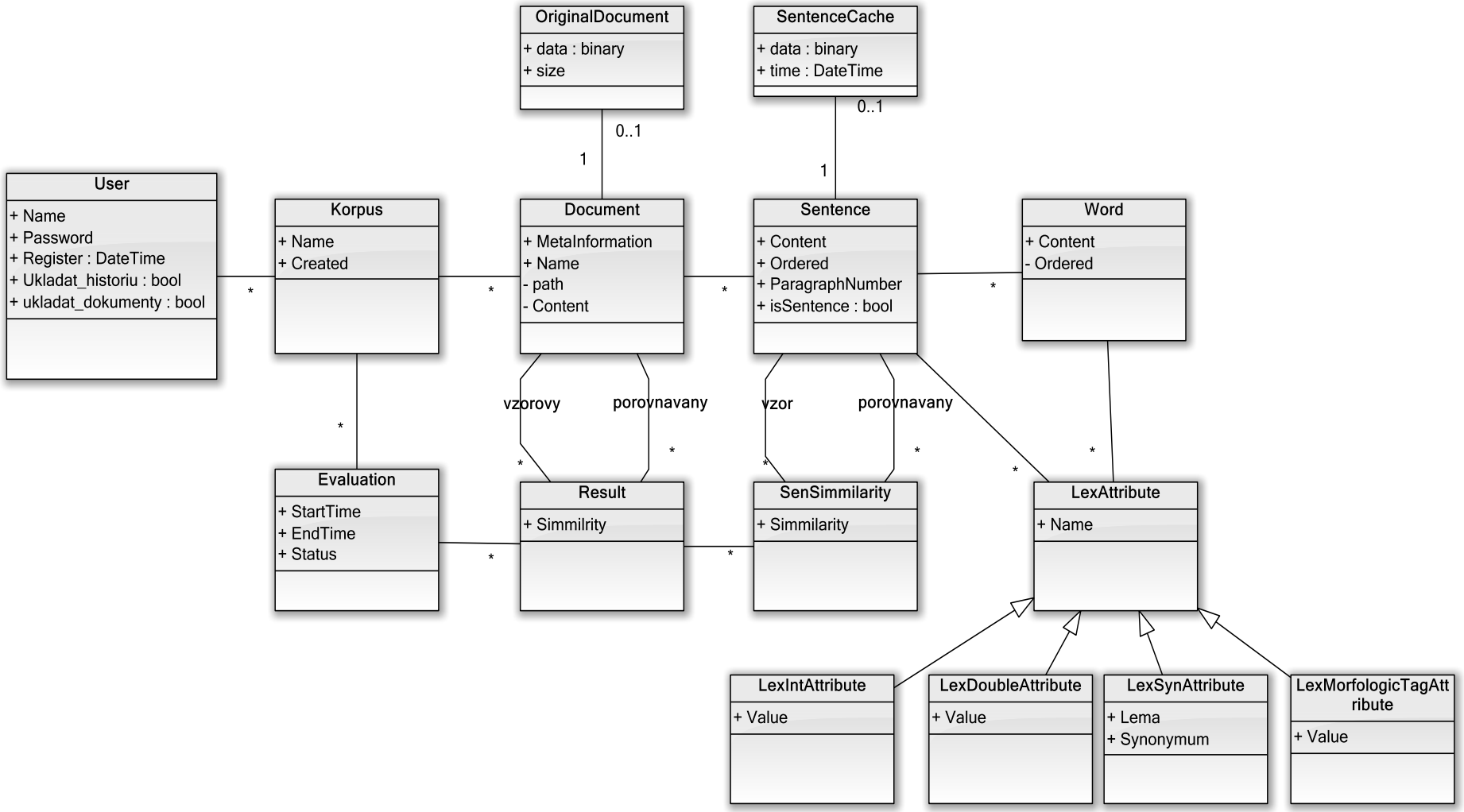
Obrázok P.7 - Diagram tried pre konverziu vetného páru na vektor pre SVM

Diagram hore zobrazuje rozhranie *IEvaluationConvertor*, ktoré sa stará o konverziu vetného páru na číselný vektor.



Obrázok P.8 - Diagram rozmiestnenia

Diagram rozmiestnenia zobrazuje spôsob nasadenia nášho systému na aplikačný a databázový server. Komponent *IPTDSK.Ws* v tomto obrázku reprezentuje nami vytvorené riešenie.



Obrázok P.9 - Diagram entít dátového modelu

Diagram entít zobrazuje entity vytvorené *Entity Frameworkom*, ktoré slúžia na ukladanie dát do databázy. Entita *User* predstavuje používateľa, ktorý spravuje korpusy v entite *Korpus*. *Korpus* obsahuje dokumenty rozdelené na vety a slová (entity *Document*, *Sentence*, *Word*) a k nim priradené vlastnosti v entite *LexAtribute*. Entita *SentenceCache* slúži na uloženie serializovaných viet s vlastnosťami a tým urýchľuje získavanie predspracovaných viet z databázy. Entita *OriginalDocument* uchováva originálne binárne súbory dokumentov. Entita *Evaluation* nesie informácie o odštartovaných a ukončených identifikáciách parafrázovania v korpuse. Entita *Result* uchováva, ktoré dokumenty boli parafrázované a entita *SemSimmilarity* uchováva, ktorá veta je s ktorou parafráza v rámci jednej identifikácie parafrázovania.

Príloha B – Opis projektov riešení (solution) zdrojových kódov

V tabuľke nižšie (Tabuľka P.1) uvádzame opis projektov v riešení (*solution* – terminológia z *Visual Studia*), ktorý sme implementovali.

Tabuľka P.1 - Opis projektov riešení (solution) zdrojových kódov

Názov projektu	Popis projektu
IPVTDSK.Preprocessing	Knižnica na konverziu dokumentov, predspracovanie textu a výpočet *tf-idf*.
IPVTDSK.Paraphrasing	Knižnica na identifikáciu parafrázovania.
IPVTDSK.Dd	Knižnica vytvorená pomocou *Entity Frameworku*, slúži ako vrstva dátových služieb.
IPVTDSK.Utils	Knižnica obsahujúca pomocné triedy a *extensions*, napríklad na prácu s XML, reflexiou, súbormi.
rmark.Interfaces	Knižnica pre morfologické značkovanie.
rmark.Miner	Aplikácia na vytvorenie HMM z korpusu r-mark.
rmark.Miner2	Aplikácia na vytvorenie HMM z korpusu *r-mark*, odoslaná na *SAV*.
pmark.Precision	Aplikácia na validáciu morfologického značkovania.
IPVTDSK.EntrphyCalculator	Aplikácia pre zadaný korpus počíta entropiu a idf termov.
IPVTDSK.SampleGuiCollector	Aplikácia na zber parafrázovaných viet.
IPVTDSK.SimpleTrestingSvm	Aplikácia na validáciu metód identifikácie parafrázovania.
IPVTDSK.Trainer	Aplikácia pre natrénovanie SVM pre nasadenie.
IPVTEDSK.CollectorDistibuteAndMerge	Aplikácia na distribúciu aplikácie pre zber parafráz a následné spojenie súborov s parafrázami do jedného.
IPVTDSK.Client	Jednoduchý desktopový klient konzumujúci webovú službu na identifikáciu parafrázovania.
IPVTDSK.WS	Webová služba pre identifikáciu parafrázovania.
IPVTDSK.WS.Host	Aplikácia umožňujúca spustiť webovú službu na identifikáciu parafrázovania, bez nutnosti IIS.
IPVTDSK.Registrator	Aplikácia slúžiaca na registráciu nových používateľov.
Testing	Unit testy pre časti celého projektu (najmä algoritmy).

Príloha C - Tabuľka filtrov

V tejto prílohe uvádzame tabuľku filtrov použitých pri predspracovaní viet pri procese identifikácie parafrázovania. Tabuľka obsahuje informácie o tom, v ktorej časti procesu je filter použitý a aj slovný opis jeho činnosti.

Tabuľka P.2 -Tabuľka filtrov

Názov filtra	Použije sa na	Opis činnosti
AbbreviationsFilter	dokument	Expanduje najpoužívanejšie slovenské skratky.
NameResolverFilter	dokument	Odstraňuje prvé a stredné meno a iniciály.
RemoveBracesFilter	dokument	Odstraňuje text medzi zátvorkami.
LenghtRelativeSentenceFilter	veta	Určí relatívnu dĺžku vety v slovách.
OunNunSentenceFilter	veta	Označí vetu, ak obsahuje vlastné podstatné meno.
ModalnostSentenceFilter	veta	Určuje modálnosť vety.
RmarkTaggerFilter	veta	Pridá slovám vety morfologické značky.
LematizingSynonymMetaFilter	slovo	K slovu priradí jeho lemu a synonymum.
LematizingTextFilter	slovo	Slová nahradí ich lemou.
StopWordMarker	slovo	Označí stop slová.
LcsMarkerPairFilter	dvojica viet	Výpočet prekryvu pomocou metódy LCS nad slovami a značkovanie prekrytých slov vo vetách.
LcsMorgologicPairFilter	dvojica viet	Výpočet prekryvu pomocou metódy LCS nad morfologickými značkami a značkovanie prekrytých slov vo vetách.
LcsPairFilter	dvojica viet	Výpočet prekryvu pomocou metódy LCS nad lemami slov a značkovanie prekrytých slov vo vetách.
LcsSynonymPairFilter	dvojica viet	Výpočet prekryvu pomocou metódy LCS nad synonymami slov a značkovanie prekrytých slov vo vetách.
NGramMorfologicPairFilter	dvojica viet	Výpočet prekryvu pomocou 3-gramov nad morfologickými značkami a značkovanie prekrytých slov vo vetách.
NGramPairFilter	dvojica viet	Výpočet prekryvu pomocou 3-gramov nad lemami slov a značkovanie prekrytých slov vo vetách.
NGramSynonymPairFilter	dvojica viet	Výpočet prekryvu pomocou 3-gramov nad synonymami slov a značkovanie prekrytých slov vo vetách.
SimpleOverlapFilter	dvojica viet	Počíta jednoduchý prekryv – pomer počtu zhodných slov k počtu slov v kratšej vete.
SimSkipGramFilter	dvojica viet	Výpočet prekryvu pomocou 4-skip-gramov nad lemami slov a značkovanie prekrytých slov vo vetách.
SkipGramMorfologicPairFilter	dvojica viet	Výpočet prekryvu pomocou 4-skip-gramov nad morfologickými značkami a značkovanie prekrytých slov vo vetách.
SkipGramSynonymPairFilter	dvojica viet	Výpočet prekryvu pomocou 4-skip-gramov nad synonymami slov a značkovanie prekrytých slov vo vetách.

Názov filtra	Použije sa na	Opis činnosti
WeightLcsEntrophyFilter	Dvojica viet	Vypočíta informačný prekryv pomocou LCS, za použitia *idf* a entropického váhovania.
WeightMogphologicLcsPairFilter	Dvojica viet	Vypočíta informačný prekryv pomocou LCS, priradením váh (v rozsahu 0,0 až 1,0) pre jednotlivé slová podľa slovného druhu.
WeightMogphologicLcsPairFilter2	Dvojica viet	Vypočíta informačný prekryv pomocou LCS, priradením váh (v rozsahu 1,0 až 2,0) pre jednotlivé slová podľa slovného druhu.
WeightNGramPairFilter	Dvojica viet	Vypočíta informačný prekryv pomocou 3-gramov, priradením váh pre jednotlivé slová podľa slovného druhu.
WeightSkipGrams	Dvojica viet	Vypočíta informačný prekryv pomocou 4-skip-gramov, priradením váh pre jednotlivé slová podľa slovného druhu.

Príloha D – Opis webových služieb

V tejto kapitole si opíšeme jednotlivé webové služby, ktoré poskytuje náš systém.

Tabuľka P.3 - Opis webových služieb

Webová služba	Popis
`Guid CreateEmptyCorpus(string name)`	Vytvorí nový prázdny korpus a vráti jeho id.
`Void RemoveCorpus(Guid id)`	Zmaže korpus.
`List<CorpusInfoDto> GetMyCorpuses()`	Služba vráti kolekciu objektov, nesúce informácie o korpusoch používateľa. Každý objekt nesie identifikátor a meno korpusu, počet dokumentov v korpuse.
`CorpusContentDto GetDocumentsInCorpus(Guid id)`	Vracia základné informácie o dokumentoch nachádzajúcich sa v korpuse.
`Gui dAddPlainDocument(Guid corpusId, string Content, DocumentInfoDto metainformation)`	Pridá dokument do korpusu. Dokument je reprezentovaný textovým reťazcom a metainformáciami.
`OriginalDocumentResponse AddDocument(OriginalDocumentDto document)`	Pridá dokument do korpusu ako binárny súbor (musí byť v podporovanom formáte).
`void RemoveDocument(Guid documentId)`	Zmaže dokument v korpuse.
`Stream GetOriginalDocument(Guid documentId)`	Vráti originálny dokument, ak existuje.
`void StartEvaluation(Guid copusId)`	Služba spustí proces identifikácie parafrázovania v korpuse.
`EvaluationStatusDto GetEvaluationStatus(Guid corpusId)`	Vráti správu o tom, v akom stave (progrese) sa nachádza proces identifikácie parafrázovania. (Stavy: žiaden, predspracovanie, proces prebieha, proces bol dokončený, proces skončil s chybou.)
`EvaluationResultsDto GetEvaluationResult(Guid corpusId)`	Služba vráti výsledky identifikácie parafrázovania. Výsledky obsahujú dvojice dokumentov, ktoré boli parafrázované. Ku každej dvojici je zoznam viet s pozíciami v pôvodných dokumentoch, ktoré boli parafrázované.

Príloha D - Konzumácia webovej služby

V tejto kapitole poskytujeme návod, ako konzumovať webovú službu pre identifikáciu parafrázovania vo *Visual Studiu 2010* a jazyku C#.

1. V otvorenom projekte, v ktorom chceme konzumovať webovú službu, klikneme pravým tlačidlom myši a zvolíme *Add Service Reference*.
2. V okne *Add Service Reference* zadáme adresu služby (napríklad *http://localhost:8732/ipvtdsk/*), vyplníme *namespace* a stlačíme *OK*. Týmto krokom sa vytvorili potrebné proxy triedy pre volanie webovej služby.
3. V programovom kóde je potrebné pri inicializácii služby zadať prihlasovacie meno, heslo a zakázať validáciu certifikátu.

V nasledujúcej ukážke kódu je príklad toho, ako vytvoriť nový korpus.

```
IpvtdskContractClient client = new IpvtdskContractClient();
client.ClientCredentials.UserName.UserName = „name“;
client.ClientCredentials.UserName.Password = „password“;

client.ClientCredentials.ServiceCertificate.Authentication.Certific
ateValidationMode =
System.ServiceModel.Security.X509CertificateValidationMode.None;
Guid corpusId = client.CreateEmptyCorpus(„MyFirstCorpus“);
```

Príloha E – Obsah elektronického média

Priečinky a súbory v koreňovom priečinku elektronického média:

`Anotations`	-	Priečinok s anotáciami.
`Aplication`	-	Priečinok s binárnymi súbormi.
`Conferences`	-	Priečinok s príspevkami na konferencie.
`Datasets`	-	Priečinok s dátovými vzorkami.
`DP_Jozef_Gajdos.pdf`	-	Text diplomovej práce.
`Licence`	-	Priečinok s licenčnými zmluvami k *r-mark* aj k samotnej práci.
`SourceCode.rar`	-	Archív obsahujúci zdrojové kódy a zálohu databázy.
`Zadanie.pdf`	-	Zadanie diplomovej práce.

Elektronické médium **neobsahuje korpus *r-mark***, pretože nemôže byť poskytnutý tretím osobám, bez súhlasu *Slovenskej akadémie vied*.

Príloha F – Príspevky na konferencie

Táto príloha obsahuje nasledujúce 2 články:

- Článok „*Identification of Higher Paraphrasing in Slovak Language*“ publikovaný a prezentovaný na konferencii IIT.SRC 2014, ktorá sa konala dňa 29.04.2014.
- Článok „*Higher Paraphrasing Identication in Slovak Language with SVM*“ na konferenciu TSD 2014 (*Text, Speech and Dialogue*), ktorá sa bude konať v Brne dňa 8.12.2014. Článok je aktuálne v štádiu posudzovania.

Identification of Higher Paraphrasing in Slovak Language

Jozef GAJDOŠ*

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova2, 842 16 Bratislava, Slovakia
xgajdos@is.stuba.sk

Abstract. This paper deals with the identification of higher paraphrasing in text documents written in the Slovak language. By higher paraphrasing we mean that the paraphrased text was created by using more advanced methods, for instance synonyms replacement. We present here our own approach that uses support vector machine (SVM) and contains several specifics in the pre-processing phase regarding Slovak language. Our solution takes into account the Slovak language characteristics and use of the currently available methods and dictionaries. The main result of our work is the created POS tagger for Slovak language and the way how two sentences and their properties are represented in a vector and then compared on similarity. The carried experiments clearly show that our approach is working well.

1 Introduction

Today large volume of data can be found in an electronic version that is usually available online on the Internet. Most of these data are texts written in natural language. Users often have to search for relevant content on the web based on keywords, phrases or by selecting different categories. Being able to query such data and detect similar content within different texts is therefore crucial these days. Such similarity detection on higher level is however an open problem. To the open problems in detecting similarity belongs the identification of paraphrasing at a higher level. By a paraphrasing on a higher level we refer to the combination of changes in word order, inserting or removing words from sentences and using synonyms.

Paraphrasing is basically a way of saying something in different words while the original idea and its importance during paraphrasing do not change. There are many ways how to paraphrase text in Slovak language. Moreover, Slovak language belongs to the group of inflecting languages. Core vocabulary consists of 60 000 words, has a complex syntax, morphology, and is quite ambiguous.

Identification of paraphrasing in natural language can be helpful in identifying plagiarism or detecting topic similar documents. Such identification can be for instance useful when clustering text data or in case we perform knowledge summarization.

This document is structured as follows. Section 2 describes the existing work in the field of plagiarism identification. In sections 3 and 4 we describe and experiment with our own approach for plagiarism detection. Section 5 summarizes this paper.

---

* Master degree study programme in field: Software Engineering
Supervisor: TomášKučečka, Institute of Informatics and Software Engineering, Faculty of Informatics and Information Technologies STU in Bratislava

IIT.SRC 2014, Bratislava, April 29, 2014, pp. 1–6.

2 Related work

In papers [1], [2] and [3] authors introduce an approach to identification of paraphrasing through using the machine learning. The granularity of identification was on sentence level. All proposed methods were designed for the English language and tested on Microsoft corpus, which has marked paraphrased sentences.

In paper [1] are described experiments with machine learning algorithms, SVM and k-nearest neighbour. As a distance measure authors used the normalized Manhattan distance and maximum entropy. Input to the machine learning algorithms were pairs of sentences with the following features:

- the count of matching words between sentences to the total number of words,
- overlap of 4-skip-gramsbetween sentences,
- overlap of sequences in sentences found by LCS (longest common subsequence) algorithm,
- semantic similarity between words,
- sentence containing its own noun received flag 1, else the flag was 0.

In several experiments authors investigated the effect of these features(LCS, n-grams and skip-grams, semantic overlap, etc.). The following table (Table 1) shows the overall results. These experiments showed that the most appropriate technique to identify the paraphrasing is the SVM.

Table 1.Precision and recall for used types of machine learning

Machine learning algorithm	k - nearest neighbor	Maximum entropy	SVM
Precision	*64.68%*	*66.44%*	*70.43%*
Recall	*71.13%*	*70.50%*	*74.12%*

In paper [2] it is also stated that the SVM is the most appropriate approach to the identification of paraphrasing. Input to the SVM represented:

- string similarity
- count of matching words
- Levenshtein distance between words,
- stems of words,
- marking semantically similar words (synonyms).

The authors used linear kernel with the following parameters –constant10^{-3} and complexity 0.5. The results showed that the overlap of phrases and labelling of semantically similar words significant impact on the precision.

In work [3]the proposed solution also used the SVM where the input to the learning algorithm represented a pair of sentences with following features:

- Overlap between sentences determined by the skip-gram method.
- Overlap, which is determined by the LCS algorithm calculated as the ratio of the number of words found by LCS to the number of words in each sentence.
- The semantic similarity of verbs and nouns in sentences. Semantic similarity was determined by the semantic content of the word, the likelihood of its occurrence and information from WordNet.
- Similarity of numeric attributes in sentences.

- Presence of own nouns in a sentence.

In experiments authors used the linear and the Gaussian kernel function. The authors state that the addition of lexical and semantic information increased accuracy of plagiarism identification. The resulting precisions amounted to 70%.

3 Our solution

We designed our solution to the identification of paraphrasing based on an analysis of existing approaches that were described in the previous section. The main difference is that we focus on Slovak language. Therefore, we have to take into account the specifics of this language, which can be seen mainly in its pre-processing stages. We decided to use support vector machines, because the paper [1] shows their relevance in identifying paraphrasing in natural language.

Input to our method is a textual document that is first converted into a plain text. This text is then chunked on sentences. The two documents are compared on sentence level which means that we detect paraphrasing within pairs of sentences. In the following subsections we give a detail description how our method works.Identification of paraphrasing.

In the first step we determine various properties of the sentence as its length and modality. In the second step individual sentences are divided into words and these words are assigned to individual properties. The words are assigned morphological tags, their lemma and common synonym are determined and marked if they are a stop-word.

Subsequently, for every pair of sentences we determine their overlap on semantic, morphological and lexical level through:

- LCS algorithm,
- 3-grams where 1-gram is one word,
- 4-skip-grams where 1-gram is one word,
- simple overlap-count of words equals a minimum count of words in sentences.

3.1 Vector representation

The key step of our approach is the way how we represent pair of sentences with their properties in a vector. This vector is then handled by the SVM. In the first part of the vector are the normalized attributes that the two sentences overlap. These attributes are followed by the two sentences. In the first part of each sentence are the properties of a sentence, followed by the words of the sentence. Individual words together with their properties inserted one character after the other, as in the original sentences. For every word there is a fixed number of items reserved in the vector. If a word has fewer letters than these items, the remaining items are filled with zeroes. If a word has more letters than these items, the rest of the letters will be discarded. Length of these items is third quantile of lengths of words in ordinary Slovak text. Similarly, if a sentence contains fewer words than spots reserved in the vector, the remaining spots of the vector are zero. An example of such vector is in Figure 1.

The final step is to evaluate paraphrasing using support vector machine. SVM is a binary classifier, which returns whether it is a paraphrase or not.

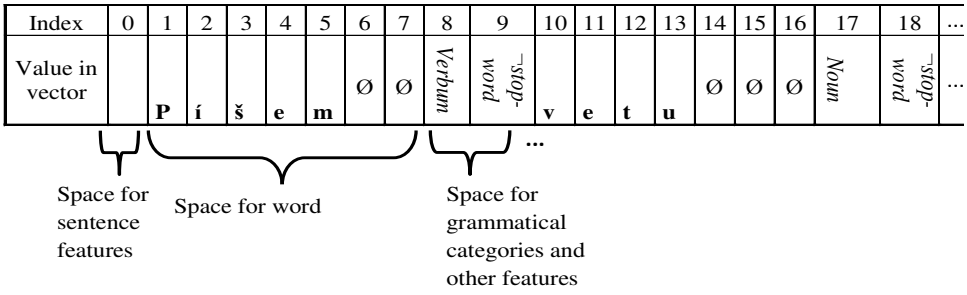


Figure 1.Example how two sentences and different features are represented by a vector. The text in the vector *"Píšem"*is one word and*"vetu"* is the second word.

3.2 POS tagger

POS tagger assigns part of speech to words from sentences. In its process it uses hidden Markov models and the Viterbi algorithm. Viterbi sequence of words in a sentence assigns the most likely sequence of morphological marks.

Viterbi algorithm requires a set of initial states probabilities (the probability that the sentence starts with the word, which is part of speech), transition matrix (expresses the probability of transitions between morphological markers) and generation of output probability matrix of part of speech *t* from the word *w*. Just not likelihood of generating output brands we replaced *tagWordPropability(1)* function, which determines the probability.

$$tagWordPropability(w,t) = \begin{cases} M_1[w,t], & if \exists M_1[w,t] \\ M_2[suffix(w),t], & if\ \exists M_2[suffix(w),t] \\ P(t), & else \end{cases} \quad (1)$$

Where *w* is a word and morphological tag (part of speech) *t* calculates the probability that the word belongs to the part of speech. $M_1$is a matrix containing the probability that a word *w* belongs the part of speech *t*. $M_2$is a matrix containing the probability that any word ending with *suffix(w)* is of speech *t*. Function *suffix(w)* returns the suffix of *w* - the last three letters of the word *w* and *P(t)* returns the probability of occurrence of speech *t*. The length of suffix was determined experimentally.

Matrices M_1, M_2 and function *P(t)* are created with a corpus of *r-mark* (Hand morphologically annotate corpus), which is part of the Slovak national corpus provided by the Language Institute of Ľ. Štúr Slovak Academy of Sciences.

4 Experiments

Our solution was verified on two data sets. The first dataset is a manually annotated corpus (r-mark), which was created with a help of the *Slovak Academy of Sciences*. In this corpus there are to each lexical unit assigned its grammatical categories. Currently the corpus contains 77 755 phrases consisting of 1 199 224 lexical units.

The second dataset was manually created by us and it is used for identification of paraphrases. It contains 1 350 pairs of sentences where half of these pairs contain two sentences that are a plagiary of each other. The rest of the pairs are those in which the first sentence is not a plagiary of the send sentence in the given pair.

Further in this section we describe two experiments. In the first experiment we evaluated the performance of our own POS tagger.

4.1 POS tagger performance

In this experiment we evaluated the performance of our own POS tagger that we use to identify morphological tags in the pre-processing phase. The overall accuracy of this POS tagger reached up to 95.35%. Table 2 shows the precision and recall for all speech and other morphological tags that our POS tagger identifies. The results were achieved on the r-mark dataset.

Table 2.Results that we achieved by using our POS taggerto identify morphologic tags.

Morphologic tag	Precision	Recall
noun	*94,04%*	*97,66%*
adjective	*94,42%*	*92,00%*
pronoun	*95,74%*	*98,41%*
numeral	*97,97%*	*91,16%*
verb	*95,57%*	*97,84%*
adverb	*91,67%*	*85,57%*
clutch	*92,39%*	*92,73%*
particle	*82,95%*	*82,07%*
interjection	*94,54%*	*56,05%*
participle	*98,30%*	*54,04%*
reflexive verb	*98,34%*	*98,70%*
conditional morfen	*99,91%*	*99,88%*
full stop, comma, question mark, exclamation point	*99,81%*	*99,80%*
other	*85,15%*	*29,92%*

4.2 Paraphrasing identification

In this experiment we evaluated the performance of the identification of paraphrases on the second dataset (the one that contains manually created pairs of sentences). In the comparison phase of our solution we used six different kernels with the SVM. Based on the experiments results as the most appropriate turned out to be Gaussian and linear kernel.

Overall, we obtained the best results (accuracy 90%) when using the Gaussian kernel with *complexity* ranging from 0.2 to 0.7.Other very good results, about 85 to 90%, were achieved by using a linear kernel with parameter *constant*set to value 2.0. The results are depicted in Figure 2.

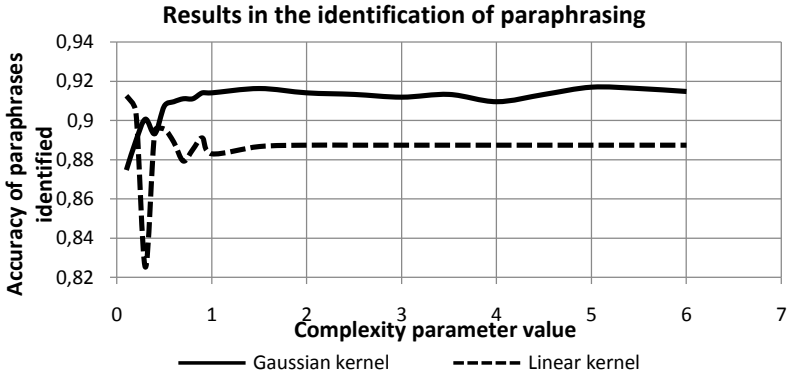


Figure 2.Results of paraphrasing identification based on complexity parameter.

5 Conclusion and future work

In this paper we have proposed a novel approach for identification of paraphrases in Slovak language. The most significant parts of this approach are our own POS tagger that we use in the pre-processing phase and the way how we represent and evaluate plagiarism between two sentences using SVM.

We managed to achieve 95.35% accuracy with our POS tagger in identifying morphological tags. We consider this as an important enhancement compared to the original approach that scored 87%.Overall, in the area of plagiarism identification we achieved good results when compared with the existing approaches that identify paraphrasing in English language. The best results in our experiments were obtained for the linear and Gaussian kernel and varied from 85% to 91.70%.

In the future we plan to integrate into our solution watching of information overlap that, in addition to overlapping phrases takes into account how much of the informational value of the sentence overlaps. For now, we begin to experiment with the calculation of entropy and tf-idf.

In real situations it may happen that the person who paraphrases divides the original sentence into several sentences. Therefore, the identification of a paraphrased sentence has its limitations. We think that this problem can be addressed by comparing one sentence to two or more sentences by compounding these sentences.

References

[1] Kozerva, Zornitsa. 2006. Paraphrase Identification on the Basis of Supervised Machine Learning Techniques. s.l. :*ACM, 2006.*

[2] Brockett, Chris a Dolan, William. 2005. Support Vector Machines for Paraphrase Identification and Corpus Construction. s.l. :*Natural Language Processing Group Microsoft Research, 2005*.

[3] Chitra, A. 2010. Paraphrase Identification using Machine Learning Techniques. s.l. : ACM, 2010. Zv. *RECENT ADVANCES in NETWORKING, VLSI and SIGNAL PROCESSING.* ISBN: 978-960-474-162-5.

[4] Lintean, Michain. 2009. Paraphrase Identification Using Weighted Dependencies and Word Semantics. Informatica. s.l. :*ACM, 2009*.

[5] Microsoft. WCF Extensibility Guidance. [Online] Microsoft. [Date: 25. 8 2013.] *http://msdn.microsoft.com/en-us/library/gg132853.aspx.*

[6] Nemirovsky, Danil a Dobrynin, Vladimir. 2008. Word importance discrimination using context. s.l. :*TREC, 2008*.

[7] Sangeetha, R. a Kalpana, B. 2010. Optimizing the kernel selection for support vector machines using performance measures. *New York : ACM, 2010*. ISBN: 978-1-4503-0194-7.

[8] Taria, Hirothosi a Haruno, Masahiko. 1999. Feature selection in SVM text categorization. s.l. :*ACM, AAAI, 1999*. ISBN:0-262-51106-1.

[9] Vapnik, V. N. 1995. The Nature of Statistical Learning Theory. *New York : Springer, 1995*.

Higher Paraphrasing Identification in Slovak Language with SVM

Gajdos Jozef* and Kucecka Tomas

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkoviova2, 842 16 Bratislava, Slovakia
xgajdos@is.stuba.sk
http://www.fiit.stuba.sk/

Abstract. This paper deals with the identification of higher paraphrasing in text documents written in the Slovak language. By higher paraphrasing we mean that the paraphrased text was created by using more advanced methods, for instance synonyms replacement. In general there are several ways to identify paraphrasing. In this work we present our own approach that uses support vector machine (SVM) and contains several specifics in the pre-processing phase regarding Slovak language. Our solution takes into account the Slovak language characteristics and use of the currently available methods and dictionaries. The main result of our work is the created POS tagger for Slovak language and the way how two sentences and their properties are represented in a vector and then compared on similarity. Conducted several experiments show that our proposed approach works well.

Key words: paraphrasing identification, part of speech identification

1 Introduction

Today large volume of data can be found in an electronic version that is usually available online on the Internet. Most of these data are texts written in natural language. Users often have to search for relevant content on the web based on keywords, phrases or by selecting different categories. Being able to query such data and detect similar content within different texts is therefore crucial these days. Such similarity detection on higher level is however an open problem. To the open problems in detecting similarity belongs the identification of paraphrasing at a higher level. By a paraphrasing on a higher level we refer to the combination of changes in word order, inserting or removing words from sentences and using synonyms.

Paraphrasing is basically a way of saying something in different words while the original idea and its importance during paraphrasing do not change. There

* Please note that the LNCS Editorial assumes that all authors have used the western naming convention, with given names preceding surnames. This determines the structure of the names in the running heads and the author index.

are many ways how to paraphrase text in Slovak language. Moreover, Slovak language belongs to the group of inflecting languages. Core vocabulary consists of 60 000 words, has a complex syntax, morphology, and is quite ambiguous.

Identification of paraphrasing in natural language can be helpful in identifying plagiarism or detecting topic similar documents. Such identification can be for instance useful when clustering text data or in case we perform knowledge summarization.

2 Related work

In [1] Autors identify paraphrase using the word weighted semantics. The mere paraphrasing determined as the ratio of the similarities and differences of the two sentences. Similarity and difference is calculated based on dependencies between words in sentences. For calculating similarity in sentences valued most similar words are paired, and calculate the weighted average of them. Divergence is calculated with unmatched words. paraphrasing is identified by comparing the threshold with a share similarities and differences. This method is about 71%, but setting the threshold depends on the selected corpus.

In papers [2], [3] and [4] authors introduce an approach to identification of paraphrasing through using the machine learning. All proposed methods were designed for the English language and tested on Microsoft corpus, which has marked paraphrased sentences. The granularity of identification was on sentence level.

In paper [2] are described experiments with machine learning algorithms, SVM and k-nearest neighbour. As a distance measure authors used the normalized Manhattan distance and maximum entropy. Input to the machine learning algorithms were pairs of sentences with the following features: the count of matching words between sentences to the total number of words, overlap of 4-skip-gramsbetween sentences, overlap of sequences in sentences found by LCS (longest common subsequence) algorithm, semantic similarity between words and containing its own noun received flag 1, else the flag was 0.

In several experiments authors investigated the effect of these features (LCS, n-grams and skip-grams, semantic overlap, etc.). The best results using SVM achieved precission 70.43% and recall 74.12%. These experiments showed that the most appropriate technique to identify the paraphrasing is the SVM.

In paper [3] it is also stated that the SVM is the most appropriate approach to the identification of paraphrasing. Input to the SVM represented: string similarity, count of matching words, Levenshtein distance between words, stems of words, marking semantically similar words (synonyms).

The authors used linear kernel with the following parameters - *constant* 10^{-3} and *complexity* 0.5. The results showed that the overlap of phrases and labelling of semantically similar words significant impact on the precision.

In work [4] the proposed solution also used the SVM where the input to the learning algorithm represented a pair of sentences with the following features:

- ☐ Overlap between sentences determined by the skip-gram method.

- Overlap, which is determined by the LCS algorithm calculated as the ratio of the number of words found by LCS to the number of words in each sentence.
- The semantic similarity of verbs and nouns in sentences. Semantic similarity was determined by the semantic content of the word, the likelihood of its occurrence and information from WordNet.
- Similarity of numeric attributes in sentences.
- Presence of own nouns in a sentence.
- In experiments authors used the linear and the Gaussian kernel function. The authors state that the addition of lexical and semantic information increased accuracy of plagiarism identification. The resulting precision is amounted to 70%.

3 Our solution

Our solution is intended to identify paraphrasing based on SVM approach that was used in the existing work that we described in the previous section. The main difference is that we focus on Slovak language and threfore we take into account the specifics of this language, which can be seen mainly in its pre-processing stages. We decided to use support vector machines, because the paper [3] shows their relevance in identifying paraphrasing in natural language.

The basic idea of our approach is following. Input to our method is a plain text extracted from a textual document. In the first step the text of documents is chunked on sentences. Then we determine various properties of the sentence as its length and modality. Individual sentences are divided into words and these words are assigned to individual properties. The words are assigned morphological tags using our own approach called POS tagger. Lemma and a common synonym is determined for every word and the word is marked if it is a stop-word. In the second step the two textual documents are compared on sentence level which means that we detect paraphrasing within all pairs of sentences from the two documents. In this comparison process the SVM approach is used. For every pair of sentences we determine their overlap on semantic, morphological and lexical level through: LCS algorithm, 3-grams where 1-gram is one word, 4-skip-grams where 1-gram is one word, simple overlap-count of words equals a minimum count of words in sentences.

In the following subsections we give a detail description how our method works.

3.1 POS tagger

POS tagger assigns part of speech to words from sentences. In its process it uses hidden Markov models and the Viterbi algorithm. Viterbi sequence of words in a sentence assigns the most likely sequence of morphological marks.

Viterbi algorithm requires a set of initial states probabilities (the probability that the sentence starts with the word, which is part of speech), transition matrix (expresses the probability of transitions between morphological markers) and generation of output probability matrix of part of speech t from the word w. Just not likelihood of generating output brands we replaced $tagWordPropability$ function, which determines the probability.

$$tagWordPropability(w,t) = \begin{cases} M_1[w,t] & \text{if } \exists M_1[w,t] \\ M_2[suffix(w),t] & \text{if } \exists M_2[suffix(w),t] \\ P(t) & \text{else} \end{cases} \quad (1)$$

Where w is a word and morphological tag (part of speech) t calculates the probability that the word belongs to the part of speech. M_1 is a matrix containing the probability that a word w belongs the part of speech t. M_2 is a matrix containing the probability that any word ending with $suffix(w)$ is of speech t. Function $suffix(w)$ returns the suffix of w - the last three letters of the word w and $P(t)$ returns the probability of occurrence of speech t. The length of suffix was determined experimentally.

Matrices M_1, M_2 and function $P(t)$ are created with a corpus of *r-mark* (Hand morphologically annotate corpus), which is part of the Slovak national corpus provided by the Language Institute of . tr Slovak Academy of Sciences.

3.2 Vector representation

The key step of our approach is the way how we represent pair of sentences with their properties in a vector. This vector is then handled by the SVM. We have proposed several vector representations for SVM.

The first representation is called *Basic*. The first part of the vector are the normalized features that the two sentences overlap. These features are followed by the two sentences. In the first part of each sentence are the properties of a sentence, followed by the words of the sentence. Individual words together with their properties inserted one character after the other, as in the original sentences. For every word there is a fixed number of items reserved in the vector. If a word has fewer letters than these items, the remaining items are filled with zeros. If a word has more letters than these items, the rest of the letters will be discarded. Length of these items is third quantile of lengths of words in ordinary Slovak text. Similarly, if a sentence contains fewer words than spots reserved in the vector, the remaining spots of the vector are zero. An example of such vector is in Figure 1.

The second representation which we call *Enriched vector model document* consists of two parts - overlapping features and aggregated tf-idf vector representation. This approach is based on the vector space model. The second part of the vector is formed by merging two vector representations of sentences using an aggregation function ω as follows

$$v_i = \omega(tf_{1,i}, tf_{2,i}).idf_i \ . \quad (2)$$

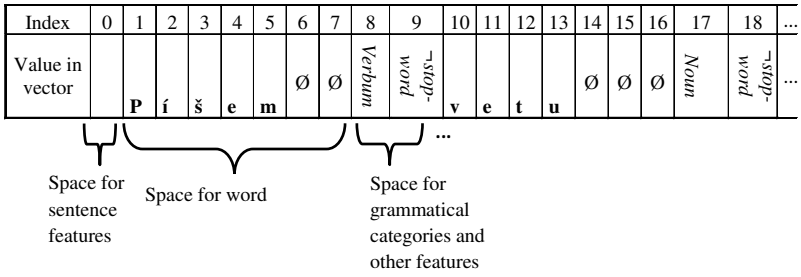


Fig. 1. Example how two sentences and different features are represented by a vector. The text in the vector *"Pisem"* (english *"I write"*) is one word and *"vetu"* (english *"sentence"*) is the second word.

where v_i is the value in the second part of the vector, ω is an aggregate function, $tf_{1,i}$ is the i-th frequency of terms belonging to the first sentence and the idf_i is the inverse frequency of terms belonging to the i-th term. The aggregation functions was defined by us as:

$$\omega(tf_{1,i}, tf_{2,i}) = \beta.min(tf_{1,i}, tf_{2,i}) + (1 - \beta).max(tf_{1,i}, tf_{2,i}) \; . \tag{3}$$

The final step of our approach is than to evaluate paraphrasing using a binary classifier SVM that says whether one of the two sentences in a pair is or is not a paraphrase of the other sentence in the same pair.

4 Experiments

Our solution was verified on two data sets. The first dataset is a manually annotated corpus (*r-mark*) that was created by *Slovak Academy of Sciences*. In the corpus are to each lexical unit assigned its grammatical categories. Currently the corpus contains 77 755 phrases consisting of 1 199 224 lexical units.

The second dataset was manually created by us and it is used for identification of paraphrases. It is made of fiction and purified from grammatical errors. It contains 1 350 pairs of sentences where half of these pairs contain two sentences that are a plagiary of each other. The rest of the pairs are those in which the first sentence is not a plagiary of the send sentence in the given pair.

Further in this section we describe two experiments. In the first experiment we evaluated the performance of our own POS tagger and the second experiment shows results of paraphrasing identification.

4.1 POS tagger performance

In this experiment we evaluated the performance of our own POS tagger that we use to identify morphological tags in the pre-processing phase. The overall

accuracy of this POS tagger reached up to 95.35%. Table 1 shows the precision and recall for all speech and other morphological tags that our POS tagger identifies. The results were achieved on the r-mark dataset.

Table 1. Results that we achieved by using our POS tagger to identify morphologic tags.

Morphologic tag	Precision	Recall
noun	*94.04%*	*97.66%*
adjective	*94.42%*	*92.00%*
pronoun	*95.74%*	*98.41%*
numeral	*97.97%*	*91.16%*
verb	*95.57%*	*97.84%*
adverb	*91.67%*	*85.57%*
clutch	*92.39%*	*92.73%*
particle	*82.95%*	*82.07%*
interjection	*94.54%*	*56.05%*
participle	*98.30%*	*54.04%*
reflexive verb	*98.34%*	*98.70%*
conditional morfen	*99.91%*	*99.88%*
full stop comma,question mark, exclamation point	*99.81%*	*99.80%*
other	*85.15%*	*29.92%*

4.2 Paraphrasing identification

In this experiment we evaluated the performance of the identification of paraphrases on the second dataset (the one that contains manually created pairs of sentences). In the comparison phase of our solution we used six different kernels with the SVM. Based on the experiments results as the most appropriate turned out to be Gaussian and linear kernel.

Overall, we used *Basic* vector representation, we obtained the best results (accuracy 90%) when using the Gaussian kernel with *complexity* ranging from 0.2 to 0.7. Other very good results, about 85 to 90%, were achieved by using a linear kernel with parameter *constant* set to value 2.0. The results are depicted in Figure 2.

The *Enriched vector model document* obtained the best results (accuracy 94.33%) when using the Gaussian kernel with *complexity* ranging from 0.4 to 0.5. Other very good results, 94% ,were achieved by a linear kernel with parameter *constant* set to value 1.0. The results are depicted in Figure 3.

5 Conclusion and future work

In this paper we have proposed a novel approach for identification of paraphrases in Slovak language. The most important two parts of this approach is our own

POS tagger that we use in the pre-processing phase and the way how we represent and evaluate plagiarism between two sentences using SVM.

We managed to achieve 95.35% accuracy with our POS tagger in identifying morphological tags. We consider this as an important enhancement compared to the original approach that scored 87%. Overall, in the area of plagiarism identification we achieved good results when we compare our work with the existing approaches that identify paraphrasing in English language. The best results in our experiments that we obtained in the detection of paraphrased sentences was with using the *Enriched vector model document* representation together with linear and Gaussian kernel. The results varied from 94% to 94.33%.

In the future we plan to integrate into our solution watching of information overlap that, in addition to overlapping phrases takes into account how much of the informational value of the sentence overlaps. For now, we begin to experiment with the calculation of entropy.

In real situations it may happen that the person who paraphrases divides the original sentence into several sentences. Therefore, the identification of a paraphrased sentence has its limitations. We think that this problem can be addressed by comparing one sentence to two or more sentences by compounding these sentences.

References

1. Lintean, M. C., Rus, V.:Paraphrase Identification Using Weighted Dependencies and Word Semantics. Informatica (Slovenia), vol. 34, Slovenian Society Informatika, no. 1, pp.19-28, (2010)
2. Kozerva, Zornitsa.: Paraphrase Identification on the Basis of Supervised Machine Learning Techniques. s.l. :ACM (2006)
3. Brockett, Chris a Dolan, William.: Support Vector Machines for Paraphrase Identification and Corpus Construction. s.l. :Natural Language Processing Group Microsoft Research, (2005)
4. Chitra, A.: Paraphrase Identification using Machine Learning Techniques. s.l. : ACM, 2010. Zv. RECENT ADVANCES in NETWORKING, VLSI and SIGNAL PROCESSING. ISBN: 978-960-474-162-5, (2010)
5. Lintean, Michain.: Paraphrase Identification Using Weighted Dependencies and Word Semantics.Informatica. s.l. :ACM, (2009)
6. Microsoft. WCF Extensibility Guidance, `http://msdn.microsoft.com/en-us/library/gg132853.aspx`
7. Nemirovsky, Danil a Dobrynin, Vladimir.: Word importance discrimination using context. s.l. :TREC, (2008)
8. Sangeetha, R. a Kalpana, B.: Optimizing the kernel selection for support vector machines using performance measures. New York : ACM, 2010. ISBN: 978-1-4503-0194-7, (2010)
9. Taria, Hirothosi a Haruno, Masahiko: Feature selection in SVM text categorization. s.l. :ACM, AAAI, 1999. ISBN:0-262-51106-1 (1999)
10. Vapnik, V. N.: The Nature of Statistical Learning Theory. New York : Springer (1995)

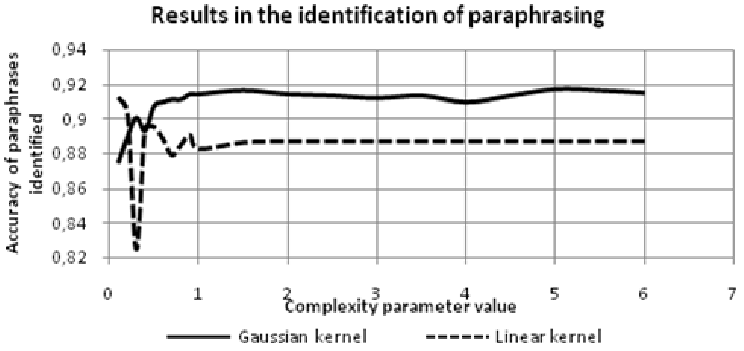


Fig. 2. Based profile paraphrasing identification based on complexity parameter.

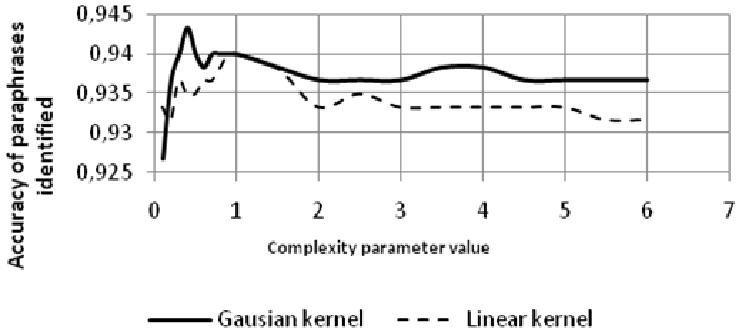


Fig. 3. Enriched vector model document profile paraphrasing identification based on complexity parameter.