Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Vasja Laharnar

Kontekstualno ujemanje in iskanje na modelu spletne oglasne deske

diplomsko delo
na univerzitetnem študiju

Mentor: izr. prof. dr. Marko Bajec

Ljubljana, 2013

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01906/2013
Datum: 06.03.2013

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **VASJA LAHARNAR**

Naslov: **KONTEKSTUALNO UJEMANJE IN ISKANJE NA MODELU SPLETNE OGLASNE DESKE**

CONTEXTUAL MATCHING AND SEARCH USING WEB-BASED BULLETIN BOARD MODEL

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Spletne oglasne deske so specializirani iskalniki, ki lahko namesto dokumentov (npr. spletne strani, slike, besedila), indeksirajo uporabnike sistema. Primer so socialni iskalniki, ki ne vrnejo neposrednega odgovora, ampak se poizvedba pošlje uporabniku, ki ga sistem spozna za relevantnega in nato le ta odgovori. Takšni iskalniki morajo torej bolj upoštevati uporabniške profile in njihove kontekste.

V okviru diplomske naloge naj kandidat izdela spletno storitev, ki bo omogočala objavljanje besedil in njihovo iskanje. Pri tem naj kandidat preuči korake procesiranja besedil za potrebe splošnega indeksiranja. Poleg tega naj pri implementaciji iskanja in primerjanja besedil upošteva tudi semantične podatke, k jih pridobi iz besedil ali profilov uporabnikov. Nazadnje naj izdelano storitev testira na domeni študijskih praks, kjer v sistemu sodelujejo profesorji, študenti in podjetja.

Mentor:

izr. prof. dr. Marko Bajec

Dekan:

prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Vasja Laharnar, z vpisno številko **63040086**, sem avtor diplomskega dela z naslovom:

Kontekstualno ujemanje in iskanje na modelu spletne oglasne deske

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom izr. prof. dr. Marka Bajca,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 22. marca 2013 Podpis avtorja:

Zahvala družini, Stefanie Schneider, SWiM Bildung, izr. prof. dr. Marku Bajcu in uni. dipl. ing. Slavku Žitniku za izkazano pomoč, zaupanje in znanje!

Kazalo

Seznam uporabljenih kratic in simbolov

BOW Bag of Words
BSON Binary JSON
HTTP Hypertext Transfer Protocol
IMAP Internet Message Access Protocol
JSON JavaScript Object Notation
LAMP/WAMP Linux/Windows, Apache, MySQL, PHP/Perl/Python
LCS Longest Common Subsequence
MVC Model–View–Controller
NLP Natural Language Processing
PHP PHP: Hypertext Preprocessor
POP3 Post Office Protocol version 3
RDBMS Relational Database Management System
SOAP Simple Object Access Protocol
SQL Structured Query Language
SVM Support Vector Machine
URL Uniform Resource Locator
VSM Vector Space Model
WAF Web Application Framework
WSD Word-sense Disambiguation
XML Extensible Markup Language
XML-RPC XML Remote Procedure Call
XML-TEI XML Text Encoding Initiative P5

Povzetek

Oglasne deske so že vrsto let znan in uveljavljen način za posredovanje informacij različnim skupinam ljudi. Izpopolnjen princip delovanja oglasne deske smo želeli prenesti na svetovni splet, pri čemer smo se morali najprej seznaniti z osnovnimi nalogami obdelave naravnega jezika. Med drugim smo izvedli tokenizacijo vsebine objav, lematizacijo dobljenih pojavnic in zgradili strukturo semantično podobnih besed v nerelacijski podatkovni bazi. Besedila smo tudi klasificirali s pomočjo naivnega Bayesovega klasifikatorja in tako omogočili kontekstualno združevanje objav. Implementiran sistem iskanja in združevanja smo uspešno testirali na domeni pripravništva v obliki spletne storitve, ki temelji na objavljanju vsebin tako s strani izobraževalnih ustanov kot tudi podjetij in organizacij.

Ključne besede:

obdelava naravnega jezika, semantična podobnost, lematizacija, klasifikacija, naivni Bayesov klasifikator, iskanje informacij, iskanje, ujemanje

Abstract

Bulletin boards are, for many years, a well-known and established way of providing information to different groups of people. We tried to present an improved internet-based form of a bulletin board where we had to first inform ourselves about the basic natural language processing tasks. Among other things, we performed tokenization of the published content, lemmatization of the obtained tokens and also built a structure of semantically similar words in a non-relational database. We also classified the texts using a naive Bayesian classifier, thus allowing the contextual matching of the posts. We successfully tested the implemented search and match systems on an internship problem domain in the shape of a web service based on the supplied content from educational institutions as well as companies and organizations.

Keywords:

natural language processing, semantic similarity, lemmatization, classification, Naive Bayes classifier, information retrieval, search, matching

Poglavje 1

Uvod

Diplomsko delo se ukvarja z nalogami obdelave naravnega jezika v namene iskanja informacij. Na modelu spletne oglasne deske analizira problematiko funkcij iskanja in ujemanja sporočil ter skuša iz ugotovitev tudi implementirati ključne metode in opisati morebitne izboljšave.

1.1 Pobuda

Glede na hiter življenjski slog ljudi in ob poplavi informacij, ki jih je prinesel internet, je v današnjem razvitem svetu težnja vsakega posameznika ta, da se do svojih potreb dokoplje hitro in učinkovito. Hitro v smislu čimkrajšega trajanja v času in učinkovito predvsem v smislu pridobivanja želenega s čimmanjšim vloženim naporom. Slehernik se torej namensko poslužuje vseh mogočih bližnjic, že izkušenih storitev, najbolj direktnih poti, da bi v čimmanjšem številu korakov prišel do tistega, kar išče. Čas je torej dragocen in za posameznika je pomembno predvsem to, da ima na njegovo porabo sam kar največ vpliva. Ljudje radi vzamemo stvari v svoje roke in smo neradi odvisni od drugih dejavnikov ali motečih faktorjev. Lahko bi torej sklepali, da so ključen vmesni člen med iskanim in najdenim predvsem orodja in storitve, ki so uporabniku dostopna in ki mu, v množici sorodnih oziroma na nek način povezanih stvari, pomagajo najti želeno.

Podatek statističnega urada Evropske komisije Eurostat [3], da je v letu 2011 znotraj EU-27 kar 73 % gospodinjstev imelo dostop do interneta, nam daje slutiti, da je raba interneta znotraj Evropske unije precej razširjena. V Sloveniji je po podatkih Statističnega urada Republike Slovenije [1] v prvem četrtletju 2012 imelo dostop do interneta 74 % gospodinjstev. Raba medmrežja je torej postala že del vsakdanjika in vse več storitev se seli na svetovni splet. Podatki Eurostat [3] kažejo, da je v letu 2011 58 % uporabnikov interneta v starosti 16-74 let v EU-27 v zadnjih 12 mesecih preko interneta naročilo blago ali storitve. Najvišji deleži so opaženi v Veliki Britaniji (82 %), na Danskem in v Nemčiji (obe 77 %) ter na Švedskem (75 %).

Medmrežje je skoraj neomejen vir najrazličnejših informacij. Spletni iskalniki so nam pri tem v veliko pomoč. Obstaja pa tudi nemalo spletnih storitev, ki so osredotočene ali specializirane za reševanje bolj specifičnih tematik. Tukaj imamo v mislih, na primer, spletne oglase, nepremičninske agencije, primerjalnike cen, borzo dela ipd. Slednja je namenjena povečini iskalcem zaposlitve oziroma ljudem na trgu dela. Študentom in dijakom so na voljo študentski servisi, ki se ubadajo s podobno tematiko. Mladim omogočajo nabiranje izkušenj in dodaten zaslužek. Žal pa tega ne moremo enačiti s pripravništvom, saj je študentsko delo le občasno ali začasno delo, ki ga študent, dijak ali druga upravičena oseba opravlja preko pooblaščene organizacije na podlagi napotnice (Uradni list RS, št. 42/2002, 103/2007) in torej osnovni namen ni usposabljanje ali izobraževanje.

Praksa oziroma praktično usposabljanje je drugi termin za široko uporabljena izraza “pripravništvo” ali “praktično usposabljanje” (angl. internship). Pripravništvo je v tujini dodobra uveljavljen način, kako lahko mladi pridobivajo znanje in izkušnje na področjih, ki jih zanimajo. Center RS za mobilnost in evropske programe izobraževanja in usposabljanja vsako leto objavlja razpise, ki nudijo različne priložnosti na področju izobraževanja, poučevanja, usposabljanja, razvoja in sodelovanja v mednarodnem okolju. Tukaj se cilja predvsem na vključevanje slovenskih študentov v programe usposabljanja v tujini ali pa sodelovanje slovenskih podjetij s tujimi študenti,

kar vpliva na prilagajanje potrebam evropskega trga dela, žal pa neposredno ne zbližuje domačega gospodarstva in izobraževalnoraziskovalnih ustanov.

1.2 Namen

Lahko si torej omislimo spletni projekt namenjen zbliževanju industrije oziroma gospodarstva z izobraževalnimi ustanovami, ki bi omogočal objavljanje idej, interesov, pobud ipd. Ključno pri vsem tem bi bila možnost iskanja objavljene vsebine in tudi samodejno združevanje relevantno ujemajočih se objav. Pomembni sta torej predvsem funkciji iskanja in združevanja ali ujemanja. Zagotovilo natančnosti oziroma uspešnosti teh procesov je tudi pravilna interpretacija ali analiza uporabnikove poizvedbe. Iskalni mehanizem si mora prizadevati, da v največji možni meri poistoveti rezultate analize z namero uporabnika.

1.3 Cilji

Raziskati želimo osnovne naloge obdelave naravnega jezika, kot so tokenizacija, blokiranje, lematizacija ipd. Obdelanim besedam želimo pripisati tudi semantično podobne besede, predvideti njihovo indeksno strukturo in jih uvrstiti v primeren razred na osnovi klasifikacije. Ugotovljeno skušamo nato za dosego boljših rezultatov upoštevati pri implementaciji funkcij iskanja ter ujemanja sporočil.

1.4 Struktura diplomske

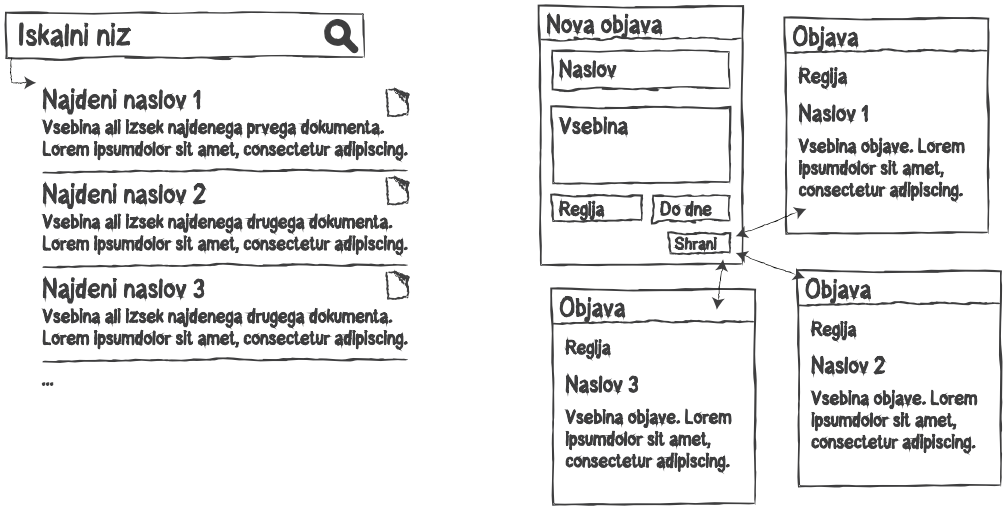
Diplomska naloga je v grobem razdeljena na dva dela. V prvem delu se posvečamo analizi nalog obdelave naravnega jezika, strukturiranju obdelanih pojavnic ter glavnim lastnostnim relevantnega iskanja ter ujemanja besedilnih dokumentov. Drugi del je posvečen zasnovi in izdelavi rešitve, ki omenja uporabljene tehnologije in orodja, opisuje implementacijo osnovnih metod

iskanja in ujemanja ter nadaljnjih možnosti razvijanja oziroma izboljšav.

Poglavje 2

Opis problema

Naša spletna storitev v principu deluje kot oglaševalna deska ali pano. Bistvo panoja je, da se nanj pripenja informacije namenjene določeni skupini ljudi. Spletni projekt je namenjen sodelovanju oziroma zbliževanju industrije oziroma gospodarstva z izobraževalnimi ustanovami, torej ima naš projekt dva glavna akterja. Izmenjava informacij poteka obojestransko, torej lahko, na primer, predstavnik nekega podjetja na pano obesi objavo za vse zainteresirane predstavnike različnih izobraževalnih ustanov in po drugi strani lahko nek profesor na pano objavi, da študentje določenih smeri iščejo prakso ali pripravništvo. Objave vsebujejo vse osnovne informacije kot so vsebina in obseg dela, lokacija in podobno. Sistem mora biti zmožen tudi sam združiti objave obeh omenjenih akterjev, seveda če se te ujemajo v čim več skupnih točkah. Potrebno bi bilo torej izluščiti ključne podatke objav in najti morebitne ujemajoče se pare obeh vpletenih akterjev. V mislih moramo imeti tudi starostno lestvico obiskovalcev našega spletnega mesta oziroma ciljne skupine. Ta vključuje tudi starejše osebe, torej mora biti uporabniška izkušnja oziroma uporabniški vmesnik prilagojen tudi za starejše uporabnike. V ta namen želimo vsebino objave ohraniti v večji meri kot celoto in je ne razčleniti na manjše entitete, s pomočjo katerih bi lahko izvajali tudi filtriranje objav. Edini ločeni, a ne obvezni, podatek je regija v državi, na katero se objava osredotoča. Uporabniško izkušnjo želimo torej ohraniti preprosto in enostavno.



Slika 2.1: Skica želenega delovanja metod iskanja in ujemanja.

Projekt v osnovi obsega ustvarjanje uporabniških profilov obeh vrst akterjev, možnost dodajanja in urejanja objav ter zmožnost iskanja in samodejnega združevanja do neke mere ujemajočih se objav. Iskanje poteka po principu prikazanem na skici 2.1. Leva stran skice prikazuje vnos iskalnega niza in odziv sistema, torej rezultate iskanja predstavljene v obliki seznama z naslovi in vsebino najdenih dokumentov. Desna stran skicira delovanje funkcije ujemanja. Za vnos nove objave je potrebno izpolniti obrazec, pri čemer sta obvezna podatka naslov in sama vsebina objave. Ko je nova objava vnesena, se obdelana vsebina primerja z obstoječimi objavami. Dokumenti, ki se novo vneseni objavi pomensko dovolj dobro prilegajo, se pripnejo kot predlogi k objavi.

Poglavje 3

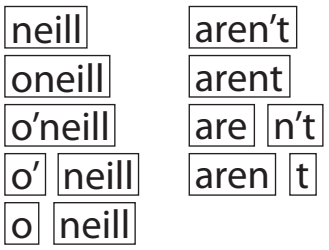
Analiza

3.1 Obdelava naravnega jezika

Prvi korak pri reševanju problema smo nekako že omenili. Potrebno bi bilo pregledati vsebino vnesene objave in iz nje izluščiti ključne podatke. Podatki so v našem primeru besede, proces obdelovanja besed pa spada v področje obdelave naravnega jezika (*angl. natural language processing, NLP*). To področje je zelo obsežno, zato se bomo v naslednjih podpoglavjih dotaknili le bistvenih pojmov, ki jih je dobro poznati in upoštevati v namene izboljšanja rezultatov iskanja.

3.1.1 Tokenizacija

Tokenizacija (*angl. tokenization*) po [13] razdeli besedilo na besede ali pojavnice (*angl. token*) in je pomembna, ker predstavlja temeljno fazo računalniške obdelave besedil in so od nje odvisni nadaljnji postopki. Operacijo tokenizacije nad besedilom dokumenta v slovenskem jeziku lahko v večji meri poenotimo z regularnim izrazom (*angl. regular expression*), ki iz besedila izloči samo besede, ali kakšno drugo metodo. Obstaja pa tudi verjetnost, da so določene besede v besedilu okrajšane, zaključene ali s piko ali kako drugače, in je za te primere potrebno dodatno pravilo. Raba regularnih izrazov za tokenizacijo slovenskega jezika je dokaj enostaven in relativno korekten način.



Slika 3.1: Tokenizacija dveh angleških besed z več možnimi rešitvami.

V primeru angleškega jezika pa je situacija bolj zapletena, saj se apostróf ali opuščaj uporablja na primer za opuščanje črk ali pa označevanje svojine. Dva lepa primera povzeta po [6] sta besedi "O'Neill" in "aren't". Slednjo lahko tokeniziramo na štiri načine, prvo pa celo na pet načinov, kar je prikazano na sliki 3.1. Odločimo se lahko za katerokoli varianto, važno je le, da tokeniziramo na enak način tako besedilni dokument kot samo poizvedbo oz. primerjani dokument.

3.1.2 Blokiranje

Blokiranje izloči iz nadaljnjih postopkov obdelave besedilnih dokumentov besede, ki so pomensko šibke [13]. Take besede so na primer pomožni glagoli, predlogi, vezniki, nikalnice ipd. Na medmrežju obstajajo že izdelani seznami blokiranih besed (*angl. stop words*), ki se seveda razlikujejo za različne svetovne jezike. Na spletni strani Instituta Jožef Stefan je obstajal seznam blokiranih besed za slovenski jezik[1], ki pa žal ni več direktno dostopen. Na srečo pa se s pomočjo spletnega arhiva lahko do njega vseeno dokopljemo[2]. Na seznamu med drugim zasledimo tudi z besedo zapisana števila. Verjetno gre presoditi, če je smotrno tako z besedo kot s številko zapisana števila blokirati. Števila se namreč lahko pojavijo v samem nazivu podjetja ali usta-

[1] http://nl.ijs.si/GNUsl/lex/stop/

[2] http://web.archive.org/web/20070701081435/http://nl.ijs.si/GNUsl/lex/stop/mtestop-sl.wfl

nove. Poizvedba z iskalnim nizom "Val 202" bi nas lahko zmotno popeljala na Obalo. Tudi datum je lahko pomemben podatek. Seznam je seveda prilagodljiv, odločiti se moramo katere besede so v našem projektu šibkejšega pomena.

3.1.3 Krnjenje

Po [7] je krnjenje (*angl. stemming*) postopek obdelave besedil z algoritmi za krnjenje, s katerim avtomatsko določamo indeksne izraze, primerne za opisovanje vsebine dokumentov. Po [8] lahko povzamemo, da pri krnjenju poskušamo najti niz znakov, imenujemo ga *krn*, ki lahko predstavlja vse oblike neke besede in istočasno to besedo loči od vseh ostalih. Pogosto krn ustreza korenu besede, vendar ne nujno, zato raje govorimo o krnih. Algoritmi za krnjenje so predvsem pravila, kako v iteracijah ali določenem številu korakov priti do krna. Ponavadi so prirejeni za določen jezik in tako niso uspešni oziroma uporabni za nek drug jezik. Svetovni jeziki se med seboj razlikujejo po številu sklonov, številu sklanjatev itd. Zaradi bogate morfologije slovenskega jezika so algoritmi za krnjenje slovenščine bolj kompleksni. Štirje bolj odmevni algoritmi, ki so bili izdelani za slovenski jezik so:

- Preprost algoritem, izdelan na Medicinski fakulteti v Ljubljani,
- Algoritem Mirka Popoviča,
- Generični algoritem,
- Optimalni algoritem.

Razvoju optimalnega algoritma avtorja Dr. Jureta Dimca sta botrovali predvsem dve dejstvi. Tekom razvoja obstoječi algoritmi za slovenščino so premočno krnili in poleg tega so tedanji algoritmi modelirali splošen jezik in niso bili prilagojeni izrazju v strokovnih podjezikih. Algoritem deluje v treh korakih:

- rezanje končnice,

1. korak (rezanje končnic)
Pravila: ec → ‘c’; en → ’’; ega → ’’:
konec → konc; končen → konč; končnega → končn

2. korak (obdelava soglasniških parov)
Pravila: čn → č; nč → nc:
konč → konc; končn → konč → konc

konec → konc
končen → konc
končnega → konc

Slika 3.2: Primer krnjenja besednih oblik konec, končen in končnega. Krn, ki nastane po rezanju končnic in transformaciji soglasniških parov, lahko predstavlja vse tri oblike.

- obdelava soglasniških parov na koncu krna,
- pravila za popravljanje.

Povzeto po [10] je na sliki 3.2 prikazan primer krnjenja besed konec, končen in končnega. Krnjenje bi torej lahko upoštevali pri gradnji indeksne tabele. Ključne pojavnice besedilnih dokumentov bi pretvorili v krne, kakor tudi besede iskalnega niza poizvedbe, in na tak način izboljšali rezultate iskanja.

3.1.4 Lematizacija

Lematizacija (*angl. lemmatisation, lemmatization*) je po [11] postopek pretvarjanja besede v njeno lemo, tj. nevtralno oziroma slovarsko obliko te besede. Uporablja se predvsem na področju klasifikacije ter iskanja besedil. Lematizacija je podobna krnjenju. Razlika je v tem, da proces krnjenja poteka brez vedenja konteksta in zato ne razločuje med besedami, ki imajo lahko tudi več pomenov. Lematizacija torej lahko vsebuje tudi kompleksna opravila, kot je razumevanje konteksta, zaradi česar je njena implementacija za nov jezik zapletena in dolgotrajna naloga. Algoritme krnjenja je navadno lažje implementirati in delo opravijo hitreje, kar izkoriščajo aplikacije, pri

Izvorna vsebina	**Lematizirana vsebina**
Grška usoda, prihodnost vseh nas	Grški usoda, prihodnost ves jaz
Več tisoč Portugalcev išče boljšo prihodnost	Veliko tisoč Portugalec iskati dober prihodnost
V Berlinu odprli razširjen judovski muzej	V Berlin odpreti razširjen judovski muzej
Poglejte si novi video skupine Coldplay	Pogledati se nov video skupina Coldplay

Tabela 3.1: Lematizacija vsebine dokumentov.

katerih zadošča nižji nivo natančnosti. Z lematizacijo slovenskega jezika se med drugim ukvarja ekipa na Institutu Jožef Stefan (IJS) v Ljubljani. Ekipa razvija večjezikovno odportokodno lematizacijsko orodje poimenovano LemmaGen[3]. Podpira različne platforme in obstaja v različnih implementacijah (C++, C++.Net, Python, C#.Net). Tudi s pomočjo lematizacije bi lahko pojavnice besedilnih dokumentov in poizvedb lematizirali in izboljšali rezultate. Za primer lahko lematiziramo vsebino nekaj izmišljenih dokumentov iz zbirke, kar je prikazano v tabeli 3.1.

3.1.5 Semantična podobnost

Omenili smo, da je ciljna skupina naše aplikacije znotraj širokega starostnega intervala. Starejše generacije uporabnikov bi mogoče za dosego enakega učinka zapisale poizvedbo rahlo drugače, kot pa mlajše generacije. Tukaj imamo v mislih predvsem tvorjenje stavkov in uporabo specifičnih izrazov, ki so mogoče tehnično starinski, pomensko preširoki ali preozki ipd. V večini jezikov obstaja skupina besed, ki jim pravimo sopomenke ali soznačnice. Torej besede, ki imajo (skoraj) enak pomen kot kaka druga beseda. Slovar ali zbirko, ki vsebuje to skupino besed imenujemo *tezaver*. Slovarji vsebujejo tudi definicije in podatke o izgovorjavi, zato ne moremo govoriti o slovarju. Za slovenski jezik obstaja odprtokodni tezaver poimenovan OdprtiTezaver[4], ki omogoča iskanje sinonimov besed. Beseda delež ima recimo dva sinonima, in sicer besedi del in odstotek. Za izboljšanje rezultatov iskanja bi torej lahko

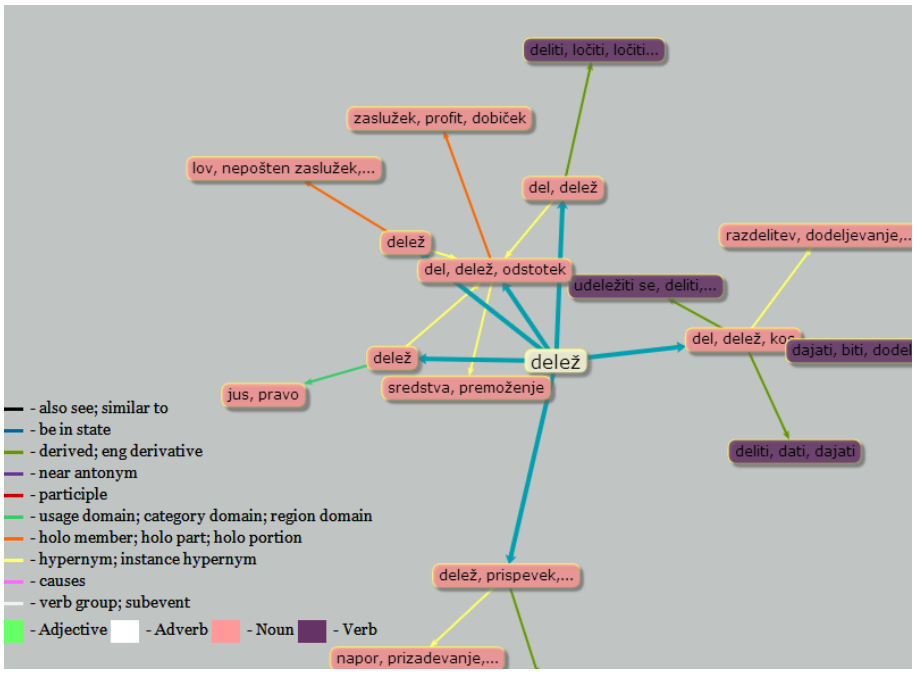
[3]http://lemmatise.ijs.si
[4]http://www.tezaver.si

pojavnicam poiskali sopomenke ali tudi sorodne besede ter na tak način zabrisali starostni faktor in po drugi strani razširili oziroma obogatili samo poizvedbo. Poprej dve omenjeni sopomenki sta z besedo delež v ekvivalenčni relaciji. Obstajajo pa tudi druge vrste relacij. Beseda premoženje je recimo nadpomenka ali hipernim besedi delež, odmerek pa je podpomenka ali hiponim. Tej vrsti relacije, torej hipernim–hiponim ali tudi "Is-a" relacija, pravimo hierarhična relacija [14], katere drevesna struktura je pogosto predstavljena v obliki taksonomije. Projekt leksikalne podatkovne baze imenovan WordNet, katerega začetki segajo v sredino osemdesetih let prejšnjega stoletja, deli angleške besede na sklope sinonimov ali sopomenske nize (*angl. synset*), ki predstavljajo semantično mrežo. Slovenska inačica tega projekta se imenuje SloWNet[5] in orodje, ki ga nadgrajuje, SloWTool[6]. Slednje omogoča prebiranje, urejanje in vizualizacijo semantične mreže s hiperboličnimi grafi in slikami. Primer vizualizacije je prikazan na sliki 3.3. Hierarhične relacije pa seveda vpeljejo hierarhijo, kar pomeni, da povezave med besedami niso več enakovredne, ampak imajo različno vrednost. Obstaja več načinov ugotavljanja pojmovne podobnosti dveh besed v hierarhični semantični mreži. Veliko teh načinov je grajenih na osnovi distanc znotraj taksonomije. Vrednosti semantične podobnosti lahko recimo tudi omejimo na realna števila med 0 in 1, pri čemer imajo sinonimi vrednost 1, semantično bolj oddaljene besede pa manjše vrednosti.

3.1.6 Oblikoslovno označevanje

Te vrste označevanje, ki je najbolj razširjena vrsta korpusnega označevanja, je po [12] postopek, ki pripiše oblikoslovne oznake vsem besednim oblikam v besedilni zbirki. Kot primer podaja besedno zvezo "seminar slovenskega jezika", kjer bi besedni obliki "slovenskega" lahko pripisali oznako P za pridevnik in besedni obliki "jezika" oznako S za samostalnik. Orodje oblikoslovnega označevanja lahko gradimo podobno kot orodje za lematizacijo. Lem-

[5]http://lojze.lugos.si/ darja/slownet.html
[6]http://nl.ijs.si/slowtool



Slika 3.3: Vizualizacija SloWNet vsebine z orodjem SloWTool.

maGen ponuja lematizacijske modele, ki so bili zgrajeni z učenjem na morfosintaktičnem oziroma oblikoskladenjskem leksikonu evropskega projekta MULTEXT-East. Leksikon ponuja obširno zbirko besed s pripadajočimi lemami in oblikoslovnimi oznakami. Oblikoslovna oznaka besede delež je "Ncmsan" po angleški formulaciji oziroma "Sometn" po slovenski. Iz te oznake lahko s pomočjo tabele 3.2 na kratko razberemo, da gre za samostalnik moškega spola. Podobno kot poteka učenje pravil lematizacije, bi lahko potekalo tudi učenje oblikoslovnega označevanja. Besedna vrsta je ena od ključnih oblikoslovnih razdelitev. V našem primeru bi lahko besednim vrstam pripisali različno pomembnost. Eden od rezultatov projekta "Sporazumevanje v slovenskem jeziku"[7] je tudi Oblikoslovni označevalnik za slovenski jezik, ki vsebuje program za označevanje besedil poimenovan PosTaggerTag. Program je izdelan v razvojnem okolju Microsoft Visual Studio

[7]http://www.slovenscina.eu

P	atribut	vrednost	koda	atribut	vrednost	koda
0	Besedna vrsta	samostalnik	S	Category	noun	N
1	Vrsta	občno ime	o	Type	common	c
		lastno ime	l		proper	p
2	Spol	moški	m	Gender	masculine	m
		ženski	z		feminine	f
		srednji	s		neuter	n

Tabela 3.2: Tabela atributov in vrednosti za samostalnik [16].

2008 in za svoje delovanje potrebujejo zagonsko okolje .NET Framework 2.0.

3.1.7 Razdvoumljanje večpomenskih besed

Beseda klop ima več pomenov glede na to, v katerem kontekstu jo najdemo. Pri razdvoumljanju večpomenskih besed (*angl. word-sense disambiguation, WSD*) gre torej za ugotavljanje konteksta besede in pripisovanje pravilnega pomena. Poznamo dva glavna sklopa metod za reševanje nalog WSD [18]. Prvi sklop vsebuje na znanju temelječe metode (*angl. knowledge-based methods*), ki ugotavljajo kontekst na osnovi vnaprej definiranih leksikalnih virov kot sta WordNet ali SloWNet. Drugi sklop so na korpusih temelječe metode (*angl. corpus-based methods*), ki razdvoumljajo večpomenske besede s pomočjo samodejno naučenih pravil na pomensko že označenih korpusih. Prvi sklop metod spada pod tako imenovane površne pristope (*angl. shallow approaches*). Ti se ne ubadajo z razumevanjem besedila, ampak upoštevajo le besede, ki obravnavano besedo obkrožajo. Torej lahko glede na okoliške besede določimo kontekst. Drugi sklop metod uvrščamo med temeljite pristope (*angl. deep approaches*). Kot omenjeno moramo pri teh najprej izdelati korpus z že opredeljenimi oziroma označenimi večpomenskimi besedami. Iz kontekstov teh označenih besed izločimo tako imenovane značilke (*angl. features*) za oblikovanje vadbenih podatkov. S pomočjo učnih algoritmov nato iz teh podatkov ustvarimo model, ki ga v končni fazi preizkusimo na testnih podatkih. Naučeni model poskuša večpomenskim besedam pripisati pravi

pomen, bodisi najbolj verjetnega ali pa množico bolj verjetnih. Dva priljubljena algoritma nadzorovanega strojnega učenja sta recimo naivni Bayes (*angl. Naïve Bayes*) ali pa Metoda podpornih vektorjev (*angl. Support vector machine, SVM*). Oba lahko uporabimo tudi pri klasifikaciji besedilnih dokumentov.

3.1.8 Klasifikacija

Naloga klasifikacije ali kategorizacije dokumentov je pripisati dokument enemu ali več razredom oziroma kategorijam [19], pri čemer so dokumenti lahko besedila, slike, skladbe, itd. Privzemimo, da naš projekt vsebuje že kar nekaj objav. Pri iskanju na primer prakse na področju računalništva, bi se lahko omejili le na objave, ki so v povezavi s področjem računalništva, kar pa brez kategorizacije samih objav težko storimo. Omejitev na podmnožico objav bi tudi pohitrila samo iskanje, kar nam je v dodatno korist. Poleg tega smo poprej že omenili, da lahko s pomočjo kategorizacije pripomoremo k razdvoumljanju večpomenskih besed in omenili dve priljubljeni metodi s področja nadzorovanega strojnega učenja. Obe metodi predstavljata podatke besedilnih dokumentov kot vrečo besed (*angl. bag-of-words, BOW*), ki je neurejena besedna zbirka brez gramatičnih pravil [21].

Naivni Bayesov klasifikator

Naivni Bayesov klasifikator (*angl. Naïve Bayes classifier*) je preprost verjetnostni klasifikator, ki temelji na uporabi Bayesovega teorema ob naivni predpostavki stroge neodvisnosti značilk [18]. Za besedilni dokument d in razred c predstavlja Bayesov teorem

$$P(c|d) = \frac{P(c) \cdot P(d|c)}{P(d)} \tag{3.1}$$

verjetnost, da razred c pripada dokumentu d. V praksi nas zanima samo števec ulomka, sa je *P(d)* enak za vse razrede, torej konstanta, tako da lahko imenovalec izpustimo. Ker je dokument predstavljen kot množica značilk,

lahko enačbo preoblikujemo v

$$P(c|d) = P(c) \cdot P(x_1, x_2, ..., x_n|c) = P(c) \cdot \prod_{i=1}^{n} P(x_i|c), \tag{3.2}$$

kjer so x_i posamezne značilke. Klasifikator išče maksimalne verjetnosti ali maksimalne "a posteriori" verjetnosti (*angl. maximum a posteriori probability, MAP*) $P(c|d)$ po vseh možnih vrednostih c_j iz množice razredov:

$$c_{MAP} = \arg\max_{c_j \in C} P(c_j) \cdot \prod_{i=1}^{n} P(x_i|c_j). \tag{3.3}$$

Verjetnosti P se računajo s pomočjo statistike vreče besed, torej s količinami značilk, kar pomeni da imamo multinomsko porazdelitev. Verjetnost razreda c_j lahko računamo kot $P(c_j) = \frac{N_j}{N}$, kjer je N število vseh dokumentov in N_j število vseh dokumentov, ki pripadajo razredu j. Prvi del produkta pa predstavlja $P(x_i|c_j) = \frac{N_{ij}}{N_j}$, kjer je N_{ij} frekvenca dokumentov razreda j, ki vsebujejo značilko i, N_j pa število dokumentov, ki pripadajo razredu j. Osnovna enačba vsebuje produkt, kar pomeni, da lahko ničelne verjetnosti značilk pri določenem razredu ($P(x_i|c) = 0$) privzamejo ničelno maksimalno verjetnost razreda. Zaradi tega se vrednost $P(x_i|c_j)$ računa skupaj z Laplaceovim glajenjem (*angl. Laplace smoothing*) kot

$$P(x_i|c_j) = \frac{N_{ij} + 1}{M + \sum_{k=1}^{M} N_{kj}}, \tag{3.4}$$

kjer je M velikost celotnega besednjaka in N_{kj} število dokumentov razreda j, ki vsebujejo besedo k iz besednjaka [20]. Namesto Laplaceovega glajenja lahko uporabimo tudi katero drugo vrsto glajenja [22]. Konča enačba multinomskega naivnega Bayesovega klasifikatorja se glasi:

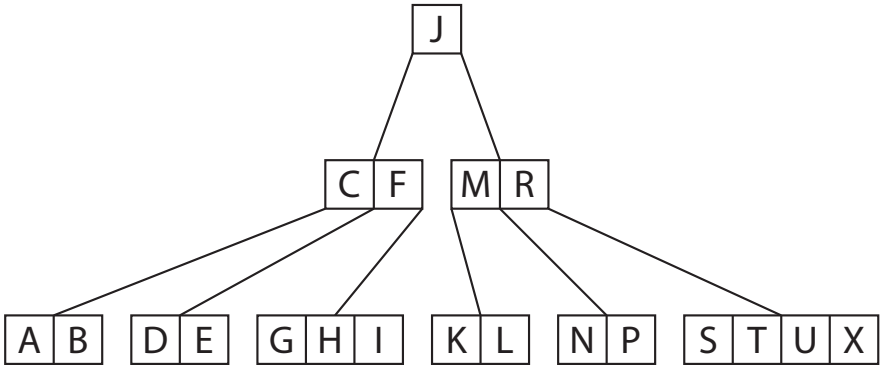
$$c_{NB} = \arg\max_{c_j \in C} \frac{N_j}{N} \cdot \prod_{i=1}^{n} \frac{N_{ij} + 1}{M + \sum_{k=1}^{M} N_{kj}}. \tag{3.5}$$

3.2 Strukturiranje obdelanih pojavnic

V prejšnjem poglavju smo opisovali različne postopke pri obdelavi naravnega jezika, pri čemer se nismo omejevali samo na naslove dokumentov ali zgolj povzetke, ampak smo v sam proces obdelave vključili vse besede oziroma pojavnice dokumentov. Ob poizvedbi se torej v proces iskanja vključi vse dokumente in besede iskalnega niza se primerja z vsemi besedami dokumentov. Tej tehniki pravimo tudi *iskanje po vsem besedilu* (*angl. full-text search*). Po [4] je razlogov za priljubljenost tovrstnega iskanja več:

- podatkovne baze so čedalje bolj v uporabi kot repozitoriji dokumentov,
- mnogo podatkovnih baz je javno dostopnih,
- shranjevanje je poceni,
- pojavljajo se nove vrste dokumentov in tudi težnje ali zahteve, da se te vrste dokumentov shranjujejo v nativni obliki,
- raziskovanje in analiziranje dokumentov in tekstovnih podatkov zahteva, da so podatki shranjeni v podatkovnih bazah zmožnih celostnega iskanja besedila,
- iskanje po vsem besedilu služi kot solidna osnova za bolj napredne tehnike analiziranja kot je na primer razširitev klasičnega podatkovnega rudarjenja (*angl. data mining*) na tekstovno rudarjenje,
- razvijalci si želijo standardiziranega vmesnika za iskanje dokumentov in tekstovnih podatkov v svojih podatkovnih bazah.

Pri velikem številu dokumentov je ta tehnika običajno razdeljena na dva dela, in sicer na *indeksiranje* ter *iskanje*. Dobršen del nalog prvega dela smo že opisali v poglavju 3.1. Končni produkt indeksiranja je indeks obdelanih besed naravnega jezika, na katerega se opira drugi del, torej iskanje. Velikokrat zasledimo tudi izraz obrnjen indeks (*angl. inverted index*), saj nam, glede na vhodno besedo, ponuja seznam dokumentov, ki vsebujejo to besedo [23].



Slika 3.4: Primer B-drevesa reda 5. Elementov je najmanj 2 in največ 4.

Obrnjeni indeksi so za namene iskanja informacij (*angl. information retrieval*) pogosto realizirani v generalizirani strukturi binarnega iskalnega drevesa imenovani B-drevo (*angl. B-tree*). Vsako notranje vozlišče B-drevesa ima število elementov na intervalu $[a, b]$, kjer sta a in b primerni pozitivni števili. Za razliko od binarnih dreves, katerih vozlišča imajo lahko največ dva sinova, imajo vozlišča B-dreves lahko več sinov. Vse operacije te strukture se v najslabšem možnem primeru izvedejo v času reda $\Theta(\log n)$. Slika 3.4 prikazuje primer B-drevesa, kjer je a 2 in b 4, torej imajo vozlišča najmanj 3 sinove oziroma 2 ključa in največ 5 sinov oziroma 4 ključe. B-drevesa so primerna za delo na diskih ali drugih zunanjih pomnilniških napravah, ker dobro minimizirajo diskovne vhodno-izhodne operacije.

3.3 Relevantno iskanje

Ena od današnjih popularnih in množično uporabljanih podatkovnih baz je odprtokodna podatkovna baza *MySQL*, ki je od leta 2010 v lasti podjetja Oracle. Iz [5] razberemo, da ta pri "full-text" poizvedbah izvaja razvrščanje (*angl. ranking*) s pomočjo modela vektorskega prostora (*angl. vector space model, VSM*). Za ugotavljanje uspešnosti ali relevantnosti zadetkov je na voljo ustreznostna mera (*angl. relevance measure*).

3.3.1 Model vektorskega prostora

Model vektorskega prostora je sklop dokumentov, ki so predstavljeni kot vektorji znotraj skupnega vektorskega prostora. Taka predstavitev je bistvenega pomena za številne dejavnosti iskanja informacij, od zadetkov dokumentov ob poizvedbah, klasifikacije dokumentov do gručenja dokumentov. Pojavitve iskanih izrazov znotraj besedil si lahko predstavljamo kot daljice v večdimenzionalnem prostoru, pri čemer ima vsak izraz svojo dimenzijo. Te daljice so lahko različnih razdalj, kjer upoštevamo, da je enota razdalje enaka eni pojavitvi izraza v besedilu. Kaj kmalu ugotovimo, da lahko na tak sistem apliciramo splošno trigonometrijo za izračun razdalj, ki se enačijo z merami podobnosti. Na ta način je osnovano tudi *tf–idf* uteževanje.

3.3.2 Tf–iDf

Te vrste uteževanje v splošni statistični obliki za določen izraz iz nekega dokumenta zapišemo kot

$$tf_{i,d}\text{–}idf_i = w_{i,d} = tf_{i,d} \cdot idf_i, \tag{3.6}$$

pri čemer:

- $w_{i,d}$ predstavlja utež izraza i v dokumentu d,
- $tf_{i,d}$ pomeni frekvenco izraza ali kolikokrat se izraz i pojavi v dokumentu d,
- idf_i pomeni inverz frekvence dokumentov za izraz i.

Zadnji del dejansko predstavlja splošen razširjen izraz:

$$idf_i = \log_{10} \frac{D}{df_i}, \tag{3.7}$$

kjer je D število vseh dokumentov v zbirki ter df frekvenca oziroma število dokumentov, ki vsebujejo izraz i. Za lažjo predstavo si omislimo enostaven primer. Tabela 3.3 predstavlja našo podatkovno bazo, ki vsebuje zbirko petih dokumentov. Za iskanje po celotnem besedilu moramo najprej ustvariti

Št. dokumenta	Vsebina dokumenta
1	Grška usoda, prihodnost vseh nas
2	Več tisoč Portugalcev išče boljšo prihodnost
3	Na kocki je naša prihodnost in prihodnost vseh generacij
4	V Berlinu odprli razširjen judovski muzej
5	Poglejte si novi video skupine Coldplay

Tabela 3.3: Struktura podatkovne baze

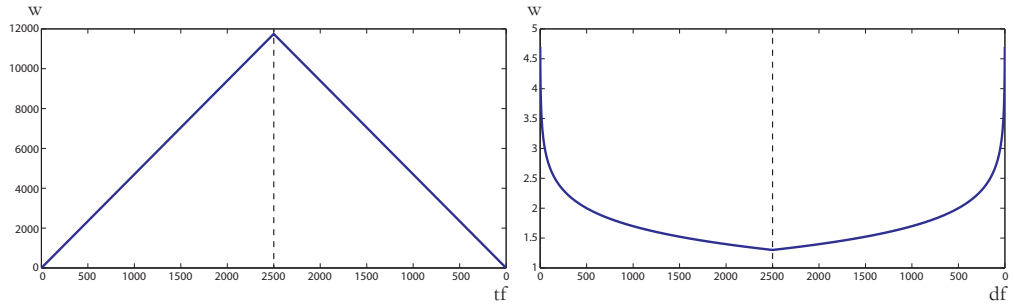
indeks s pripadajočimi ključi in njihovimi utežmi. Izračun ene od uteži izpeljimo korak po koraku. V dokumentu z zaporedno številko 3, ki vsebuje skupno 9 izrazov, se dvakrat pojavi izraz *prihodnost*. Dokument vsebuje tudi blokirane besede kot so *na*, *je*, *in*, skratka vezniki, predlogi, lahko tudi izrazi krajši od določenega števila znakov ipd. Skupno imamo torej 6 izrazov v dokumentu. Vrednost prvega dela enačbe uteži za izraz prihodnost je torej 2. Drugi del enačbe poračunamo po:

$$idf_i = \log_{10} \frac{5}{3} = 0.222. \tag{3.8}$$

Vrednost uteži w za izraz prihodnost v dokumentu z zaporedno številko 3 dobimo z zmnožkom obeh delov enačbe $w = 2 \cdot 0.222 = 0.444$. Preostale dobljene vrednosti so zapisane v tabeli 3.4. Te vrednosti smo dobili s pomočjo splošne enačbe. Obstaja pa tudi kopica variacij te enačbe, pri čemer imamo bodisi različne načine računanja frekvenc izrazov bodisi različne načine računanja frekvenc dokumentov ali pa različne tipe normalizacije. V splošnem lahko zapišemo model vektorskega prostora z uteževanjem besed na sledeč način:

$$w_{i,d} = L_{i,d} \cdot G_i \cdot N_d, \tag{3.9}$$

kjer je $L_{i,d}$ lokalna frekvenca besede i znotraj dokumenta d. G_i je globalna vrednost ali inverz frekvence dokumentov za izraz i ter N_d normalizacijski del. Iz [5] razberemo, da MySQL k splošni enačbi doda tudi normalizacijski



Slika 3.5: Spreminjanje vrednosti uteži glede na *tf* in *df* ($D = 50000$).

faktor, tako da je enačba uteži v razširjeni varianti:

$$w_{i,d} = \frac{\ln(tf_{i,d}) + 1}{sumtf} \cdot \ln\left(\frac{D}{df_i} - 1\right) \cdot \frac{U}{1 + 0.0115 \cdot U}, \tag{3.10}$$

kjer je:

- $tf_{i,d}$ število pojavitev izraza i v dokumentu d,
- $sumtf$ vsota $\log(tf_{i,d}) + 1$ za vse unikatne izraze znotraj istega dokumenta,
- U število unikatnih ali različnih izrazov v dokumentu,
- D število vseh dokumentov,
- df_i število dokumentov, ki vsebujejo izraz i.

Izhajali smo iz splošne tf–idf enačbe, ki vsebuje dva dela. Večji kot je po vrednosti prvi del ali večkrat kot se izraz ponovi znotraj dokumenta, večjo težo bo izraz imel v tem dokumentu. Drugi del pa ima obraten vpliv na utež, in sicer zmanjšuje vrednost uteži gleda na to v koliko drugih dokumentih se izraz pojavi. Spreminjanje vrednosti uteži splošne enačbe glede na *tf*, ko je *df* enako 1, in *df*, ko je *tf* enako 1, prikazuje slika 3.5.

Če bi torej v naši zbirki petih dokumentov iskali tiste, ki vsebujejo izraz prihodnost, bi s pomočjo "full-text" indeksa dobili 3 zadetke in sicer dokumente z zaporedno številko 1, 2 in 3. Dokument št. 3 bi bil prvi na lestvici, saj ima pri njem iskani izraz največjo težo in je tako najbolj relevanten.

Dok. št.	Izraz	L_o	G_o	N_o	w_o	L_r	G_r	N_r	w_r
4	berlinu	1	0.699	1	0.699	0.2	1.386	4.728	1.311
2	boljšo	1	0.699	1	0.699	0.167	1.386	5.613	1.297
5	coldplay	1	0.699	1	0.699	0.2	1.386	4.728	1.311
3	generacij	1	0.699	1	0.699	0.176	1.386	4.728	1.151
1	grška	1	0.699	1	0.699	0.2	1.386	4.728	1.311
2	išče	1	0.699	1	0.699	0.167	1.386	5.613	1.297
4	judovski	1	0.699	1	0.699	0.2	1.386	4.728	1.311
3	kocki	1	0.699	1	0.699	0.176	1.386	4.728	1.151
4	muzej	1	0.699	1	0.699	0.2	1.386	4.728	1.311
1	nas	1	0.699	1	0.699	0.2	1.386	4.728	1.311
3	naša	1	0.699	1	0.699	0.176	1.386	4.728	1.151
5	novi	1	0.699	1	0.699	0.2	1.386	4.728	1.311
4	odprli	1	0.699	1	0.699	0.2	1.386	4.728	1.311
5	poglejte	1	0.699	1	0.699	0.2	1.386	4.728	1.311
2	portugalcev	1	0.699	1	0.699	0.167	1.386	5.613	1.297
1	prihodnost	1	0.222	1	0.222	0.2	-0.405	4.728	-0.383
2	prihodnost	1	0.222	1	0.222	0.167	-0.405	5.613	-0.379
3	prihodnost	2	0.222	1	0.444	0.297	-0.405	4.728	-0.570
3	prihodnost	2	0.222	1	0.444	0.297	-0.405	4.728	-0.570
4	razširjen	1	0.699	1	0.699	0.2	1.386	4.728	1.311
5	skupine	1	0.699	1	0.699	0.2	1.386	4.728	1.311
2	tisoč	1	0.699	1	0.699	0.167	1.386	5.613	1.297
1	usoda	1	0.699	1	0.699	0.2	1.386	4.728	1.311
2	več	1	0.699	1	0.699	0.167	1.386	5.613	1.297
5	video	1	0.699	1	0.699	0.2	1.386	4.728	1.311
1	vseh	1	0.398	1	0.398	0.2	0.405	4.728	0.383
3	vseh	1	0.398	1	0.398	0.176	0.405	4.728	0.337

Tabela 3.4: Izračunane postavke in uteži izrazov zbirke dokumentov po osnovni in razširjeni enačbi. Negativne vrednosti razširjene variante pripadajo besedam, ki se pojavijo v več kot polovici dokumentov celotne zbirke in se jih pri iskanju ignorira.

3.4 Ujemanje dokumentov

Ujemanje oziroma podobnost dveh ali več dokumentov merimo na osnovi prekrivanja njihovih vsebin. Omenili smo, da lahko vsebine dokumentov predstavimo kot vektorje znotraj skupnega vektorskega prostora. Ena od možnosti za merjenje podobnosti ali ujemanja bi bil tako lahko kosinus kota dveh vektorjev. Prav to izkorišča kosinusna podobnost (*angl. cosine similarity*), ki je definirana s kotom vektorjev značilk dveh dokumentov, pri čemer ti vektorji temeljijo na frekvencah besed. V osnovi se kosinusna podobnost računa kot:

$$\cos(\theta) = \frac{a \cdot b}{\|a\|\|b\|}, \tag{3.11}$$

kjer je v števcu ulomka skalarni produkt dveh vektorjev in v imenovalcu zmnožek evklidskih norm vektorjev. S pomočjo mere tf–idf lahko besede, ki predstavljajo dimenzije vektorjev, utežimo. Kosinusno podobnost lahko tako preoblikujemo v

$$ujemanje(d_1, d_2) = \frac{\sum_{i=1}^{n} w_{i,d_1} \cdot w_{i,d_2}}{\sqrt{\sum_{i=1}^{n} (w_{i,d_1})^2} \cdot \sqrt{\sum_{i=1}^{n} (w_{i,d_2})^2}}, \tag{3.12}$$

kjer je w_{i,d_1} utežena i-ta značilka prvega dokumenta ter w_{i,d_2} utežena i-ta značilka drugega dokumenta. Če značilke vektorjev niso negativne, potem je rezultat kosinusne podobnosti na intervalu od 0 do 1, pri čemer vrednost 1 pomeni največje ujemanje.

V poglavju 3.1.5 smo govorili o semantični podobnosti besed, kjer smo omenili predvsem ekvivalenčne sopomenke ter hierarhično povezane nadpomenke in podpomenke. Te sorodne besede oziroma leme sorodnih besed lahko dodamo lemam pojavnic. V enačbi 3.12 bi to pomenilo, da dokumentu d_1 dodamo ovrednotene sorodne besede, ki jih iščemo v dokumentu d_2. Pri velikem številu sorodnih besed so lahko rezultati ujemanja zelo nizki. Lahko pa uporabimo tudi drugačen način računanja podobnosti, ki ne temelji na

vektorskem modelu [28].

$$ujemanje(d_1, d_2) = \frac{1}{2} \cdot (\frac{\sum\limits_{i \in d_1} maxSim(i, d_2) \cdot idf_i}{\sum\limits_{i \in d_1} idf_i} + \frac{\sum\limits_{i \in d_2} maxSim(i, d_1) \cdot idf_i}{\sum\limits_{i \in d_2} idf_i}) \tag{3.13}$$

Za vsako besedo i dokumenta d_1 poskušamo identificirati besedo dokumenta d_2 z največjo semantično podobnostjo ($maxSim$) in obratno. Funkcija semantične podobnosti prav tako lahko bazira na znanju temelječih metodah ali pa na korpusih temlječih metodah. Ena bolj znanih korpusnih metod je funkcija avtorjev Leacock in Chodorow

$$maxSim = -\log \frac{L}{2 \cdot D}, \tag{3.14}$$

kjer je L dolžina najkrajše poti med dvema sopomenskima nizoma (*angl. synset*) sliki 3.3 podobne semantične mreže, ki jo dobimo s štetjem vozlišč, in D največja globina taksonomije. Velika večina na znanju temlječih metod se opira na semantično mrežo WordNet in računa semantično podobnost s pomočjo distanc znotraj taksonomij.

Poglavje 4

Zasnova in izdelava rešitve

4.1 Tehnologije in orodja

4.1.1 Odprtokodni programski sestav Winginx

V spletnem kontekstu kratica WAMP (iz Windows–Apache–MySQL–PHP) navadno označuje skupek odprtokodne programske opreme, ki se poganja na precej razširjenem in popularnem operacijskem sistemu Windows in tvori delujoč spletni strežnik, namenjen večinoma razvijanju in testiranju aplikacij, ki je sposoben gostiti dinamične spletne strani. Inačica prirejena za operacijski sistem Linux nosi kratico LAMP. Winginx je primer WAMP sestava, le da je namesto strežnika Apache uporabljen strežnik Nginx, tako da bi lahko sestav označili s kratico WNMP. Za strukturiranje in shranjevanje podatkov smo uporabili poleg relacijske podatkovne baze MySQL tudi dokumentno usmerjeno (*angl. document-oriented*) nerelacijsko (*angl. non-relational, NoSQL*) podatkovno bazo MongoDB.

Nginx

Razvoj odprtokodnega HTTP strežnika Nginx se je začel leta 2002 pod vodstvom Igorja Sisojeva. Prvo javno objavo je doživel leta 2004. Obnaša se lahko tudi kot obratni posrednik (*angl. reverse proxy*) ali kot IMAP/POP3

posredniški strežnik. Njegove glavne značilnosti so majhna poraba resursov, širok nabor funkcij, enostavna konfiguracija ter stabilna in visoka zmogljivost. Pri reševanju našega problema nam bo služil kot spletni strežnik zmožen generiranja dinamičnih spletnih strani.

MySQL

Mnogo popularnih svetovnih spletnih strani z visokim podatkovnim prometom bazira na sistemu za upravljanje z relacijskimi podatkovnimi bazami (RDBMS) MySQL. Ta odprtokodna implementacija relacijske podatkovne baze, ki s podatki manipulira s pomočjo jezika SQL, slovi po svoji razširjenosti na različnih računalniških okoljih, zanesljivosti in enostavnosti pri delu. Relacijsko podatkovno bazo smo uporabili za hranjenje ključnih podatkov.

MongoDB

MongoDB je v jeziku C++ napisana odprtokodna, skalabilna in visoko zmogljiva nerelacijska podatkovna baza. Je dokumentno usmerjena, kar pomeni, da shranjuje strukturirane podatke v dokumente oblike JSON, katerih shema je lahko dinamična. Nerelacijska podatkovna baza služi za hranjenje indeksnih lem sporočil, *df* frekvenc pri računanju *idf* in drugih podatkov nad katerimi lahko izvajamo agregacijo.

PHP

Skriptni jezik na strežniški strani (*angl. server-side scripting language*) je v našem programskem sestavu PHP, ki je okrajšava za "PHP: Hypertext Preprocessor". Večinoma se ga uporablja za razvoj dinamičnih spletnih vsebin. Spletni strežnik, v našem primeru Nginx, interpretira skriptno kodo, ki je lahko prepletena z izvorno kodo HTML, in ustvari generirano spletno stran. Skripte lahko poganjamo tudi v ukaznem načinu. PHP nam bo služil za komunikacijo s podatkovnimi bazami in implementacijo osnovnih funkcionalnosti projekta.

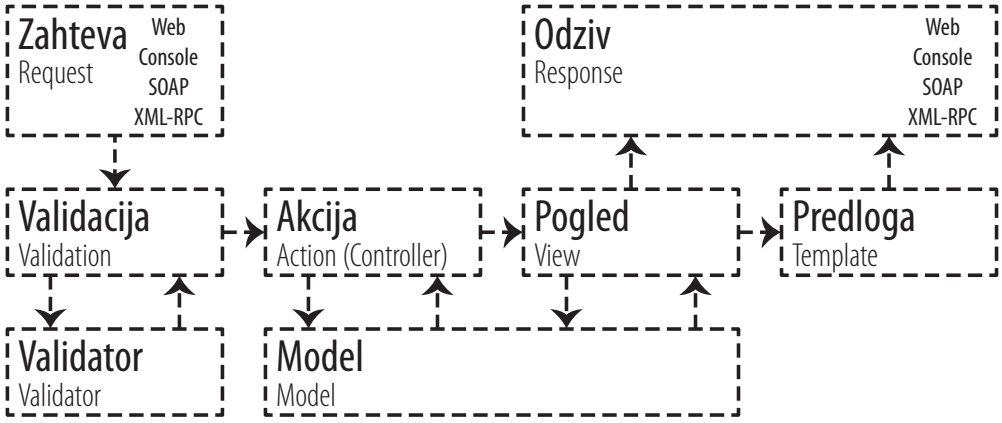
4.1.2 Agavi Framework

Obstaja cela kopica spletnih aplikacijskih ogrodij (*angl. web application framework, WAF*), katerih namen je olajševanje tipičnih ali pogostih nalog in opravil tekom razvijanja dinamičnih spletnih strani, aplikacij ali storitev. Agavi[1] je iz Mojavi 3 izhajajoče zmogljivo in razširljivo (*angl. scalable*) aplikacijsko ogrodje, ki sledi arhitekturnemu vzorcu MVC (Model—View—Controller). Spletnim razvijalcem omogoča pisanje čiste, enostavno vzdrževane in razširljive kode. Iz Mojavi 3 izhaja tudi mogoče malce bolj znano ogrodje Symfony[2]. Agavi dopušča razvijalcem veliko svobode pri implementacijskih odločitvah in njegove komponente so enostavno razširljive. Ogrodje je osnovano na fleksibilnem konfiguracijskem sistemu, ki temelji na strukturi XML. Slika 4.1 prikazuje potek izvršitve zahteve, pri čemer je ta lahko spletna, SOAP ali pa gre za klic XML-RPC ali za klic iz konzole. Vse zahteve morajo obvezno skozi proces validacije. Če kateri izmed parametrov zahteve ne prestane testa validacije, se zahteva zavrne. O tem odloča "validator". Uspešno validiran zahtevek prevzame krmilnik oziroma akcija, ki interpretira parametre zahteve in ukazuje modelom. Krmilnik se glede na ugotovljeno odloči za pravilen pogled (*angl. view*), ki pripravi, lahko skupaj z modeli, vse potrebno za oblikovanje odziva. Odziv je lahko izvršen neposredno ali pa posredno preko ustrezne predloge. Ogrodje Agavi nam bo služilo za hitro in enostavno vzpostavitev glavnih temeljev spletne storitve ter za boljše organiziran in strukturiran pristop k implementaciji osnovnih metod.

4.2 Podatkovna struktura

V poglavju 2 smo že omenili, da ima naš projekt dva glavna akterja, katerih primeri uporabe povečini vključujejo entiteto objave. Akterjema smo pripisali med drugim spol, morebiten akademski naziv ter nenazadnje tudi organizacijo, kateri pripadata. Glavna akterja si lahko izmenjujeta tudi sporočila

[1] http://www.agavi.org
[2] http://www.symfony.com



Slika 4.1: Potek izvršitve zahteve aplikacijskega ogrodja Agavi.

ali pa dodajata eden drugega na svoj seznam kontaktov.

4.2.1 Relacijski model

Slika 4.2 prikazuje konceptualni model relacijske podatkovne strukture, ki se preslika v fizični podatkovni model. Tabele znotraj tega modela lahko tudi bolj podrobno predstavimo.

Tabela "academictitle"

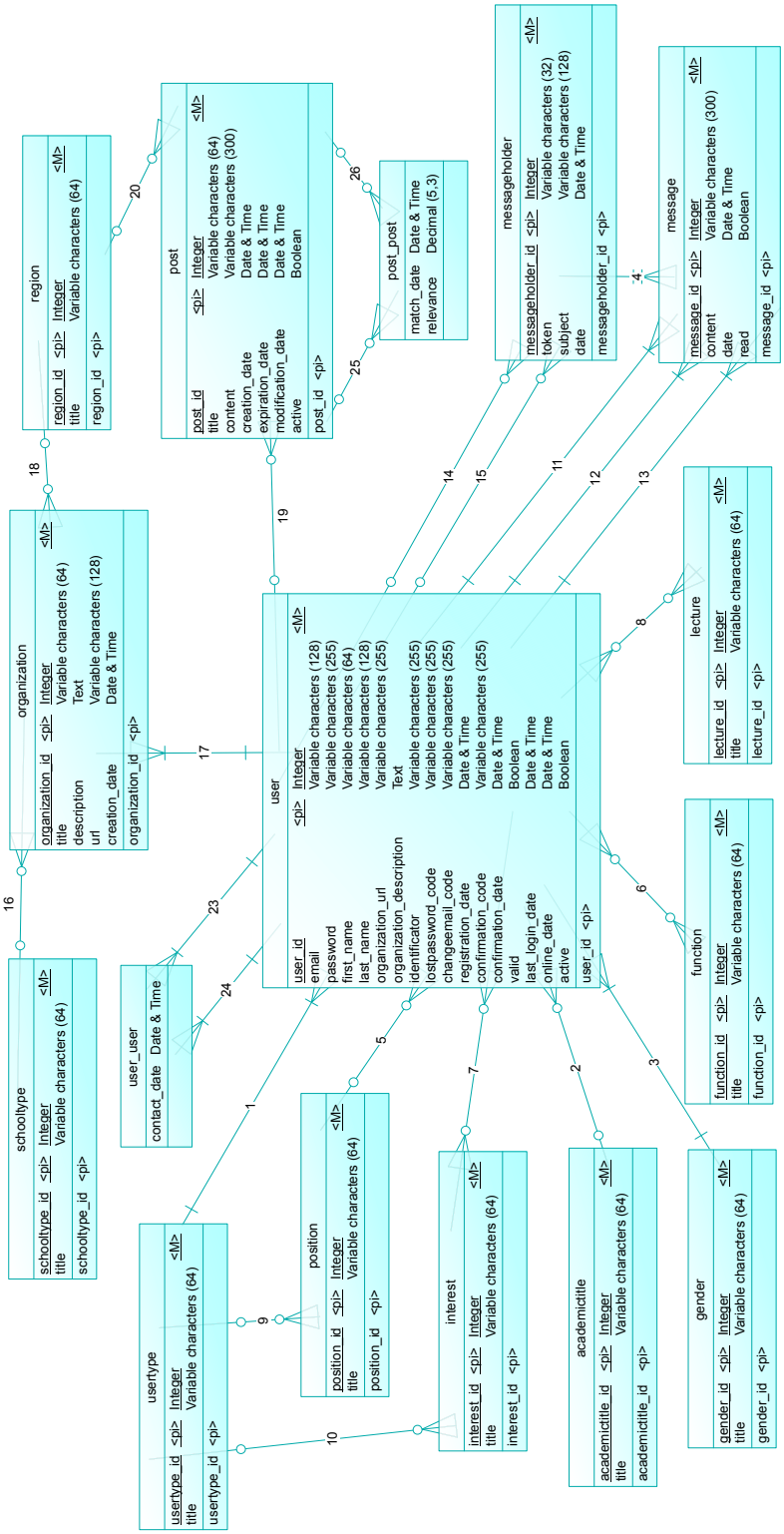
id	Unikatni identifikator akademskega naziva
title	Akademski naziv

Tabela akademskih nazivov, kot so npr. docent, redni profesor, asistent ipd. Uporabnik je lahko tudi brez akademskega naziva.

Tabela "function"

id	Unikatni identifikator funkcije
title	Naziv funkcije

Tabela funkcij oziroma področij dela uporabnikov znotraj organizacije. Primer področja je na primer nabava, oprema, kadrovanje, načrtovanje itd.



Slika 4.2: Konceptualni model relacijskega podatkovnega modela aplikacije.

Tabela "gender"

id	Unikatni identifikator spola
title	Naziv spola

Tabela spola uporabnikov, kjer sta zapisana ženski in moški spol.

Tabela "interest"

id	Unikatni identifikator interesa
usertype_id	Identifikator vrste uporabnika
title	Naziv interesa

Tabela interesov uporabnikov, ki jih je lahko tudi več. Interesi so na primer sodelovanje, usposabljanje, praksa itd.

Tabela "lecture"

id	Unikatni identifikator učnega predmeta
title	Naziv učnega predmeta

Tabela učnih predmetov oziroma predavanj. Vsebinsko sicer enaki učni predmeti imajo lahko različne nazive, zato so ti nazivi širše opredeljeni. Na seznamu srečamo recimo predmete kot so matematika, geografija, kemija, statistika, statika ipd.

Tabela "message"

id	Unikatni identifikator sporočila
messageholder_id	Identifikator nosilca sporočil
user_id	Identifikator uporabnika oz. lastnika sporočila
sender_id	Identifikator pošiljatelja
recipient_id	Identifikator prejemnika
content	Vsebina sporočila
date	Datum in čas sporočila
message_read	Zastavica ali je bilo sporočilo prebrano ali ne

Tabela sporočil vsebuje podatke o sporočilih, torej o njihovih naslovnikih, pošiljateljih, vsebini, času pošiljanja ipd. Na vsako sporočilo lahko naslovnik tudi odgovori, celoten dialog pa je združen v celoto s pomočjo unikatnega

nosilca. Ob vsakem pošiljanju se shranita dva izvoda sporočila. En izvod pripada pošiljatelju in drugi naslovniku.

Tabela "messageholder"

id	Unikatni identifikator nosilca sporočil
token	Unikatna koda nosilca
subject	Zadeva skupka sporočil
sender_id	Identifikator začetnega pošiljatelja
recipient_id	Identifikator začetnega prejemnika
date	Datum in čas pričetka dialoga

Tabela nosilcev sporočil vsebuje osnovne informacije o dialogih, torej o zadevi dialoga, pričetku, začetnemu pošiljatelju in prejemniku.

Tabela "organization"

id	Unikatni identifikator organizacije
usertype_id	Identifikator vrste uporabnika
schooltype_id	Identifikator vrste izobraževalne ustanove
region_id	Identifikator regije
title	Naziv organizacije
description	Opis organizacije
url	Spletni naslov organizacije
creation_date	Datum in čas vnosa

Tabela vsebuje podatke o organizacijah obeh vrst uporabnikov. Vrsta izobraževalne ustanove je lahko tudi nedoločena.

Tabela "position"

id	Unikatni identifikator pozicije
usertype_id	Identifikator vrste uporabnika
title_id	Naziv pozicije

Tabela vsebuje nazive pozicij obeh vrst uporabnikov znotraj organizacij. Primeri nazivov so socialni delavec ali delavka, kuhar, koordinator ipd.

Tabela "post"

id	Unikatni identifikator objave
user_id	Identifikator uporabnika
region_id	Identifiktor regije
title	Naslov objave
content	Vsebina objave
creation_date	Datum in čas vnosa objave
expiration_date	Datum in čas izteka objave
modification_date	Datum in čas spremembe objave
active	Zastavica ali je objava aktivna ali ne

Tabela vsebuje podatke o objavah, torej o lastniku objave, naslovu, vsebini itd. Regija je neobvezen podatek, pravtako tudi rok trajanja objave.

Tabela "post_post"

post_id	Unikatni identifikator objave
match_id	Unikatni identifikator ujemajoče objave
match_date	Datum in čas ujetja
relevance	Delež ujemanja

Predstavljena je tabela ujemajočih se objav objav, torej predlogov objav. Zabeležen je tudi datum in čas ugotovitve ujemanja ter relevanca ujemanja.

Tabela "region"

id	Unikatni identifikator regije
title	Naziv regije

Tabela regij vsebuje nazive vseh 12 slovenskih regij, kot so Pomurska regija, Goriška regija itd.

Tabela "schooltype"

id	Unikatni identifikator vrste izobraževalne ustanove
title	Naziv vrste izobraževalne ustanove

Tabela nazivov vrst izobraževalnih ustanov, kot so osnovna šola, srednja šola,

višja šola itd.

Tabela "user"

id	Unikatni identifikator uporabnika
usertype_id	Identifikator vrste uporabnika
email	Elektronski naslov uporabnika
password	Kriptirano geslo uporabnika
gender_id	Identifikator spola uporabnika
academictitle_id	Identifikator akademskega naziva uporabnika
first_name	Ime uporabnika
last_name	Priimek uporabnika
position_id	Identifikator pozicije uporabnika
organization_id	Identifikator organizacije
organization_url	Spletni naslov organizacije
organization_description	Opis organizacije
identificator	Razširjeni unikatni identifikator uporabnika
lostpassword_code	Koda pri resetiranju gesla
changeemail_code	Koda pri spreminjanju elektronskega naslova
registration_date	Datum in čas registracije
confirmation_code	Koda potrditve registracije
confirmation_date	Datum in čas potrditve registracije
valid	Zastavica ali je uporabnik potrjen ali ne
last_login	Datum in čas zadnje prijave
online_date	Datum in čas zadnje "online" pojavitve
active	Zastavica ali je uporabnik aktiven ali ne

Tabela vsebuje ključne podatke obeh vrst uporabnikov. Geslo uporabnika je zakodirano z močnim simetričnim enkripcijskim algoritmom imenovanim *Blowfish* z visokim stroškovnim parametrom (*angl. cost parameter*). Tabela vsebuje tudi naslov URL ter opis organizacije, čeprav je ta shranjen že pri sami organizaciji. To daje možnost uporabniku, da naslov URL spremeni ter sam opis organizacije razširi oziroma popravi. Različne kode v tabeli so uporabljene pri funkcijah spreminjanja elektronskega naslova, gesla ipd.

Tabela "usertype"

id	Unikatni identifikator vrste uporabnika
title	Naziv vrste uporabnika

Tabela z nazivoma dveh osnovnih vrst uporabnikov.

Tabela "user_function"

user_id	Identifikator uporabnika
function_id	Identifikator funkcije

Tabela vsebuje seznam funkcij za vsakega uporabnika.

Tabela "user_interest"

user_id	Identifikator uporabnika
interest_id	Identifikator interesa

Tabela vsebuje seznam interesov za vsakega uporabnika.

Tabela "user_lecture"

user_id	Identifikator uporabnika
lecture_id	Identifikator učnega predmeta

Tabela vsebuje seznam učnih predmetov, ki jih uporabniki iz vrst izobraževalnih ustanov predavajo.

Tabela "user_user"

user_id	Identifikator uporabnika
contact_id	Identifikator kontakta
contact_date	Datum in čas stvaritve kontakta

Tabela uporabnikov uporabnikov, torej kontaktov uporabnikov. Shranjen je tudi datum in čas vnosa kontakta.

Slika 4.3: Strukture glavnih dokumentov v obliki JSON.

4.2.2 Nerelacijski model

Primarno so podatki v nerelacijski podatkovni bazi MongoDB predstavljeni v obliki BSON, kar je krajše za binarni JSON. Gre za binarno zakodirano serializacijo dokumentov v obliki JSON. Glavna lastnost tega zapisa je ta, da ga je enostavno in hitro preleteti (*angl. traverse*). Tudi pretvarjanje podatkov v obliko BSON, in iz nje, je hiter proces. MongoDB uporabljamo za shranjevanje modela klasifikacije, obdelanih pojavnic oziroma lem dokumentov in uporabnikov, uteženih sorodnih izrazov ter frekvenc besed za izračun ustreznostne mere. Slika 4.3 prikazuje omenjene strukture dokumentov.

4.3 Obdelava naravnega jezika

4.3.1 Tokenizacija

Že v poglavju 3.1.1 smo govorili o regularnih izrazih. Regularni izraz je poseben znakovni niz, ki predstavlja iskalni vzorec. S pomočjo teh vzorcev lahko enostavno poiščemo posamezne črke znotraj besedil, samo številke ali

pa mogoče kakšno bolj zapleteno zaporedje znakov, kot so na primer telefonske številke. V procesu tokenizacije razgrajujemo besedila na osnovne enote, med katerimi so lahko tudi ločila. V našem primeru je bila osnovna enota beseda, tako da smo želeli iz objave izluščiti vse besede. Tokenizacijo smo implementirali s pomočjo regularnih izrazov.

Zaporedje 4.1: Tokenizacija s pomočjo regularnih izrazov

```
1 // tokenize
2 preg_match_all('/\w+/ui', $input, $tokens);
```

Nad vhodnim besedilom smo izvedli regularni izraz "\w+" in rezultat shranili v spremenljivko `tokens`. Metaznak (*angl. metacharacter*) "\w" predstavlja vse alfanumerične znake skupaj s podčrtajem, torej poišče vse besede, ki lahko vsebujejo tudi podčrtaj. Zastavica "u" na koncu izraza pomeni unikod (*angl. unicode*) način, torej zajamemo tudi besede slovenskega jezika, zastavica "i" pa neobčutljivost na velike in male črke. Glede na to, da je raba apostrofa v slovenskem jeziku le neskladenjska in da opuščaja v slovenskih imenih in priimkih navadno ne srečamo, s problemom na sliki 3.1 nismo imeli težav oziroma izraz predpostavlja zadnjo rešitev.

4.3.2 Blokiranje

S pomočjo spletnega arhiva smo se dokopali do seznama blokiranih besed omenjenega v poglavju 3.1.2. Ker so števila lahko tudi pomemben podatek, smo jih odstranili s seznama. Ta sedaj vsebuje 4135 besed. Med prvimi petimi na seznamu so recimo besede a, ali, ampak, bodisi ter in. Seznam lahko seveda tudi dopolnjujemo.

Zaporedje 4.2: Odstranjevanje blokiranih besed iz množice pojavnic

```
1 // remove stop words (case insensitive)
2 $tokens = array_udiff($tokens, $this->stopwords, 'strcasecmp');
```

Besede seznama so vsaka v svoji vrstici shranjene v datoteki. Ta se tekom obdelave naravnega jezika prebere v urejeno množico in s pomočjo ukaza

prikazanega na izvlečku 4.2 se iz množice pojavnic odstrani tiste, ki so tudi v množici blokiranih besed.

4.3.3 Lematizacija

Pojavnice smo lematizirali s pomočjo orodja LemmaGen omenjenega v poglavju 3.1.4. Uporabili smo verzijo 2.2, ki je spisana v jeziku C++, tako da lahko kodo prevedemo za različna okolja. LemmaGen ponuja tudi že izdelane binarne lematizacijske modele. Lahko pa seveda zgradimo lasten model, pri čemer rabimo primerno označen korpus. Takšen je recimo učni korpus ssj500k[3], ki temelji na obeh učnih korpusih izdelanih v okviru projekta JOS (Jezikoslovno označevanje slovenščine)[4]. Del orodja je tudi program LemmatizeWrapper, ki glede na vhodni lematizacijski model in pojavnico vrne lemo. Program smo razširili tako, da lahko sprejme tudi več vhodnih pojavnic.

Zaporedje 4.3: Glavni del razširjenega LemmatizeWrapper programa.

```
1  int main(int argc, char **argv)
2  {
3    if (argc < 3) {
4      cout << "Usage: lemmatizeWrapper model [word1 word2, ...]\n";
5      exit (1);
6    } else {
7      LoadLemmatizer(argv[1]);
8      for (int i = 2; i<argc; i++) {
9        cout << Lemmatize(argv[i]) << " ";
10     }
11     cout << "\n";
12     exit (0);
13   }
14 }
```

LemmatizeWrapper lahko kličemo iz skripne kode s pomočjo ukaza `exec`, ki vrne rezultat lematizacije in tudi izhodni status (*angl. status code*). Glede

[3]http://www.slovenscina.eu/tehnologije/ucni-korpus

[4]http://nl.ijs.si/jos

Beseda	1.0	0.98	0.96	0.94	0.92	...
računalništvo	informatika	računalnik	tehnologija	tehnika	umetna inteligenca	...
biologija	živoslovje	bioznanost	mikrobiologija	ekologija	botanika	...
sodelovanje	kooperacija	kolaboracija	zbližanje	zastopanje	vpletenost	...
eksperiment	poizkus	poskus	raziskava	pomerjanje	testiranje	...
...	...	...	...	...	...	...

Tabela 4.1: Tabela uteženih sorodnih besed.

na to, da je dolžina besedila omejena na največ 300 znakov, lahko vse pojavnice lematiziramo z enkratnim klicem.

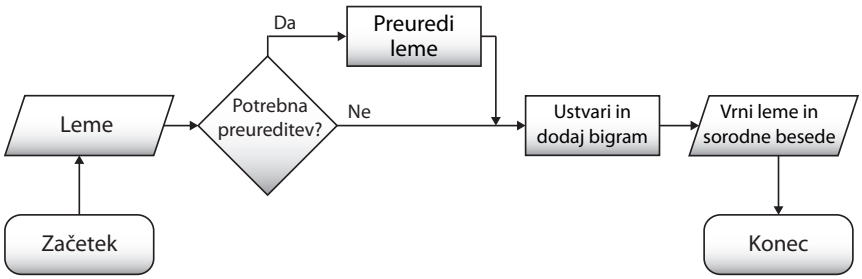
Zaporedje 4.4: Lematizacija s pomočjo orodja LemmaGen

```
1  // lemmatize
2  $output = null;
3  $return_var = null;
4  exec('lemmagenWrapper sl.bin '.implode(' ', $tokens), $output, $ret_var);
5  // check status code
6  if ($ret_var == 0) { // success
7    $output = mb_strtolower($output[0], 'UTF-8');
8  } else { // failure (wrong input)
9    $output = '';
10 }
11 // tokenize
12 preg_match_all('/[^\s]+/ui', $output, $lemmas);
```

Izhodni status nam pove, ali se je proces zaključil uspešno ali pa je prišlo do napake. V primeru neničelnega statusa moramo ustrezno ukrepati. Rezultat lematizacije so po vrsti zapisane leme pojavnic, ki jih moramo še razbiti v urejeno množico, kar lahko ponovno izvedemo s pomočjo regularnih izrazov.

4.3.4 Semantična podobnost

Urejeno množico lem lahko tudi razširimo s sopomenkami, podpomenkami, nadpomenkami oziroma s semantično podobnimi besedami. V poglavju 3.1.5 omenjen tezaver smo uporabili za iskanje sopomenk, orodje SloWTool pa



Slika 4.4: Diagram poteka funkcije `getRelatedWords`.

za ugotavljanje morebitnih podpomenk ali nadpomenk. Dobljene sorodne besede smo umestili v program za obdelavo razpredelnic Microsoft Excel in dobili strukturo prikazano v tabeli 4.1. Vrednosti uteži so zaenkrat enakomerno razporejene po stolpcih razpredelnice. Torej bolj kot se oddaljujemo od osnovne leme, manjša je utež. To tudi pomeni, da je trenutna funkcija `maxSim` omenjena v poglavju 3.4 zelo enostavna in bi jo lahko izboljšali oz. uporabili enačbo 3.13, da bi bolj precizno ovrednotili relacije med semantično podobnimi besedami. Razpredelnico smo nato uvozili v nerelacijsko podatkovno bazo v strukturi *rw* prikazani na sliki 4.3. Metodi iskanja in ujemanja uporabljata funkcijo `getRelatedWords` za razširitev lem z morebitnimi semantično podobnimi besedami. Omenjena funkcija, katere diagram poteka je prikazan na sliki 4.4, iz pojavnic ustvarja tudi besedni n-gram dolžine 2, torej bigram. N-gram je neprekinjeno zaporedje n elementov iz vhodnega zaporedja. V praksi to pomeni, da ustvarjamo zaporedne pare besed v vrstnem redu, kot ga najdemo v vhodnem besedilu. Če bi bil naš tekst le zaporedje "umetna inteligenca", bi bigrami bili "_umetna", "umetna inteligenca" ter "inteligenca_". Na tak način zaobjamemo tudi pogoste pare besed, kot je že zapisan par "umetna inteligenca", ter jim pripišemo sopomenko oziroma sorodno zvezo.

4.3.5 Oblikoslovno označevanje

Glede na to, da je dolžina objave omejena in so na nek način uporabniki prisiljeni zapisati le najpomembnejše podatke, oblikoslovno označevanje zaenkrat ni vključeno. Seveda pa bi lahko v ta namen uporabili v poglavju 3.1.6 omenjen program PosTaggerTag, ki za svoje delovanje potrebujejo zagonsko okolje .NET Framework 2.0. Ta s pomočjo modela oblikoslovnega označevanja označi vhodne besede in vrne označen korpus v formatu XML-TEI. Na ta način bi se torej lahko omejili le na samostalnike najdene v objavi, ostale besedne vrste pa bi ignorirali.

4.3.6 Klasifikacija

V namene klasifikacije objav smo izdelali naivni Bayesov klasifikator, ki deluje po principu opisanem v poglavju 3.1.8. Množenje velikega števila majhnih verjetnosti v enačbi 3.5 lahko pripelje do podliva plavajoče vejice, tako da smo enačbo preoblikovali v

$$c_{NB} = \arg\max_{c_j \in C} \log_e \left(\frac{N_j}{N}\right) \cdot \sum_{i=1}^{n} \log_e \left(\frac{N_{ij} + 1}{M + \sum_{k=1}^{M} N_{kj}}\right). \tag{4.1}$$

Najprej smo seveda morali izbrati primerne razrede klasifikacije. Glede na to, da je bilo težko predvideti vsebino realnih objav oziroma nismo imeli zagotovljene obširne obstoječe baze podatkov, smo se omejili na bolj širša področja kot so šport, kultura, mediji, naravoslovje ipd. Te razrede lahko seveda naknadno razcepljamo ali združujemo, odvisno od izoblikovane statistike objav, pri čemer moramo ponovno zgraditi sam model klasifikacije z učenjem na značilkah razredov. Učno strukturo smo gradili podobno kot strukturo semantično podobnih besed. Vsakemu razredu smo pripisali čimveč značilnih oziroma tipičnih značilk. V množici značilk športa tako najdemo nazive različnih športov, od nogometa, košarke pa športnih pripomočkov, pogostih glagolov itd. Lahko pa seveda vsako bodočo objavo ročno označimo in vsebino dodamo v strukturo, kar pripomore k še bolj obširnemu in na-

tančnemu modelu. Psevdokodo učenja naivnega Bayesovega klasifikatorja prikazuje algoritem 1. Ta vrne vsebino besednjaka, torej leme učne množice, verjetnosti $P(c_j)$ ter verjetnosti $P(x_i|c_j)$. Vse te podatke shranjujemo v nerelacijski podatkovni bazi v strukturi *clsf* prikazani na sliki 4.3. Glede na to, da razred določamo na osnovi okolice besed objave, bi lahko dejali, da besedam na ta način pripisujemo tudi kontekst.

Algoritem 1 Učenje naivnega Bayesovega klasifikatorja

procedure TrainMultinomialNB(C, D)
  $M \leftarrow IzlusciBesednjak(D)$
  $N \leftarrow PrestejDokumente(D)$
  for $c_j \in C$ **do**
    $N_j \leftarrow PrestejDokumenteRazreda(D, j)$
    $prior[j] \leftarrow N_j/N$
    $besedilo_j \leftarrow ZdruziBesedilaVsehDokumentovRazreda(D, j)$
    for $k \in M$ **do**
      $N_{kj} \leftarrow SteviloPojavitev(besedilo_j, k)$
      for $k \in M$ **do**
        $condprob[j][k] \leftarrow \frac{N_{ij}+1}{M+\sum_{k=1}^{M} N_{kj}}$
      end for
    end for
  end for
  return M, *prior*, *condprob*
end procedure

S pomočjo naučenega modela lahko še neoznačenim objavam določamo kontekst oz. razred klasifikacije. Postopek opisuje enačba 4.1.

4.4 Relevantno iskanje

V poglavju 3.2 smo omenili obrnjen indeks, ki za vsako lemo iz besednjaka indeksiranih dokumentov beleži, v katerih dokumentih se pojavlja. Struk-

tura B-drevesa, s pomočjo katere so obrnjeni indeksi pogosto realizirani, omogoča zelo hitro izvajanje osnovnih funkcij, ne pa tudi neposrednega merjenja relevantnosti. Pojem relevantnosti, bistvenosti ali pomembnosti smo omenjali skupaj s tf–idf uteževanjem. Globalni del enačbe 3.9 bi morali preračunavati vsakokrat, ko se v zbirki dokumentov pojavi nova objava. Enačba se namreč opira na velikost zbirke in na frekvence lem znotraj celotne zbirke. Pri ogromni velikosti zbirke lahko postane ponovno indeksiranje in preračunavanje globalnih vrednosti zelo počasno opravilo, kar lahko predstavlja tudi upočasnitev celotnega sistema. Zato smo za zagotavljanje učinkovitega relevantnega iskanja posegli po obstoječih rešitvah.

4.4.1 Sphinx

V jeziku C++ napisan odprtokodni strežnik, ki služi iskanju po celotnem besedilu, je bil zasnovan z namenom visoke zmogljivosti, relevance in enostavne integracije. Njegove glavne značilnosti so med drugim:

- visoka zmogljivost iskanja in indeksiranja,
- dokazana skalabilnost do več milijard dokumentov,
- na tisoče poizvedb na sekundo,
- boljše relevantno razvrščanje,
- enostavna integracija z izvori podatkov iz vrst SQL in XML.

Slika 4.5 prikazuje arhitekturo iskalnega strežnika. Aplikacija pošilja zahtevke prikritemu procesu (*angl. daemon*), ki s pomočjo indeksa lem opravi zahtevano opravilo ter vrne rezultat. Podatki posredovani aplikaciji so zgolj unikatni identifikatorji dokumentov skupaj z mero relevance, ki vsebujejo iskani niz besed. Zato je lahko potrebna še dodatna komunikacija s podatkovno bazo za prikaz vsebine najdenih dokumentov. Sphinx indeksira podatke, ki jih črpa iz podatkovne baze. Pri tem lahko dostopa neposredno do relacijske podatkovne baze MySQL ali pa črpa podatke posredno preko

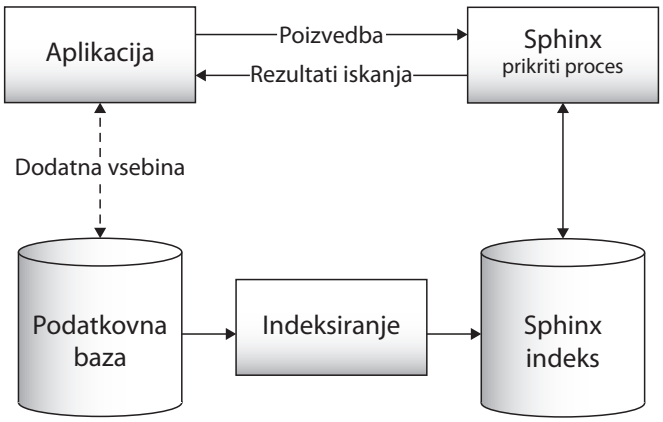
predložene strukture XML. Glede na to, da obdelane besede naravnega jezika objav oziroma leme shranjujemo v nerelacijski podatkovni bazi, moramo želeno indeksno vsebino iskalnemu strežniku posredovati v obliki XML oziroma XMLPipe2, ki je za Sphinx prirejen zapis XML. Primer tega zapisa prikazuje zaporedje 4.5.

Zaporedje 4.5: Zapis XMLPipe2 za primer, ko zbirka vsebuje en dokument.

```
<?xml version="1.0" encoding="UTF-8"?>
<sphinx:docset>
  <sphinx:schema>
    <sphinx:field name="t"/>
    <sphinx:field name="c"/>
    <sphinx:attr name="pid" type="int" bits="32"/>
    <sphinx:attr name="dt" type="int" bits="3"/>
    <sphinx:attr name="ut" type="int" bits="3"/>
    <sphinx:attr name="d" type="int" bits="24"/>
    <sphinx:attr name="md" type="timestamp"/>
    <sphinx:attr name="a" type="int" bits="1"/>
  </sphinx:schema>
  <sphinx:document id="54">
    <pid>42</pid>
    <dt>2</dt>
    <ut>2</ut>
    <d>7</d>
    <md>1354739079</md>
    <a>1</a>
    <t>naslov dokument</t>
    <c>kocka naš prihodnost ves generacija</c>
  </sphinx:document>
</sphinx:docset>
```

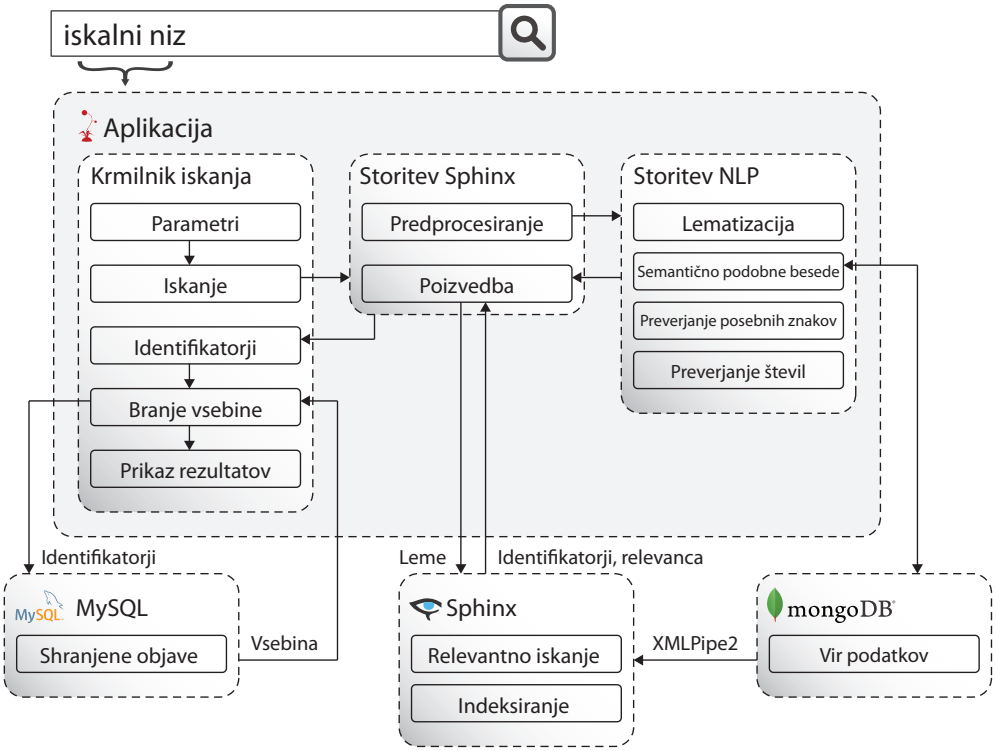
4.4.2 Potek iskanja

Iskanje se prične z vnosom iskalnega niza oziroma s poizvedbo. Parametri poizvedbe se validirajo in iskalni niz se preda storitvi, ki komunicira z iskalnim strežnikom Sphinx. Storitev nad vhodnimi besedami iskalnega niza



Slika 4.5: Arhitektura strežnika Sphinx v povezavi z aplikacijo.

izvede predprocesiranje, ki vključuje preverjanje posebnih znakov, lematizacijo, vključevanje semantično podobnih besed in preverjanje morebitnih števil v poizvedbi. Želeli smo, da bi sistem razpoznaval besede, ki vsebujejo šumnike, tudi brez strešic črk. Proces odstranjevanja črkovnih strešic imenujemo prečrkovanje. Pri tem se recimo črke č, š in ž prečrkujejo v črke c, s in z. Lematizacijo besed iskalnega niza (tudi prečrkovanih) izvede program LemmaGen skupaj z izdelanim lematizacijskim modelom. Lemam se nato pripišejo morebitne semantično podobne besede, ki jih skupaj z utežmi preberemo iz strukture *rw* nerelacijske podatkovne baze. Tako iskalni niz kot tudi objave lahko vsebujejo števila, pri čemer so ta zapisana ali z besedo ali s številko. Pri pretvarjanju števil pripisujemo pozitivnim celim številom do sto njihovo z besedo zapisano obliko in obratno. Ko so naloge predprocesiranja oz. obdelave naravnega jezika zaključene, se razširjena poizvedba pošlje iskalnemu strežniku. Ta izvede relevantno iskanje na osnovi stavčne bližine (*angl. phrase proximity*) s pomočjo mere LCS (*angl. Longest Common Sequence*) in na osnovi razvrščalne funkcije Okapi BM25, ki izvira iz tf–idf principa. Iskalni strežnik vrne razvrščene unikatne identifikatorje najdenih objav skupaj s pripadajočimi relevancami. Sledi branje same vsebine objav na osnovi identifikatorjev, ki je shranjena v relacijski podatkovni bazi,



Slika 4.6: Prikaz poteka iskanja, od poizvedbe do prikaza rezultatov.

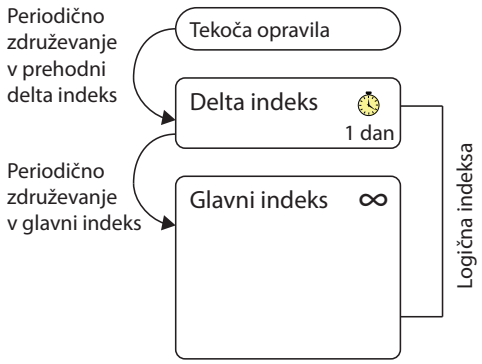
ter prikaz rezultatov. Opisan potek je prikazan tudi na sliki 4.6.

Primer dejanskega izpisa zadetkov iskanja je prikazan na sliki 4.7. Zaradi slabih rezultatov klasifikacije na račun povečini kratkih iskalnih poizvedb, metoda iskanja ne vsebuje procesa klasifikacije. Na sliki je vidno tudi delovanje prečrkovanja in preverjanja števil v iskalnem nizu. Številka 10 v prvem zadetku ustreza njeni z besedo zapisani obliki “deset” v sami poizvedbi. Poleg tega zaporedje črk “pomoc” v naslovu zadetka ustreza pravilnemu zapisu te besede, torej “pomoč”, v iskalnem nizu.

4.4.3 Osveževanje indeksa

Omenili smo, da lahko pri veliki zbirki podatkov naletimo na težavo počasnega ponovnega indeksiranja. Iskalni strežnik Sphinx rešuje ta problem tako, da

Slika 4.7: Primer rezultatov iskanja pri iskalnem nizu “oblikovanje pomoč deset”.



Slika 4.8: Rešitev problema indeksiranja z glavnim in pomožnim delta indeksom.

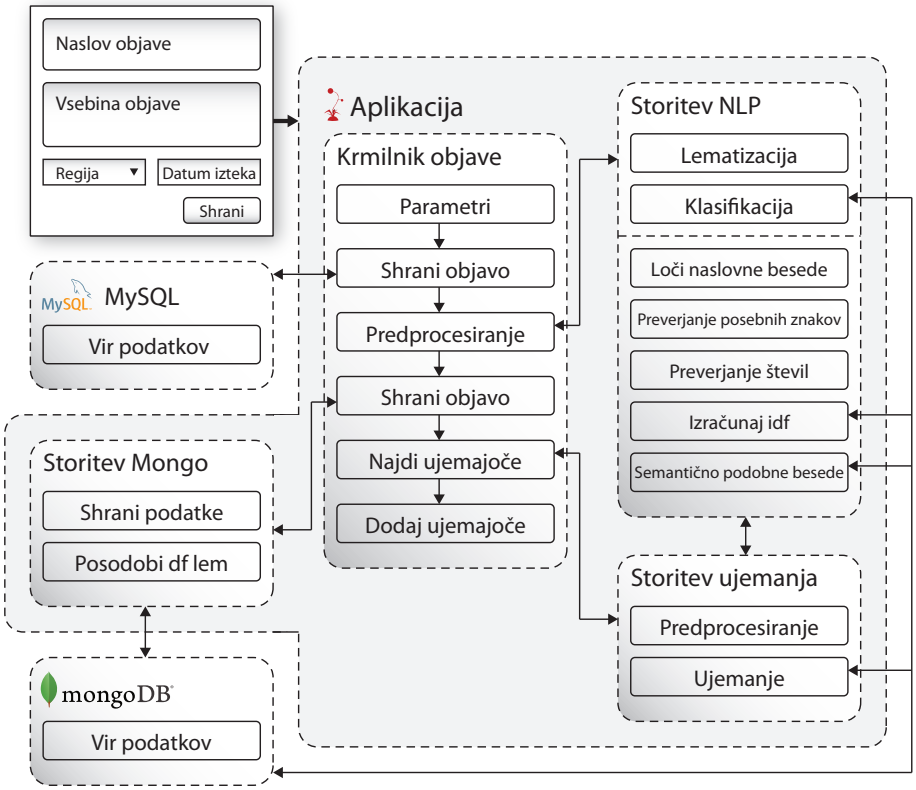
podatke kopiči v tako imenovanem glavnem indeksu in dodatnem pomožnem delta indeksu, kar prikazuje slika 4.8. Tekoča opravila, oziroma obdelane besede objav, se shranjujejo v nerelacijski podatkovni bazi. Periodično se na vsakih 5 minut izvršuje indeksiranje delta indeksa. Ko čas trajanja indeksiranja pomožnega indeksa že preseže razumno mejo, lahko prehodni indeks izpraznimo tako, da vsebino pridružimo glavnemu indeksu. Pri tem se zabeleži tudi točen čas, kdaj se je združevanje izvršilo, kar se upošteva pri nadaljnjem indeksiranju pomožnega indeksa in se tako uvozi le spremembe oziroma novosti. V našem primeru smo združevanje glavnega in pomožnega indeksa nastavili na enkrat dnevno. Lahko se seveda tudi zgodi, da uporabnik lastno objavo odstrani, ko je ta že zabeležena v glavnem indeksu. V tem primeru se objavo v glavnem indeksu označi kot neaktivno oziroma izbrisano in se jo pri iskanju ignorira. Kopičenje neaktivnih objav lahko rešujemo s ponovnim indeksiranjem glavnega indeksa.

4.5 Ujemanje dokumentov

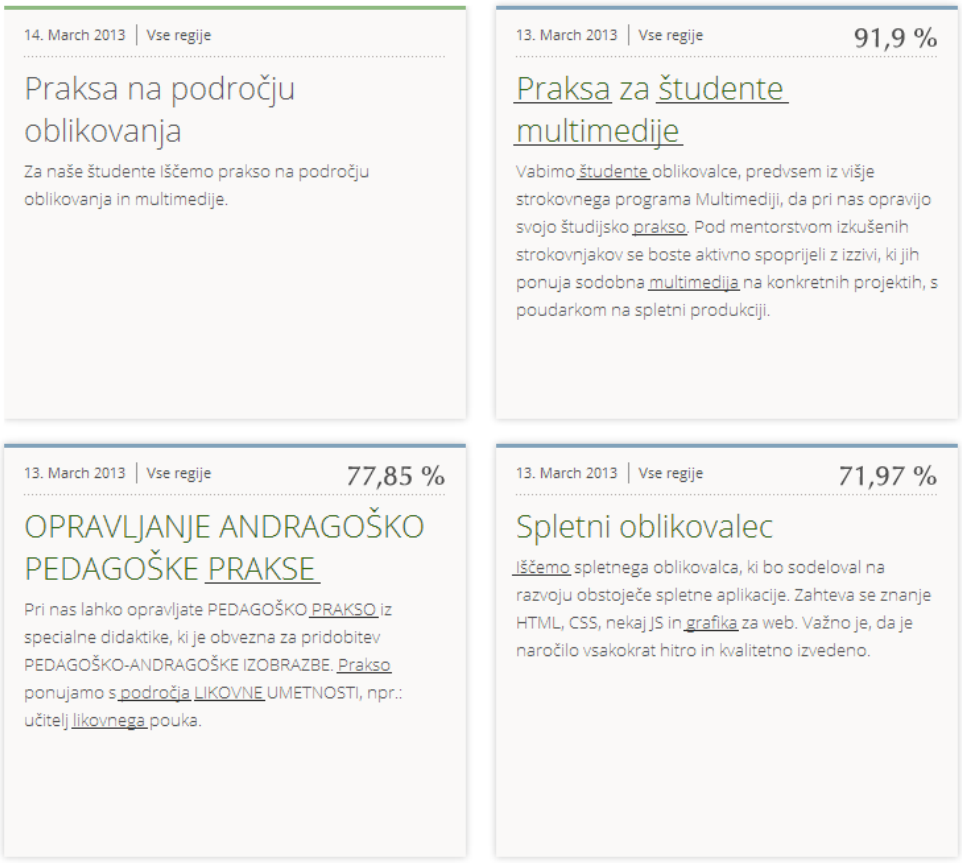
Proces ujemanja dokumentov prikazuje slika 4.9. Po vnosu obveznih podatkov v obrazec nove objave in odposlani zahtevi krmilnik objave prevzame

nadzor. Validirajo se parametri zahteve in sproži shranjevanje nove objave, pri čemer se podatki zapišejo v relacijsko podatkovno bazo. Sledi predprocesiranje naslova in vsebine objave, kjer se s pomočjo storitve NLP izvedeta lematizacija in klasifikacija. Slednja opravi postopek pripisovanja razredov s pomočjo enačbe 4.1 in podatkov v strukturi *clsf* podatkovne baze MongoDB. Obdelane besede naravnega jezika objave zapiše storitev Mongo v omenjeno nerelacijsko podatkovno bazo. Sledi postopek iskanja ujemajočih se objav, za kar je odgovorna storitev ujemanja. Ta najprej opravi predprocesiranje lem nove objave. Pri tem se loči leme, ki pripadajo naslovu in leme, ki so del vsebine objave. Naslovnim besedam smo namreč želeli pripisati večjo pomembnost. Nato se preverja prisotnost posebnih znakov, torej se izvaja prečrkovanje, kar smo že opisali pri metodi iskanja. Izračunati moramo tudi same uteži lem po prirejeni enačbi 3.10 s podatki iz strukture *df* in števcem dokumentov. Lemam se pripišejo tudi semantično podobne besede. Sledi ujemanje s pomočjo enačbe 3.13 in funkcije agregacije podatkovne baze MongoDB. Agregacija med drugim omogoča projektiranje, ujemanje, limitiranje, preskakovanje, odmotavanje, grupiranje in sortiranje. Po zaključku iskanja ujemajočih se objav pregledamo deleže ujemanja najdenih parov, pri čemer se omejimo na tiste, katerih ujemanje ali prileganje je vsaj 60 odstotno. Te nato dodamo k novi objavi kot najdene možne predloge.

Primer združevanja objav s pripisanim deležem ujemanja in označenimi značilkami je prikazan na sliki 4.10. Nosilni objavi je bil s pomočjo klasifikacije ugotovljen in pripisan razred “Umetnost”. Na predlogu oziroma objavi z naslovom “Spletni oblikovalec” opazimo, da oblike besede “oblikovalec” niso označene. Beseda se namreč ne pojavi na seznamu semantično podobnih besed za izraz “oblikovanje”. To nam je dalo slutiti, da moramo v semantični seznam vključiti čimveč primernih in s problemsko domeno povezanih pojavnic, saj je združevanje dokumentov od te strukture močno odvisno.



Slika 4.9: Prikaz poteka ujemanja, od shranjevanja objave do dodajanja ujemajočih se objav.



Slika 4.10: Primer združevanja objav s pripisanim deležem ujemanja in označenimi značilkami.

Poglavje 5

Zaključek in nadaljnje delo

Z diplomskim delom smo med drugim želeli pobližje spoznati osnovne naloge obdelave naravnega jezika. Ugotovili smo, da lahko kratka besedila dovolj zadovoljivo in zanesljivo tokeniziramo s pomočjo regularnih izrazov. Blokiranje lahko označimo za enostaven proces, saj je pomembna le dovolj obširna in natančna opredelitev pomensko šibkih besed, ki jih pri obdelavi izločujemo.

Za določanje indeksnih izrazov primernih za opisovanje vsebine dokumentov smo uporabili metodo lematizacije. Že izdelan slovenski binarni lematizacijski model projekta LemmaGen je povečini dobro opravil svojo nalogo. Model je zasnovan na podatkih Multext-East različice 3. Ker je na voljo tudi različica 4, smo v raziskovalne namene pridobili to nadgradnjo in z učenjem zgradili izpopolnjen binarni model, ki je odpravil nekatere pomanjkljivosti pravil že obstoječega binarnega modela. Za primerjavo smo izdelali še dodaten lematizacijski model s pomočjo učenja na podatkih iz učnega korpusa ssj500k, vendar so bili rezultati lematizacije slabši, čeprav korpus vsebuje več besednih oblik.

V namene ugotavljanja semantične podobnosti besed smo si zamislili strukturo shranjeno v nerelacijski podatkovni bazi. Izkazalo se je, da imajo semantično podobne besede v procesu združevanja objav velik pomen in da je naša struktura precej toga in ne omogoča enostavnega razširjanja podatkov. Zato bi bilo pri nadaljnjem delu potrebno strukturo preoblikovati in

predvsem omogočiti bolj natančno opredelitev ali ovrednotenje medbesednih relacij. Prav tako bi lahko v bodoče v obdelavo naravnega jezika vključili oblikoslovno označevanje in ugotavljali morebitne prednosti ali slabosti.

Ugotavljanje konteksta ima velik pomen pri razdvoumljanju večpomenskih besed. Kontekst objav smo določevali s pomočjo klasifikacije vsebine na osnovi naivnega Bayesovega klasifikatorja. Ta pristop se je izkazal za dobro odločitev. Natančnost klasifikacije je seveda odvisna od učne množice, ki bi jo lahko še bolj izpopolnili z vključevanjem bolj realnih že označenih vsebin objav in boljšo definicijo razredov klasifikacije. Poleg razdvoumljanja večpomenskih besed, prinaša klasifikacija prednost tudi pri samem ugotavljanju prileganja objav. Tukaj se lahko s pomočjo ugotovljenega najverjetnejšega razreda ali nekaj njih omejimo zgolj na podmnožico objav, kar pohitri potek agregacije podatkov. Metodo iskanja objav smo implementirali s pomočjo že obstoječe rešitve. Iskalni strežnik Sphinx se je dokazal kot izredno zmogljiv in tudi precej enostavno nastavljiv.

V splošnem lahko rečemo, da sta izvedbi ujemanja in iskanja objav iz poglavja 4 dobra osnova za nadaljnje izpopolnjevanje aplikacije. Presenetila nas je predvsem velika vloga oziroma pomembnost temeljito in natančno opravljenih nalog obdelave naravnega jezika, ki imajo vsekakor velik vpliv na uspešnost projekta.

Seznam slik

Seznam tabel

Literatura

[1] (2012) Uporaba informacijsko - komunikacijske tehnologije v gospodinjstvih in pri posameznikih, Slovenija, 2012 - končni podatki. Dostopno na: http://www.stat.si/novica_prikazi.aspx?id=5037

[2] (2011) Internet use in households and by individuals in 2011. Dostopno na: http://epp.eurostat.ec.europa.eu/cache/ITY_OFFPUB/KS-SF-11-066/EN/KS-SF-11-066-EN.PDF

[3] (2012) Internet access and use in 2011: Almost a quarter of persons aged 16-74 in the EU27 have never used the internet. Dostopno na: http://europa.eu/rapid/press-release_STAT-11-188_en.pdf

[4] M. Coles, H. Cotter, *Pro Full-Text Search in SQL Server 2008*, Apress, 2008, pogl. 1.

[5] (2012) Oracle, MySQL Internals Manual: Full-Text Search. Dostopno na: http://dev.mysql.com/doc/internals/en/full-text-search.html

[6] C. D. Manning, P. Raghavan, H. Schütze, *Introduction to Information Retrieval*, Cambridge University Press, 2008. Delno dostopno na: http://nlp.stanford.edu/IR-book/

[7] P. Vilar, J. Dimec (2000), *Krnjenje kot osnova nekaterih nekonvencionalnih metod poizvedovanja*, knjižnica, Ljubljana, letnik 44, številka 4 (2000), stran 7-31. Dostopno na: http://revija-knjiznica.zbds-zveza.si/Izvodi/K0004/Vilar.pdf

[8] J. Dimec, S. Džeroski, L. Todorovski, D. Hristovski, *Iskalnik za slovenske in angleške dokumente na svetovnem spletu*, 1999. Dostopno na: http://ibmi3.mf.uni-lj.si/ds/jez-teh-slo.html

[9] P. Vilar, J. Maver, *Krnjenje slovenskih besedil s področja bibliotekarstva*, knjižnica, Ljubljana, letnik 46, številka 1/2 (2002), str. 111-136. Dostopno na: http://www.dlib.si/details/URN:NBN:SI:DOC-CKREDCV0

[10] J. Dimec, *Avtomatsko opisovanje vsebine dokumentov na internetu*, COBISS Obvestila, 2000. Dostopno na: http://home.izum.si/cobiss/cobiss_obvestila/2000_4/Html/clanek_05.html

[11] M. Juršič, *Implementacija učinkovitega sistema za gradnjo, uporabo in evaluacijo lematizatorjev tipa RDR*, diplomsko delo, 2007. Dostopno na: http://lemmatise.ijs.si/Download/File/Documentation%23DiplomaThesis.pdf

[12] B. Lönneker, *Strojno oblikoslovno označevanje slovenskih besedil*, slavistična revija, letnik 53, številka 2 (2005), str. 193-210. Dostopno na: http://www.dlib.si/details/URN:NBN:SI:DOC-MFAZGQZL

[13] P. Vide Ogrin, *Digitalna knjižnica in njena orodja / Petra Vide Ogrin*, slovenska akademija znanosti in umetnosti, 2008. Dostopno na: http://www.sazu.si/files/file-82.pdf

[14] J. Jiang, D. Conrath (1997), *Semantic similarity based on corpus statistics and lexical taxonomy*, v zborniku Int'l. Conf. on Research in Computational Linguistics, str. 19-33. Dostopno na: http://arxiv.org/pdf/cmp-lg/9709008.pdf

[15] (2012) About WordNet. Dostopno na: http://wordnet.princeton.edu

[16] (2012) Oblikoskladenjske specifikacije. Dostopno na: http://nl.ijs.si/imp/msd/html-sl/

[17] R. A. Baeza-Yates, B. Ribeiro-Neto, *Modern Information Retrieval*, Addison-Wesley Longman Publishing Co., Inc., 1999, Boston, MA, USA. Dostopno na: http://people.ischool.berkeley.edu/ hearst/irbook/

[18] (2012) Wikipedia, Word-sense disambiguation. Dostopno na: http://en.wikipedia.org/wiki/Word-sense_disambiguation

[19] (2012) Wikipedia, Document classification. Dostopno na: http://en.wikipedia.org/wiki/Document_classification

[20] A. Khan, B. Baharudin, L. Hong Lee, K. Khan, *A Review of Machine Learning Algorithms for Text-Documents Classification*, v zborniku Advances in Information Technology (JAIT), letnik 1, številka 1, februar 2010.

[21] (2012) Wikipedia, Bag-of-words model.
Dostopno na: http://en.wikipedia.org/wiki/Bag-of-words_model

[22] Q. Yuan, G. Cong, N. Magnenat Thalmann, *Enhancing naive bayes with various smoothing methods for short text classification*, v zborniku 21st international conference companion on World Wide Web (WWW '12 Companion), ACM, 2012, New York, ZDA, strani 645-646. Dostopno na: http://doi.acm.org/10.1145/2187980.2188169

[23] J. Lin, C. Dyer, *Data-intensive text processing with MapReduce*, v zborniku Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Tutorial Abstracts (NAACL-Tutorials '09). Association for Computational Linguistics, Stroudsburg, 2009, ZDA, strani 1-2. Dostopno na: http://lintool.github.com/MapReduceAlgorithms/MapReduce-book-final.pdf

[24] (2012) Nginx Wiki. Dostopno na: http://wiki.nginx.org

[25] (2012) Agavi. Dostopno na: http://www.agavi.org

[26] (2012) MongoDB. Dostopno na: http://www.mongodb.org

[27] (2012) Binary JSON. Dostopno na: http://bsonspec.org

[28] R. Mihalcea, C. Corley, C. Strapparava, *Corpus-based and knowledge-based measures of text semantic similarity*, v zborniku 21st national conference on Artificial intelligence - izdaja 1 (AAAI'06), Anthony Cohn (Ed.), 2006, izd. 1, strani 775-780.

[29] (2012) About Sphinx.
Dostopno na: http://sphinxsearch.com/about/sphinx