UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Janez Cergolj

Spletni iskalnik po hierarhični zbirki obrtnikov

DIPLOMSKO DELO NA UNIVERZITETNEM ŠTUDIJU

MENTOR: doc. dr. Janez Demšar

Ljubljana, 2010

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01602/2009
Datum: 15.10.2009

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

Kandidat: **JANEZ CERGOLJ**

Naslov: **SPLETNI ISKALNIK PO HIERARHIČNI ZBIRKI OBRTNIKOV**
SEARCH ENGINE FOR HIERARCHICAL CATALOGUE OF CRAFTSMEN

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Sestavite spletni iskalnik po podatkovni bazi obrtnikov, pri čemer so posamezne obrti razvrščene v hierarhično urejene kategorije. Pri iskanju zadetkov in svetovanju morebitnih natančnejših poizvedb uporabljajte podano hierarhijo kategorij in znano zgodovino poizvedb.

Praktično implementacijo razvitih metod primerno preskusite, ovrednotite rezultate in jih primerjajte s podobnimi alternativnimi pristopi.

Mentor:

doc. dr. Janez Demšar

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Franc Solina

UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Janez Cergolj

Search Engine for hierarchical catalogue of craftsmen

DIPLOMSKO DELO NA UNIVERZITETNEM ŠTUDIJU

MENTOR: doc. dr. Janez Demšar

Ljubljana, 2010

Zahvala

Na tem mestu bi se rad zahvalil mentorju diplomske naloge doc. dr. Janezu Demšarju. Hvala ti, za ves trud, strokovno pomoč, nasvete, usmeritve in čas, ki si si ga vzel zame.

Hvala tudi moji družini, ki mi je ves ta čas študija stala ob strani in me podpirala.

Kazalo

Kratice, okrajšave, simboli

a. TF-IDF (term frequency–inverse document frequency): formula, ki se uporablja za uteževanje besed v besedilih.

b. PWN (Princeton WordNet): prvi semantični leksikon razvit v 80 letih prejšnjega stoletja na univerzi Princeton. Razvil ga je George A. Miller za angleški jezik. »Wordnet je leksikalna podatkovna zbirka, ki vsebuje samostalnike, glagole, pridevnike in prislove.« (Darja Fišer, 2009) [6].

c. RDR (Ripple down rule): postopek za lematizacijo, ki temelji na uporabi pravil.

d. Clog: razširjeni postopek RDR, pri tvorjenju pravil se upošteva tudi oblikoslovno oznako besede.

e. HTML (Hypertext Markup Language): je označevalni jezik, ki se uporablja za spletno predstavitev dokumentov, predvsem spletnih strani.

Povzetek

Internet in spletne tehnologije omogočajo hiter prenos in enostavno iskanje informacij. V ta namen so se razvile različne spletne strani, ki ponujajo enostavno iskanje le-teh. Nekatere so bolj specializirane od drugih. Pred približno letom dni je nastala spletna stran www.najdiobrtnika.si. Njen namen je zbrati čim več samostojnih podjetnikov, ki bodo na spletni strani predstavili svojo dejavnost. Obiskovalci pa bodo lahko preko iskalnika poiskali želenega samostojnega podjetnika. Po drugi strani samo hranjenje oz. podajanje informacije samo po sebi še ni dovolj. Pomembna je tudi uporabniška izkušnja, ki zahteva, da so informacije predstavljene pregledno in da je dostop do njih kar se da enostaven.

V diplomski nalogi smo skušali izboljšati uporabniško izkušnjo, ki smo jo v tem primeru omejili na iskanje samostojnih podjetnikov glede na iskalni niz, ki ga vpiše obiskovalec spletne strani. Po drugi strani smo želeli beležiti in vrednotiti iskalne nize obiskovalcev in jih zapisati v dnevnik. Na podlagi v dnevnik vpisanih iskalnih nizov smo želeli obiskovalcu ponuditi pomoč pri iskanju, tako da smo mu predlagali sorodne iskalne nize. Poleg predlaganih iskalnih nizov obiskovalcu tudi ponudimo, da pregleda podkategorije, ki so najbolj zastopane med rezultati – predstavitvami samostojnih podjetnikov. Dodatno širino rezultatov smo skušali doseči tudi z uporabo slovenskega semantičnega leksikona, s katerim smo želeli poiskati sopomenke za besede iskalnega niza. Sopomenke bi nato pripeli iskalnemu nizu in jih tako kot sam iskalni niz uporabili kot ključne besede pri iskanju.

Z uporabo zgoraj opisanih postopkov smo želeli, da bi obiskovalec spletne strani hitreje in z manj truda prišel do podatkov želenega samostojnega podjetnika.

KLJUČNE BESEDE: lematizacija, sloWNet, odkrivanje zakonitosti iz besedil, spletni iskalniki

Abstract

Internet and web services are capable of providing vast amounts of information and offer efficient search over it. A number of more or less specialized search engines have evolved over time for this purpose.

The web page www.najdiobrtnika.si was developed about a year ago. Its objective is to gather as much craftsmen as possible and present information on them. The basic goal of the page is to provide a user friendly search engine, designed to search for these craftsmen. On the other hand, only storing and deploying information is not enough. User experience has become more important than ever: the information should be presented clearly and access to it should be simple and intuitive.

Our goal is to improve the user experience on this web page, in particular the craftsmen search engine regarding to user's search queries. We record and evaluate search queries in a log. We use the log to assist in further users' searches by suggesting similar search queries. Besides suggesting related search queries, we provide relevant subcategories which are most common within the results of the given query. We also attempt to provide automatic search for search query synonyms with the use of Slovene wordnet. The main idea was to append synonyms to user's original search query of user and use it altogether as key words for searching.

KEY WORDS: lemmatization, sloWNet, text mining, search engines

1. Uvod

Spletna stran najdiObrtnika.si je zagledala luč spomladi leta 2009. Primarna ideja, ki se bo v prihodnosti še razvijala, je na enem mestu zbrati in pregledno predstaviti obiskovalcu spletne strani registrirane slovenske samostojne podjetnike in obrtnike. Dasiravno je v spletu moč najti kar nekaj podobnih projektov, ki želijo predstaviti vsa slovenska podjetja, sem tudi sam želel ponuditi podobno storitev. Spletna stran najdiObrtnika.si je iskalnik že registriranih samostojnih podjetnikov in samostojnih podjetnikov na spletni strani glede na njihove obrti in dejavnosti ter glede na predstavitev njihove dejavnosti oz. obrti. Na sliki 1 je prikazana domača stran skupaj z iskalnikom samostojnih podjetnikov.

Slika 1: Domača stran najdiObrtnika.si skupaj z iskalnikom.

Osnovna storitev, ki jo omogoča spletna stran najdiObrtnika.si, je iskanje in predstavitev samostojnih podjetnikov ter njihove dejavnosti glede na iskalni niz, ki ga vnese obiskovalec. Obiskovalec lahko omeji iskanje glede na regijo, iz katere prihaja samostojni podjetnik. Po drugi strani lahko obiskovalec spletne strani išče po posameznih kategorijah.

V diplomski nalogi smo želeli izboljšati uporabniško izkušnjo, ki smo jo omejili na iskanje samostojnih podjetnikov v bazi glede na iskalni niz, ki ga vpiše obiskovalec, ki želi z njim poiskati določenega samostojnega podjetnika ali skupino samostojnih podjetnikov. Cilj diplomske naloge je bilo izdelati algoritem, ki bo obiskovalcu glede na vpisani iskalni niz vrnil oz. predstavil tiste samostojne podjetnike, ki se najbolje ujemajo z vpisanim iskalnim

nizom obiskovalca. Rezultat diplomske naloge je, da obiskovalec hitreje in z manj truda pride do želenega samostojnega podjetnika.

Pred diplomsko nalogo je bilo iskanje in razporejanje samostojnih podjetnikov zastavljeno tako, da se je v naključnem vrstnem redu izpisalo samostojne podjetnike glede na to, koliko besed iskalnega niza obiskovalca vsebujejo, in sicer na sledeč način: algoritem je preštel besede iskalnega niza v nazivu samostojnega podjetnika in opisu njegove dejavnosti za vsakega samostojnega podjetnika. Samostojni podjetniki, ki so imeli v svojem nazivu ali predstavitvi dejavnosti dovoljkrat omenjene besede iskalnega niza, so bili predstavljeni obiskovalcu.

Prvi hip se zdi, da bi lahko za rešitev opisanega problema posegli po nekaterih rešitvah, ki jih uporabljajo uspešni svetovni spletni iskalniki – v prvi vrsti Google. Spletni iskalniki glede na iskalni niz, ki ga vpiše obiskovalec, vrnejo rezultate iskanja – seznam spletnih strani oz. drugih spletnih dokumentov. Ker besede iskalnega niza navadno vsebuje več spletnih strani, si je potrebno pomagati z dodatno klasifikacijo relevantnosti teh spletnih strani. Google v ta namen uporablja tako imenovani algoritem PageRank. Relevantnost spletne strani se pri tem algoritmu določi glede na število vhodnih povezav na spletno stran in glede na pomembnost spletne strani, ki nudi posamezno vhodno povezavo [1].

Zgoraj opisana rešitev v našem primeru ne pride v poštev. Če naredimo analogijo med samostojnimi podjetniki in spletnimi stranmi, vidimo, da ne obstajajo spletne povezave med samostojnimi podjetniki, tako kot pri spletnih straneh. Zaradi tega na podlagi algoritma PageRank ne moremo določiti bolj ali manj relevantnih samostojnih podjetnikov glede na iskalni niz.

Namesto tega bomo nalogo reševali s pomočjo hierarhije dejavnosti, ki nam omogoča določiti samostojne podjetnike, ki sicer ne ustrezajo danemu iskalnemu nizu, ki ga vpiše obiskovalec, vendar lahko glede na položaj v hierarhiji dejavnosti in glede na položaj teh, ki nizu ustrezajo, sklepamo, da bi obiskovalca utegnili zanimati.

Reševanje zastavljenega problema smo razdelili na sklope:

1. lematizirati iskalni niz, naziva samostojnega podjetnika in opisa njegove dejavnosti;
2. določiti relevantnost samostojnih podjetnikov glede na iskalni niz z formulo TF-IDF;
3. zapisati iskalne nize v dnevnik in predlaganje za obiskovalca potencialno zanimivih iskalnih nizov, ki so že bili zapisani v dnevnik;
4. predlagati obiskovalcu podkategorije, ki so najbolj zastopane med zadetki;
5. uporabiti slovenski semantični leksikon za iskanje sopomenk.

V prvem poglavju, Uvod, je najprej predstavljena spletna storitev in stran najdiObrtnika.si. Poglavje se nadaljuje s predstavitvijo ciljev diplomske naloge, in sicer izdelati algoritem za boljše iskanje samostojnih podjetnikov in nuditi pomoč obiskovalcem pri iskanju. Poglavje se nadaljuje s predstavitvijo pretekle rešitve iskanja na spletni strani in razjasni, zakaj znani algoritem PageRank ne bi bil primerna rešitev. Poglavje se zaključi s predstavitvijo, kako smo se v tej nalogi lotili reševanja zastavljenega problema.

V drugem poglavju, Teoretični del, smo predstavili teoretične osnove diplomske naloge. V njem predstavimo dva postopka za lematizacijo: RDR in Clog. Nadaljujemo s predstavitvijo načina za uteževanje besed, in sicer s formulo TF-IDF. Poglavje zaključimo s teorijo semantičnih leksikonov.

V tretjem poglavju, Reševanje problema, najprej predstavimo, kako smo lematizirali nazive in opise dejavnosti samostojnih podjetnikov. Nadaljujemo z ocenjevanjem relevantnih zadetkov s formulo TF-IDF, predstavimo dnevnik iskalnih nizov in na kakšen način obiskovalcem predlagamo nove iskalne nize. Poglavje se zaključi s predstavitvijo, kako obiskovalcu predlagamo med zadetki najbolj zastopane podkategorije.

V četrtem poglavju, Opis okolja in aplikacije, si pogledamo, s katerimi orodji smo razvili aplikacijo, in na kratko predstavimo platformo, na kateri teče aplikacija. Nadaljujemo s predstavitvijo tabel, ki smo jih ustvarili in uporabili pri diplomski nalogi. Poglavje zaključimo s predstavitvijo ene izmed funkcij, ki smo jo uporabili v diplomski nalogi.

V petem poglavju, Rezultati diplomske naloge, smo si zastavili problem – primer poizvedbe – in na njem predstavili delovanje algoritma ter njegove rezultate.

2. Teoretični del

V tem poglavju je predstavljena lematizacija in dva postopka zanjo: RDR in Clog. Poglavje se nadaljuje s predstavitvijo spletne storitve na naslovu [8] za lematizacijo in njenih rezultatov v tabeli [1]. Sledi opis načina za uteževanje besed. Uteževanje besed smo izvedli s formulo TF-IDF. Osnovna ideja je, da ima vsaka izbrana beseda v izbranem besedilu svojo vrednost, ki je odvisna od pogostosti besede v izbranem besedilu v primerjavi s pogostostjo besede v celotni zbirki besedil. Poglavje se zaključi s predstavitvijo semantičnih leksikonov. Predvsem smo se posvetili slovenskemu imenovanemu sloWNet avtorice Darje Fišer.

Lematizacija

Lematizacija je postopek, pri katerem besedi določimo njeno normalizirano (osnovno) obliko, imenovano lema. Primer: hodimo → hoditi. Lematizacija se izvaja glede na vnaprej določena pravila. Pravila se lahko določi ročno ali s pomočjo strojnega učenja. V prvem primeru jih izdela človek, v drugem primeru pa omogočimo, da se lematizator (program za lematizacijo) na množici že lematiziranih besed nauči oz. določi pravila za lematizacijo [5]. Pri strojnem učenju kot vhodni parameter nastopa množica parov. Par sestavljata beseda in njena lematizirana oblika. Rezultat je množica pravil, ki se iz vhodnih parov nauči, kako besedo pretvoriti v lematizirano besedo [3]. V diplomski nalogi smo lematizacijo uporabili za normalizacijo iskalnega niza, ki ga je vpisal obiskovalec v iskalno vrstico, in za normalizacijo naziva samostojnega podjetnika in opisa njegove dejavnosti. Z normalizacijo smo preprečili, da bi zaradi morfološke raznolikosti besed spregledali ključne besede, ki jih predstavlja vpisani iskalni niz, v nazivu in opisu dejavnosti posameznega samostojnega podjetnika. Vzemimo primer, da obiskovalec v iskalno vrstico vpiše niz *iščem vodovodarja*. Brez lematizacije bi obiskovalec kot relevantne samostojne podjetnike dobil samo tiste, ki imajo v svojem nazivu ali opisu dejavnosti točno ti dve besedi, *iščem* in *vodovodarja*. Z lematizacijo se iskalni niz pretvori v *iskati* in *vodovodar*. Ker so tudi nazivi samostojnih podjetnikov in njihovi opisi lematizirani, ni morfološke raznolikosti besed. Zato iskanje ključnih besed v nazivu in predstavitvi samostojnih podjetnikov da boljše rezultate.

Za potrebe diplomske naloge smo uporabili lematizator na spletni strani http://nl2.ijs.si/analyze/, ki uporablja postopka Ripple down rule in Clog. Razvili so ga na Institutu Jožef Stefan na oddelku za Tehnologije znanja, pod vodstvom Joëla Plissona.

Pri RDR je učna množica, sestavljena iz besed, ki jih opisujejo njihove končnice, in preslikav končnic besed v končnice njene lematizirane besede. Zaradi tega so pravila preslikav bolj natančna. Preslikava preslika končnico izbrane besede v končnico lematizirane besede. Primer: avtomehanik**i** → avtomehanik, preslikava končnic je sledeča i → _. Pravila RDR so predstavljena v obliki drevesne strukture, in sicer kot if, then, except – else stavki. Ponavadi pravila vsebujejo tudi because of, ki pojasnjuje except in else stavke. Avtor [3] navaja sledeč primer:

***if** V **then** _to_ **because of** [BRESKEV, POSTAVITEV]*

***except if** HTEV **then** _toA **because of** [ZAHTEV]*

Clog se uči odločitvenih seznamov prvega reda. Glavna razlika v primerjavi s postopkom RDR je, da Clog poleg para besede in njene lematizirane oblike vsebuje tudi oblikoslovno oznako. »Oblikoslovne oznake podajo besedno vrsto ter množico drugih oblikoskladenjskih lastnosti besed; tako npr. oznaka Ncmpi pomeni PoS: Noun, Type: common, Gender: masculine, Number: plural, Case: instrumental, oziroma Samostalnik, vrsta: občno ime, spol: moški, število: množina, sklon: orodnik« [2]. Iz učnega korpusa se vzame besedno obliko (bo), ki se jo želi lematizirati, lematizirano besedo (le) in oblikoslovno oznako (oo). Vsaka taka trojka je primer analize v obliki oo(bo, le), pri čemer je bo vhodni in le izhodni argument [2]. Lematizator Clog se tako nauči pravila za vsako oblikoslovno oznako posebej in je zato tudi bolj uspešen pri lematizaciji kot RNR postopek. Spodaj vidimo primer Clog pravila za samostalnike prve ženske sklanjatve, »kjer končnico leme –a nadomesti v rodilniku ednine končnica -e« (Sašo Džeroski, 2000) [2]. Glavna slabost je, da je potrebno pred lematizacijo uporabiti oblikoslovno označevanje.

Primer Clog pravila iz dela [2]:

n0fsg(A, B) :-

rule(A, B, [e], [a]), ! .

Za preverjanje uspešnosti RDR in Clog pristopa so avtorji [3] izvedli dva poizkusa. Prvi poizkus so izvedli za RDR postopek, in sicer z uporabo MULTEXT-East leksikona, ki vsebuje 557970 različnih besednih oblik. RDR je z učno množico besed, ki jo je predstavljal MULTEXT-East leksikon, izdelal pravila. Njegovo natančnost so preverili na besedah iz tega korpusa. Najboljša natančnost, ki so jo dosegli, je bila 93.2% (±0,1), ki so jo določili z uporabo 5-kratnega prečnega preverjanja, v primerjavi s preteklim delom (Mladenić, 2002; Džeroski in Erjavec 2004) [3]. Drugi poizkus je obsegal oba pristopa tako RDR kot Clog. Izvedli so ga s pomočjo MULTEXT-East leksikona in z uporabo IJS-ELAN korpusa. IJS-ELAN korpus (Erjavec, 1999) [3] je sestavljen iz 15 terminološko bogatih besedil in vsebuje milijon besed. Polovico besedil je slovenskih in polovico angleških. Za določitev RDR in Clog pravil so uporabili MULTEXT-East leksikon. Nato so testirali pravilnost RDR pravil na delu IJS-ELAN korpusa, ki je vseboval tako poznane kot nepoznane besede. Poleg tega so izvedli dodatno testiranje na podmnožici, ki je vsebovala 10315 nepoznanih besed. To množico besed so avtorji [3] dobili tako, da so iz IJS-ELAN korpusa odstranili besede, ki so

se nahajale tudi v MULTEXT-East leksikonu. Zato to množico besed imenujejo nepoznane besede, saj jih RDR ni srečal med učno množico. Tako so ustvarili težje pogoje za lematizator. Clog pravila so bila določena s pomočjo MULTEXT-East leksikona. Pravila so bila nato testirana na podmnožici besed IJS-ELAN korpusa. Testne besede niso bile del učne množice. Rezultati so bili sledeči: pristop RDR je dosegel 90,0% natančnost pri poznanih in nepoznanih besedah skupaj in 87,6% natančnost pri nepoznanih besedah. Pristop Clog je dosegel 92,0% natančnost pri nepoznanih besedah [3].

Oglejmo si še primer lematizacije. Kot originalno besedilo smo vezli naziv in opis dejavnosti samostojnega podjetnika: *IZDELAVA NAGROBNIH SPOMENIKOV IN CEMENTNIN VINKO KRAMAR S.P. Izdelujemo nagrobne spomenike, okenske police, stopnice, mize in ostale izdelke iz naravnega kamna. Izdelke izdelujemo tudi po želji in želenih merah.* Lematizacijo smo izvedli s pomočjo spletnega servisa za lematizacijo na spletni strani http://nl2.ijs.si/analyze/ in rezultat prikazali v tabeli [1].

Originalna beseda	oblikoslovna oznaka	lematizirana beseda
IZDELAVA	Sozei	izdelava
NAGROBNIH	Ppnmmr	nagroben
SPOMENIKOV	Sommr	spomenik
IN	Vp	in
CEMENTNIN	Sozmr	cementnina
VINKO	Slmei	vinko
KRAMAR	Slmei	kramar
S.P.	O	s.p.
Izdelujemo	Ggnspm	izdelovati
nagrobne	Ppnmmt	nagroben
spomenike	Sommt	spomenik
,	,	,
okenske	Ppnzmt	okenski
police	Sozmt	polica
,	,	,
stopnice	Sozmt	stopnice
,	,	,
mize	Sozmt	miza
in	Vp	in
ostale	Ppnmmt	ostal
izdelke	Sommt	izdelek
iz	Dr	iz
naravnega	Ppnmer	naraven
kamna	Somer	kamen
.	.	.
Izdelke	Sommt	izdelek
izdelujemo	Ggnspm	izdelovati
tudi	L	tudi
po	Dm	po
želji	Sozem	želja
in	Vp	in
želenih	Pdnzmr	želen
merah	Sozmm	mera
.	.	.

Tabela 1: Primer lematizacije.

Uteževanje besed

Ena ključnih stvari pri diplomski nalogi je poiskati besede vpisanega iskalnega niza obiskovalca v bazi samostojnih podjetnikov, in sicer v nazivu samostojnega podjetnika in njegovem opisu dejavnosti, ter določiti, kako pomembne so te besede pri določenem samostojnem podjetniku.

V ta namen smo uporabili formulo TF-IDF (term frequency – inverse document frequency), ki je namenjena uteževanju besed v besedilih. Pri TF-IDF najprej določimo, kolikokrat se izbrana beseda pojavi v izbranem dokumentu, in to delimo s številom vseh besed v izbranem dokumentu (ang. term frequency – TF). Nato določimo še, v koliko različnih dokumentih se izbrana beseda pojavlja (ang. inverse documetn frequency – IDF). IDF predstavlja utež za TF. Bolj kot je beseda zastopana v dokumentih, manjšo vrednost ima IDF in posledično je tudi beseda manj pomembna v izbranem dokumentu. Produkt TF * IDF je vrednost, ki nam pove, kako je zastopana izbrana beseda v izbranem dokumentu v odvisnosti od zastopanosti te besede v celotnem korpusu dokumentov. Besede, ki so pogoste v izbranem dokumentu, ne pa v celotni zbirki dokumentov, imajo večjo TF-IDF vrednost kot besede, ki so pogoste v vseh dokumentih.

Vzemimo, da imamo iskalni niz w in posamezne besede iskalnega niza w_i. Podano imamo tudi zbirko dokumentov D in posamezen dokument d ϵ D. $f_{wi,d}$ predstavlja vrednost, ki pove, kolikokrat se beseda w_i pojavi v dokumentu d deljeno vse besede dokumenta d. $f_{wi,D}$ predstavlja število dokumentov, v katerih se beseda w_i pojavi. Formula [1] podaja vrednost formule TF-IDF za besedo i.

$$w_{id} = f_{wi,d} * \ln\left(\frac{|D|}{f_{wi,D}}\right)$$

Formula 1: Formula za izračun TF-IDF vrednosti za posamezno besedo w_i.

Končno vrednost formule TF-IDF prikazuje formula [2], ki predstavlja vsoto posameznih vrednosti za besede w_i, kjer i predstavlja število besed v iskalnem nizu [4].

$$w_d = \sum_i w_{id}$$

Formula 2: Končna vrednost TF-IDF algoritma je vsota TF-IDF vrednosti za posamezne besede.

Primer: Podane imamo spodnje podatke:

	dokument 1	dokument 2
št. vseh besed v dokumentu	10	20
pojavitve beseda *omara*	1	8

pojavitve besede *mizarstvo*	2	0
$f_{omara\ ,d}$	$\frac{1}{10} = 0{,}1$	$\frac{8}{20} = 0{,}4$
$f_{mizarstvo\ ,d}$	$\frac{2}{10} = 0{,}2$	$\frac{0}{20} = 0$

$|D| = 2, f_{omara\ ,D} = 2, f_{mizarstvo\ ,D} = 1$

Določimo TF-IDF posebej za besedo *omara* in posebej *mizarstvo*, ter nato TF-IDF za *omara mizarstvo* skupaj.

a) Uporabimo **formulo [1]**:

$$w_{omara\ ,1} = 0.1 * \ln\frac{3}{2} = 0{,}0406,\ w_{omara\ ,2} = 0.4 * \ln\frac{3}{2} = 0{,}1622$$

$$w_{mizarstvo\ ,1} = 0.2 * \ln\frac{3}{1} = 0{,}2197,\ w_{mizarstvo\ ,2} = 0 * \ln\frac{3}{1} = 0$$

b) V tem primeru moramo sešteti vrednosti za oba dokumenta. Uporabili smo **formulo [2]**:

$$w_1 = w_{omara\ ,1} + w_{mizarstvo\ ,1} = 0{,}2603$$

$$w_2 = w_{omara\ ,2} + w_{mizarstvo\ ,2} = 0{,}1622$$

V prvem primeru bi bil za besedo *omara* dokument 2 relevantnejši kot dokument 1, za besedo *mizarstvo* bi bila situacijo obrnjena. V drugem primeru pa ima dokument 1 večjo težo oz. je bolj relevanten za izbrani iskalni niz.

Semantični leksikon

Semantični leksikoni spadajo v skupino semantičnih zbirk in predstavljajo način, kako človeški mentalni leksikon, ki je skupek pomenov besed in njihovih povezav z drugimi pomeni besed, prevesti v strojnega (računalniku razumljivega). Semantični leksikoni pri računalniški obdelavi naravnega jezika predstavljajo most med jezikom in znanjem, ki ga z jezikom izražamo. Glede na pristop pri gradnji jih delimo na pristope, ki se ukvarjajo s semantičnimi lastnostmi, in na leksikalne semantične mreže. Pri pristopih, ki se ukvarjajo s semantičnimi lastnostmi, so besede v leksikonu povezane s svojimi lastnostmi. Pri leksikalnih semantičnih mrežah, kamor spada tudi wordnet [10], so besede med seboj povezane s pomenskimi razmerji.

Semantični leksikon temelji na združevanju besed z istim pomenom v pojme in povezavi teh pojmov glede na sorodnost z leksikalnimi in pomenskimi razmerji. V Sloveniji je v obliki doktorske disertacije Darja Fišer izdelala slovensko različico semantičnega leksikona, imenovano sloWNet. Pri izdelavi je izhajala iz predpostavke, da so prevodi verodostojen semantični vir in da je iz obstoječih virov mogoče izluščiti relevantne semantične informacije o slovenskih besedah. Gradnja semantičnega leksikona pomeni generiranje sopomenskih nizov. Pri gradnji slovenskega semantičnega leksikona je uporabila tri pristope – slovarskega, korpusnega in enciklopedičnega.

Pri slovarskem pristopu je bila osnovna ideja, da se je sopomenske nize (sinsete) srbskega semantičnega leksikona prevedlo v slovenščino. Pri tem se je glede na predpostavko razširitvenega pristopa oz. modela ohranila struktura PWN (Princeton WordNet). Bistvo razširitvenega modela je ohranjanje strukture referenčnega semantičnega leksikona, saj velja načelo: »če med pojmoma v referenčnem wordnetu velja neko razmerje, velja isto razmerje tudi med ekvivalentnima pojmoma v ciljnem wordnetu« (Vossen 1996, 716) [6].

Pri korpusnem pristopu je avtorica [6] uporabila vzporedni korpus SEE-ERA.NET, ki vsebuje nekaj manj kot 1.5 milijona besed v 8 jezikih. Za gradnjo slovenskega semantičnega leksikona je uporabila naslednje jezike: angleščino, češčino, romunščino, bolgarščino in slovenščino in njihove semantične leksikone. Najprej je bilo potrebno označiti in lematizirati korpus in nato narediti poravnavo korpusa na ravni besed, sledilo je luščenje dvojezičnih leksikonov. Če je avtorica v večjezičnem leksikonu našla vnos, ki se je ujemal s sopomenskim nizom, si je to zapomnila. Sopomenski niz oz. vnos je bil verodostojnejši, če je bil najden v večjem številu jezikov. Glavna prednost korpusnega pred slovarskim pristopom je avtomatsko razreševanje večpomenskih besed.

Pri enciklopedičnem pristopu je avtorica [6] uporabila večjezikovne vire Wikipedijo, Wikislovar, Wikivrste in Eurovoc za luščenje dvojezičnih leksikonov. Leksikone je avtorica izluščila glede na spletne povezave med članki na isto temo v obeh jezikih. Tako dobljene leksikone je avtorica primerjala z enopomenskimi literati iz PWN. V primeru ujemanja se je lahko zgradilo nov sopomenski niz za slovenski semantični leksikon. Avtorica je pri enciklopedičnem pristopu dobila največje število sopomenskih nizov.

Rezultati teh treh pristopov – sopomenski nizi – so osnovni gradniki semantičnega leksikona. Avtorica [6] nam je rezultat poslala v obliki xml datoteke. Pri diplomski nalogi smo želeli sloWNet uporabiti za iskanje sopomenk oz. besed z enakim pomenom (npr. slikopleskar = malar). Kasneje se je izkazalo, da sloWNet zaradi nezadostnega števila sopomenskih nizov ne pride v poštev.

3. Reševanje problema

V tem poglavju je predstavljeno delovanje aplikacije in na kakšen način smo se lotili reševanja zastavljenega problema diplomske naloge. Najprej predstavimo, kako smo se lotili lematizacije iskalnega niza obiskovalca ter naziv in opis dejavnosti samostojnih podjetnikov. Rezultate lematizacije shranimo v tabelo. Sledi predstavitev uteževanja besed v besedilih, ki jih obiskovalec vpiše v iskalni niz, in sicer s formulo TF-IDF. Besede iskalnega niza predstavljajo ključne besede, ki jih nato iščemo med nazivi in opisi dejavnosti samostojnih podjetnikov. Poglavje se nadaljuje z dnevnikom iskalnih nizov. Namen njihovega beleženja je možnost, da se naslednjim obiskovalcem, če vpišejo enak iskalni niz, pod določenimi pogoji predlaga nov iskalni niz. Poglavje zaključimo s predstavitvijo, na kakšen način obiskovalcem predlagamo podkategorije in katere so omejitve.

Poglejmo si, kaj se zgodi, ko obiskovalec vpiše iskalni niz. Predno besede iskalnega niza uporabimo za ključne besede iskanja po nazivu in opisu dejavnosti samostojnih podjetnikov z uporabo formule TF-IDF, je potrebno besede iskalnega niza, ki še nikoli niso bile lematizirane, lematizirati. Že v preteklosti lematizirane besede iskalnega niza so shranjene v tabeli, tako da njihovo lematizirano obliko preberemo iz tabele. Lematizacijo izvedemo s spletno storitvijo na spletni strani [8]. S spletno storitvijo se povežemo preko zahteve tipa POST, kateri dodamo besede, ki jih želimo lematizirati. Rezultat dobimo v obliki asociativne tabele v kateri je ključ beseda, vrednost pa njena lematizirano oblika. V tabelo t_beseda shranimo lematizirano besedo. V drugo tabelo t_beseda_lem_beseda pa njeno nelematizirano obliko in id lematizirane besede v prvi tabeli kot tuj ključ. Če se bo nelematizirana oblika besede še kdaj pojavila v iskalnem nizu kakega obiskovalca, je ne bo potrebno lematizirani, ampak se bo samo preko tujega ključa poiskalo njeno lematizirano obliko. Nato sledi uteževanje besed. Utežujemo lematizirane besede iskalnega niza z lematiziranimi besedami naziva in opisa dejavnosti samostojnih podjetnikov. Lematizacijo besed naziva in opisa dejavnosti samostojnega podjetnika je izvedena ob njegovem dodajanju ali spreminjanju vrednosti v teh poljih. Istočasno z lematizacijo se prešteje posamezne lematizirane besede in se jih shrani v tabelo. Se pravi, da za vsakega obrtnika vemo, katere lematizirane besede vsebuje in koliko jih je. Skupno število lematiziranih besed se prav tako shrani v tabelo. Tako če želimo uporabiti formulo TF-IDF, gledamo vrednosti v tabelah za vsakega obrtnika. Na podlagi rezultatov formule TF-IDF določimo najrelevantnejše samostojne podjetnike za dani iskalni niz in jih predstavimo obiskovalcu. Če so izpolnjeni pogoji, se obiskovalcu predlaga tudi podkategorije, ki so najbolj zastopane med zadetki, tj. med prikazanimi samostojnimi podjetniki. Če so izpolnjeni določeni pogoji, ki so predstavljeni v nadaljevanju poglavja, se obiskovalcu predlaga nov iskalni niz, ki bi ga utegnil zanimati. Vsak iskalni niz obiskovalca

se tudi shrani v dnevnik iskalnih nizov. Če obiskovalec vpiše nov iskalni niz, se v predhodnega zabeleži id novega. Glede na to povezavo se naslednjemu obiskovalcu, ki vpiše predhodni iskalni niz, predlaga novega, seveda le v primeru, če je zadoščeno določenim pogojem.

Reševanje zastavljenega problema diplomske naloge smo razdelili na več delov: lematizacijo, uporabo formule TF-IDF za uteževanje besed v besedilih, beleženje iskalnih nizov v dnevnik in njihovo vrednotenje in predlaganje kategorij. Lematizacijo smo uporabili, ker smo želeli poiskati leme oz. normalizirane oblike besed naziva in opisa dejavnosti samostojnega podjetnika ter iskalnega niza obiskovalcev. S formulo TF-IDF določimo najrelevantnejše samostojne podjetnike glede na iskalni niz obiskovalcev spletne strani. Iskalni niz predstavlja kar ključne besede za iskanje po nazivu in opisu dejavnosti samostojnega podjetnika. Vsak iskalni niz zapišemo tudi v dnevnik in vsakemu iskalnemu nizu določimo utež ter predhodni iskalni niz. Če so izpolnjeni določeni pogoji, ki so natančneje predstavljeni v nadaljevanju poglavja, se obiskovalcu predlagajo iskalni nizi prejšnjih obiskovalcev. Iskalni niz obiskovalca vrne rezultate – v določenem vrstnem redu se predstavijo samostojni podjetniki. Samostojni podjetniki so tudi razdeljeni v kategorije in podkategorije. Obiskovalcu se predlagajo podkategorije, ki so najbolj zastopane med temi zadetki. Kot je bilo napisano že v uvodu, je bila zadnja ideja reševanja problema tudi iskanje sopomenk v iskalnem nizu obiskovalca. Sopomenke smo želeli iskati v slovenskem semantičnem leksikonu, imenovanem sloWNet. Ker so sopomenski nizi semantičnega leksikona slabo pokrivali besedišče dejavnosti, smo uporabo sloWNeta izločili iz rešitve.

Lematizacija

Lematizacija se izvaja nad iskalnim nizom, ki ga vpiše obiskovalec in nad nazivom samostojnega podjetnika in njegovim opisom dejavnosti. Predstavitev samostojnega podjetnika na spletni strani najdiObrtnika.si je sestavljena iz več polj: naziv samostojnega podjetnika, naslov, regija, telefon, elektronska pošta, spletna stran, opis dejavnosti in zemljevid. Zdelo se nam je smiselno, da se ključne besede, ki jih predstavlja iskalni niz, išče samo v dveh poljih, in sicer naziv samostojnega podjetnika in opis dejavnosti, zato smo tudi lematizacijo izvedli samo v teh dveh poljih.

Lematizacijo naziva samostojnega podjetnika in njihovih opisov dejavnosti je bilo potrebno razdeliti na dva dela. V prvem delu je bilo potrebno poskrbeti za lematizacijo naziva in opisa dejavnosti že obstoječih samostojnih podjetnikov, ki se nahajajo v bazi, saj lematizacija nad njimi še ni bila izvedena. Drugi korak je zajemal lematizacijo naziva in opisa na novo prijavljenih samostojnih podjetnikov. To smo storili tako, da se ob dodajanju novega samostojnega podjetnika v bazo istočasno izvede tudi lematizacija njegovega naziva in opisa dejavnosti. Lematizirane besede se nato shranijo v tabele baze. Za že registrirane samostojne podjetnike, katerih polji naziv in opis dejavnosti še nista bili lematizirani, smo spodaj opisani postopek izvedli naknadno.

Celotni postopek lematizacije je sestavljen iz več korakov:

1. Gradnja niza besed za lematizacijo

Preden se preko vtičnice povežemo s storitvijo na gostitelju, kjer se nahaja lematizator, je potrebno izvesti gradnjo niza besed, ki jih želimo lematizirati. To pomeni, da iz tabele v bazi v niz skupaj zapišemo naziv samostojnega podjetnika in opis njegove dejavnosti. Pri tem odstranimo vsa ločila, številke, pogoste besede brez informacijske vrednosti npr. s.p. in ostale znake, ki se jih ne da lematizirati. Pri lematizaciji iskalnega niza, ki ga vpiše obiskovalec, ta korak ni potreben, saj niz za lematizacijo predstavlja kar celoten iskalni niz obiskovalca.

Gradnja niza besed za lematizacijo poteka na sledeč način (velja samo za lematizacijo naziva in opisa dejavnosti samostojnih podjetnikov): naprej se odstrani oznako html za odebeljen tekst in in besede s.p. in d.o.o ne glede na velike in male črke. V naslednjem čiščenju oz. gradnji niza se znebimo vseh ločil in številk. Tako odstranimo naslednja ločila in znake: ., ;, !, ?, :, <, >, „,), (, +, -, «, », /, \\, …, _, |. Na enak način odstranimo številke od 0 do 9.

2. Povezava preko vtičnice na gostitelja

Za lematizacijo smo uporabili spletni servis, ki se nahaja na naslovu http://nl2.ijs.si/analyze/ [8]. Strežnik sprejme zahtevo tipa POST v naslednji obliki: argument[niz] = TEXT=$zgrajeni_niz&GET=show.

TEXT in GET sta imeni html elementov znotraj html elementa form na spletni strani [8]. Zgrajeni_niz je niz iz prve točke ali iskalni niz obiskovalca. Za povezavo preko vtičnice na gostitelja, kjer se nahaja lematizator, smo uporabili funkcijo, ki smo jo našli na spletnem naslovu [7]. Argumenta funkcije sta naslov gostitelja in niz za lematizacijo, ki smo ga zgradili v prvem koraku. V primeru, da gre za iskalni niz obiskovalca, je argument namesto izgrajenega niza kar iskalni niz.

3. Shranjevanje lematiziranih besed v tabele

Povezovanje na gostitelja in lematizacija zgrajenega niza sta časovno zahtevni operaciji. Zato smo se odločili, da se lematizacija naziva in opis dejavnosti samostojnih podjetnikov izvede samo enkrat. Rezultate lematizacije smo shranili v tabelo. Tabela t_beseda vsebuje vse lematizirane besede. V drugi tabeli, imenovani t_obrtniki_besede, se nahaja id obrtnika, id lematizirane besede in kolikokrat se ta beseda pojavi v nazivu samostojnega podjetnika in opisu njegove dejavnosti. Število pojavitev v nazivu in opisu dejavnosti smo potrebovali za formulo TF-IDF.

Podobno smo storili tudi v primeru vpisa iskalnega niza. Najprej smo preverili, ali so besede že bile kdaj lematizirane. Vse v iskalnem nizu že kdaj koli prej lematizirane besede so namreč zapisane v bazi v tabeli t_beseda_lem_beseda. Tabela vsebuje besedo in id njene lematizirane različice. Če besede še ni bilo v tabeli smo jo lematizirali glede na prej predstavljen postopek in jo shranili v tabelo t_beseda_lem_beseda. S shranjevanjem lematiziranih besed v tabele smo močno omejili zahteve po dostopu do lematizatorja na spletni strani [8].

Ocenjevanje relevantnih zadetkov

Formula TF-IDF se uporablja za iskanje in zbiranje informacij (information retrieval) v dokumentih. V našem primeru smo to formulo uporabili za iskanje ključnih besed, ki jih predstavlja iskalni niz obiskovalca spletne strani najdiObrtnika.si. Ključne besede smo iskali v nazivu in opisu dejavnosti samostojnih podjetnikov.

Glede na postopek, ki smo ga opisali v uvodu, je bilo potrebno izluščiti nekatere parametre. Ker se naziv in opis dejavnosti samostojnega podjetnika redkokdaj spreminjata, je dovolj, da se indeksiranje parametrov izvede samo ob primerih, ko se dodaja ali briše samostojnega podjetnika ali ob spreminjanju naziva ali opisa dejavnosti posameznega samostojnega podjetnika.

Parametri, ki so shranjeni v bazi:

1. število vseh samostojnih podjetnikov – |D|;
2. število vseh besed pri posameznem samostojnem podjetniku - $f'_{ai,d}$;
3. število kolikokrat se beseda pojavi v nazivu in opisu dejavnosti samostojnega podjetnika – $f'_{wi,d}$;
4. pri koliko samostojnih podjetnikih se posamezna beseda pojavi – $f_{wi,D}$.

Število vseh samostojnih podjetnikov se shrani v tabelo t_num_all, ki ima samo eno polje, imenovano num_all s samo enim vnosom – številom vseh samostojnih podjetnikov. Število vseh besed posameznega samostojnega podjetnika je shranjeno v tabeli t_obrtniki_besede_all. Tabela vsebuje polji z id-jem samostojnega podjetnika in številom vseh besed v nazivu in opisu dejavnosti samostojnega podjetnika. V točki 3 se v tabelo t_obrtniki_besede shranijo id samostojnega podjetnika, id lematizirane besede iz tabele t_beseda in število teh besed v nazivu in opisu dejavnosti samostojnega podjetnika. V točki 4 se v tabelo t_beseda_doc shrani id lematizirane besed iz tabele t_beseda in kolikokrat se ta beseda pojavi v nazivu in opisu dejavnosti samostojnega podjetnika.

Funkcija TF-IDF kot argument sprejme lematizirani iskalni niz, ki ga vpiše obiskovalec spletne strani. Besede iskalnega niza predstavljajo ključne besede, ki se iščejo med samostojnimi podjetniki (v poljih: naziv in opis dejavnosti), ki so tudi lematizirana. TF-IDF vrednost izračunamo za vsakega samostojnega podjetnika posebej tako, da seštejemo TF-IDF vrednosti posameznih besed iskalnega niza po **formuli [2]**. w_d je TF-IDF vrednost samostojnega podjetnika d in w_{id} je TF-IDF vrednost za besedo i iskalnega niza. Da bi lahko izračunali TF-IDF vrednost, se za vsakega samostojnega podjetnika in vsako lematizirano besedo iskalnega niza iz baze preberejo $f'_{ai,d}$, $f'_{wi,d}$, $f_{wi,D}$. Logično je, da število vseh samostojnih podjetnikov iz baze preberemo samo enkrat. TF-IDF vrednost izračunamo s **formulo [1]**, pri tem je $f_{wi,d} = \frac{f'_{wi,d}}{f'_{ai,d}}$.

Ker smo med eksperimentiranjem dobili kar nekaj primerov, da je samostojni podjetnik, ki v svojem nazivu in opisu dejavnosti ni imel vseh besed iskalnega niza, imel večjo TF-IDF vrednost kot samostojni podjetnik, ki je imel v svojem nazivu in opisu dejavnosti vse besede iskalnega niza, smo TF-IDF vrednost slednjega pomnožili s 4. Tako imajo slednji samostojni podjetniki večjo TF-IDF vrednost, saj se nam zdijo nazivi in opisi dejavnosti pri samostojnih podjetnikih, ki vsebujejo vse besede iskalnega niza relevantnejši, kot pri tisti samostojnih podjetnikih, ki vsebujejo samo nekatere.

Funkcija na koncu vrne padajoče urejeno asociativno tabelo, kjer so ključi tabele id-ji samostojnih podjetnikov in vrednosti tabele TF-IDF vrednosti za samostojnega podjetnika.

Dnevnik iskalnih nizov in njihovo predlaganje obiskovalcu

Eden izmed korakov diplomske naloge zaobjema tudi beleženje iskalnih nizov obiskovalcev spletne strani najdiObrtnika.si v dnevnik, ki se shranjuje v tabelo baze.

V tabeli se zabeleži:

a. lematiziran iskalni niz;

b. čas vpisa;

c. id seje obiskovalca;

d. id samostojnih podjetnikov, ki so bili prikazani;

e. id samostojnih podjetnikov, katerih povezave je obiskovalec kliknil;

f. id novega iskalnega niza;

g. utež;

h. število prikazov predlaganega iskalnega niza;

i. število klikov predlaganega iskalnega niza.

V nadaljevanju bomo podrobneje predstavili nekatera prej našteta polja tabele. Polja tabele so za potrebe diplomske naloge kar poimenovana z zaporednimi črkami, ki se nahajajo pred njimi. Vsakemu obiskovalcu spletne strani se dodeli unikatni id PHP seje. Z AJAX tehnologijo smo realizirali, da se v tabelo zabeleži tudi vsak klik na povezave, ki jih nudi opis posameznega samostojnega podjetnika. Povezave v opisu so sledeče: elektronska pošta, spletna stran in zemljevid na spletni strani najdi.si, ki pokaže lokacijo samostojnega podjetnika. Ko obiskovalec klikne katero od prej omenjenih povezav, se to zabeleži v tabeli. Ker smo želeli opazovati tudi, kako obiskovalci izoblikujejo iskalne nize, če le-ti niso dali želenih rezultatov, se pod določenimi pogoji v polje (f) zapiše id iskalnega niza, ki je bil vpisan za predhodnim nizom istega obiskovalca. Pogoji, ki morajo biti izpolnjeni, da se id

novega niza zapiše, so naslednji: čas, ki preteče od vpisa predhodnega niza do vpisa novega niza istega obiskovalca, mora biti večji od 2s in manjši od 90s. Spodnjo mejo smo postavili, da smo preprečili skriptno vnašanje niza iskalne vrstice. Zgornja meja je bila namenjena obiskovalcem, in sicer smo predvideli, da bo obiskovalec v največ 90s vpisal nov iskalni niz, ki se bo navezoval na predhodnega. V nasprotnem primeru smo smatrali, da gre za od predhodnega iskalnega niza neodvisen iskalni niz.

id_log	query	time	session	id_show_o	id_clicked_o	id_new_query	weight	new_query_shown	new_query_clicked
1261	masaža	2010-03-23 15:46:08	1ec8f85f0a5f05eafb5fc8de97e85b	138, 139, 143, 145, 147, 148, 149, 150, 151, 154, ...		43	0.1	0	0
1262	masaža hrbet	2010-03-23 15:49:08	1ec8f85f0a5f05eafb5fc8de97e85b	138, 139, 143, 145, 147, 148, 149, 150, 151, 154, ...		0	0.5	4	0
1263	masaža	2010-03-23 15:46:37	42a7d278b2bb80a83bd623751fd2486	138, 139, 143, 145, 147, 148, 149, 150, 151, 154, ...		0	0.5	0	0
1264	masaža	2010-03-23 15:48:34	bdced5403b2c551d553a0f800dc44b	138, 139, 143, 145, 147, 148, 149, 150, 151, 154, ...		44	0.1	0	0
1265	masaža ljubljana	2010-03-23 15:49:08	bdced5403b2c551d553a0f800dc44b	7, 14, 35, 138, 139, 143, 145, 147, 148, 149, 150, ...		0	0.5	2	0
1266	masaža	2010-03-23 15:48:58	bdced5403b2c551d553a0f800dc44b	138, 139, 143, 145, 147, 148, 149, 150, 151, 154, ...		45	0.1	0	0
1267	masaža stopati	2010-03-23 15:49:08	bdced5403b2c551d553a0f800dc44b	138, 139, 143, 145, 147, 148, 149, 150, 151, 154, ...		0	0.5	1	0
1268	masaža	2010-03-23 15:49:08	bdced5403b2c551d553a0f800dc44b	138, 139, 143, 145, 147, 148, 149, 150, 151, 154, ...		0	0.5	0	0
1269	a rent car	2010-03-23 15:55:50	bdced5403b2c551d553a0f800dc44b	4, 111, 169, 171, 190, 227,	4	0	0.9	0	0

Slika 2: Izsek iz dnevnika iskalnih nizov.

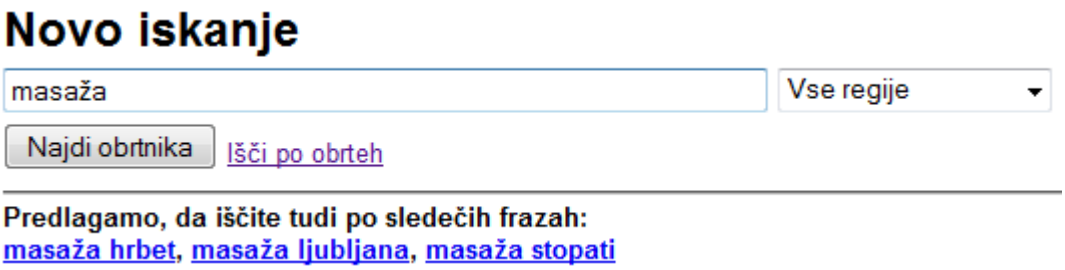
Prej opisana prehodni in novi iskalni niz nastopata v paru. Povezana sta preko id novega iskalnega niza, ki je zapisan v polju (f) predhodnega niza. Povezave med predhodnim in novim iskalnim nizom nismo ustvarili, če je par že obstajal v tabeli. Polje (f) predhodnega niza je v tem primeru ostalo prazno. Na podlagi teh id-jev v polju (f) smo obiskovalcu predlagali nov iskalni niz v obliki povezave. S klikom na to povezavo se je predlagan iskalni niz vpisal v iskalno vrstico in avtomatsko se je sprožilo novo iskanje. Več podrobnosti o predlaganih iskalnih nizih bomo predstavili kasneje. Vsakemu vpisanemu iskalnemu nizu obiskovalca smo dodelili utež, in sicer so uteži odvisne od dejanj obiskovalca. Interval za uteži se giblje med 0,1 in 0,9. Privzeta vrednost uteži je 0,5. Ta vrednost ostane nespremenjena, če obiskovalec samo vpiše iskalni niz in potem zapusti stran brez vpisa novega iskalnega niza ali ne da bi kliknil na katero od povezav (elektronska pošta, zemljevid, spletna stran) samostojnih podjetnikov. O tem obiskovalcu nismo mogli zbrati nobene informacije, zato je vrednost uteži nespremenjena in na polovici intervala. Drug primer je, da obiskovalec vpiše iskalni niz in v intervalu od 2s do 90s nov iskalni niz. V tem primeru smo predpostavili, da obiskovalec ni bil zadovoljen z vrnjenimi rezultati. Predhodni iskalni niz je v tem primeru dobil vrednost 0,1, novi pa vrednost 0,5. Predhodni iskalni niz je z utežjo 0,1 dobil oznako kot neustrezen. Naslednji primer je, da obiskovalec vpiše iskalni niz, klikne na katero od povezav samostojnih podjetnikov in šele nato vpiše nov iskalni niz. Predhodni iskalni niz v tem primeru dobi vrednost 0,65, novi pa 0,5. V tem primeru sklepamo, da je obiskovalec dobil nekaj rezultatov – predstavljenih samostojnih podjetnikov. Samostojni podjetniki so ga tudi zanimali, vendar ni našel tistega, kar je iskal. Zato je vpisal nov iskalni niz. Zadnji primer je, da obiskovalec vpiše iskalni niz in nato klikne na povezavo katerega od prikazanih samostojnih podjetnikov. V tem primeru smo predpostavili, da je obiskovalec s svojim iskalnim nizom našel ustrezne samostojne podjetnike. Utež tega niza smo torej postavili na 0,9. Utež vedno shranimo v polje (g).

V polje (h) shranimo, kolikokrat je bil predlagani iskalni niz prikazan obiskovalcem. V polje (i) shranimo, kolikokrat je bil predlagan iskalni niz kliknjen.

Del dnevnika iskalnih nizov je prikazan na sliki 2. Zapišimo nekaj opažanj. Prvi zapis z id-jem 1261 ima podan id_new_query, kar pomeni, da ima ta niz povezavo za novi predlagan niz. Ker je new_query_shown manjši od 20, bo novi iskalni niz prikazan obiskovalcu, dokler new_query_shown ne doseže vrednosti 20. Prikaz je nato odvisen od tega, ali niz z id-jem 1261 zadošča pogojem. Utež ima vrednost 0,1 kar pomeni, da je bil v intervalu med 2s in 90s vpisan nov iskalni niz, ne da bi obiskovalec kliknil kakšno povezavo na spletno stran ali elektronsko pošto. Iskalni nizi z id-ji 1262, 1263, 1265, 1267, 1268 imajo vrednost uteži 0,5. To pomeni, da so obiskovalci vpisali iskalni niz in nato v 90s niso vpisali novega, ali pa so zapustili stran brez vpisa novega iskalnega niza. Pri zadnjem iskalnem nizu z id-jem 1269 vidimo, da je obiskovalec kliknil na eno od ponujenih povezav (povezavo na spletno stran, elektronsko pošto ali na spletno stran z zemljevidom). V id_clicked_o vidimo, da je obiskovalec izbral samostojnega podjetnika z id-jem 4.

Sedaj ugotavljamo, v katerih primerih se obiskovalcu predlagajo novi iskalni nizi. Kot vemo že iz prejšnjega odstavka, sta dva niza, ki ju nek obiskovalec vpiše enega za drugim,

povezana preko polja (f), ki vsebuje id novega (predlaganega) niza. Osnovni pogoj, da je obiskovalcu predlagan iskalni niz, je da obiskovalec vpiše iskalni niz, ki vsebuje id novega iskalnega niza. Osvetlimo zadnji stavek s primerom: Obiskovalec A vpiše v iskalno vrstico *najem kombija* z id število 1 in nato vpiše nov iskalni niz *najem kombi vozila* z id številko 2. V tabelo se pod zaporedno številko id-ja 1 shrani v polje (f) 2, ki je id novega iskalnega niza. Če bo v prihodnosti obiskovalec B vpisal iskalni niz *najem kombija*, se mu bo predlagalo pod določenimi pogoji, naj pogleda oz. klikne tudi na iskalni niz *najem kombi vozila*. Na sliki 5 vidimo, kako zgoraj opisan primer izgleda na spletni strani najdiObrtnika.si.



Slika 3: Obiskovalcu se glede na njegov iskalni niz pri določenih pogojih predlaga sorodne iskalne nize.

Oglejmo si sedaj še ostale pogoje, da se obiskovalcu predlaga nov iskalni niz. Prvi pogoj je, da polje (d), ki beleži id-je samostojnih podjetnikov, ni prazno. Ta omejitev je popolnoma logična, saj nima smisla predlagati nov iskalni niz, če ta niz ne predlaga nobenega samostojnega podjetnika. V primeru, da polje z id-ji samostojnih podjetnikov ni prazno, se najprej pogleda, kolikokrat je bil iskalni niz prikazan. Če še ni bil prikazan 20-krat, se ta iskalni niz predlaga obiskovalcu. Če pa je že bil predlagan 20-krat, se izračuna razmerje med prikazi in kliki predlaganega niza $\frac{\text{kliki predlagane niza}}{\text{prikazi predlaganega niza}}$. Nato to vrednost pomnožimo z utežjo, ki se nahaja v polju (g) predlaganega iskalnega niza. Če je produkt $\frac{\text{kliki predlagane niza}}{\text{prikazi predlaganega niza}} * utež > 0{,}13$ se obiskovalcu predlaga ta iskalni niz. Vrednost 0,13 smo določili takole: želeli smo, da je razmerje med prikazi in kliki predlaganega iskalnega niza vsaj 0,2. Po drugi strani se nam je zdelo smiselno, da je minimalna utež predlaganega niza vsaj 0,65. Produkt teh dveh števil je ravno 0,13. Seveda je možno, da se za isti iskalni niz predlaga več iskalnih nizov. Torej vsakega, katerega produkt uteži in razmerja med prikazi in kliki iskalnega niza presega vrednost 0,13.

Predlaganje najbolj zastopanih podkategorij med zadetki

Samostojni podjetniki so razdeljeni v katergorije. Vsaka kategorija ima svoje podkategorije. Kategorije so naslednje: (tradicionalne) Domače obrti, Avtomobilizem, Dom in vrt, Industrijske dejavnosti, Izbraževanje in inštrukcije, Lepotne storitve in osebna nega, Ostalo, Računalništvo in poslovne storitve, Transport in najem vozil. Zaradi množičnosti podkategorij ne bomo naštevali. Klasifikacijo nekaterih kategorij in njihovih podaktegorij si

lahko ogledate na sliki 4. Samostojni podjetniki so v kategorije in v podkategorije klasificirani ročno ob dodajanju samostojnega podjetnika v bazo.

Slika 4: Klasifikacija nekaterih kategorij in podkategorij.

Vsak samostojni podjetnik pripada eni kategoriji in največ dvema podkategorijama. V diplomski nalogi smo se odločili, da bomo obiskovalcu glede na njegov iskalni niz predlagali tiste podkategorije, ki so med rezultati iskanja, se pravi med predstavljenimi samostojnimi podjetniki, najbolj zastopane. Za predlaganje podkategorij smo uporabili sledečo formulo:

$$\chi^2 = \frac{Z_k - N_k}{N_k} > 34{,}00$$

Formula 3: Formula za predlaganje podkategorij.

$N_k = \frac{Z}{N} * N_{1k}$ in N je število vseh samostojnih podjetnikov, Z je število vseh samostojnih podjetnikov med rezultati glede na iskalni niz, Z_k je število samostojnih podjetnikov med rezultati iskalnega niza glede na določeno podkategorijo k in N_{1k} število vseh samostojnih podjetnikov v bazi posamezne podkategorije. χ^2 se nanaša na podkategorije predstavljenih samostojnih podjetnikov glede na iskalni niz obiskovalca. χ^2 leži znotraj sledečega intervala: $0 < \chi^2 < \infty$. Podkategorija se obiskovalcu predlaga, če je $\chi^2 > 34{,}00$.

Mejno vrednost 34,00 smo določili empirično, tako da smo vnesli nekatere iskalne nize in pogledali, kakšna je vrednost χ^2 za posamezno podkategorijo. Da bi določili mejno vrednost, smo izbrali naslednje iskalne nize*: najem vozil* (tabela [2]), *vodovodne inštalacije* (tabela [3]) in *polaganje parketa* (tabela [4]). Nato smo za vsako podkategorijo pogledali, kakšna je njena vrednost χ^2. Vrednosti χ^2 za izbrane iskalne nize so zbrane v tabelah [2], [3] in [4]. V tabelah so zapisane samo tiste podkategorije, katerih vrednost χ^2 je večja od 0. Iz tabel je razvidno, da ima večina podkategorij, ki ne ustrezajo iskalnemu nizu, χ^2 manjši od 10. Nekaj podkategorij je imelo vrednost χ^2 do 34,00. Samo ena ali dve podkategoriji pa imata χ^2

vrednost večjo od 34,00. V tabeli [2] in [3] najdemo tudi 2 neskladji. V tabeli [2] kategorija *Rent a car* ni uvrščena med predlagane podkategorije obiskovalcu. To pripisujemo dejstvu, da v to podkategorijo ni uvrščeno zadostno število samostojnih podjetnikov. V tabeli [3] gre za manjše neskladje, ki ga pripisujemo istemu razlogu kot zgoraj. In sicer podkategorija *Vodovodne inštalacije* ni na prvem mestu, dasiravno bi to glede na iskalni niz *vodovodne inštalacije* pričakovali. Ne glede na to ta podkategorija še vedno presega prag za prikaz, kar je dobro.

podkategorija	χ^2
Najem kombi vozil	61,94
Najem avtodoma	38,82
Avtoservis	33,44
Vulkanizerstvo, platišča in pnevmatike	19,41
Rent a car	19,41
Taksi	15,48
Avtobusni prevozi	6,82
Odpad rabljenih vozil	6,82
Nepremičnine	6,82
Dostava	3,98
Avtoprevozniki	3,98
Avtovleka	1.77
Čiščenje	0.022

Tabela 2: Vrednosti χ^2 po podkategorijah za iskalni niz *najem vozila*.

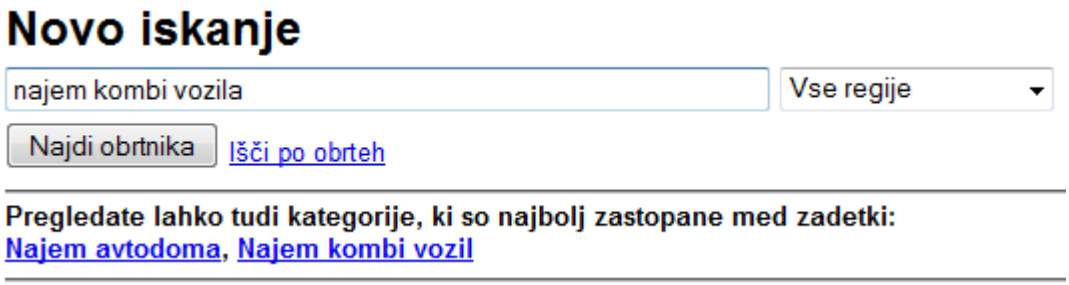
podkategorija	χ^2
Gretje, ventilacija, klimatizacija	126.71
Vodovodne inštalacije	86.69
Elektromehanika	5.31
Ostalo	2.99
Strehe in krovstvo	1.86
Zaključna gradbena dela	0.77
Slikopleskarska dela	0.56

Tabela 3: Vrednosti χ^2 po podkategorijah za iskalni niz *vodovodne inštalacije*.

podkategorija	χ^2
Parketarstvo in talne obloge	93.03
Zaključna gradbena dela	26.14
Slikopleskarska dela	6.01
Urejanje vrta in okolice	4.26
Restavratorji	1.68
Čiščenje	0.72
Mizarstvo	0.23

Tabela 4: Vrednosti χ^2 po podkategorijah za iskalni niz *polaganje parketa*.

Slika 8 prikazuje, kako se obiskovalcu predlagajo podkategorije. S klikom na povezavo podkategorije se nato predstavijo oz. prikažejo vsi samostojni podjetniki kliknjene podkategorije.



Slika 5: Primer predlaganja podkategorij za iskalni niz *najem kombi vozila*.

Ker se vrednosti spremenljivk enačb $\chi^2 = \frac{Zk - Nk}{Nk}$ in $N_k = \frac{Z}{N} * N_{1k}$ ne spreminjajo pogosto jih lahko shranimo v tabeli baze. Spremenljivka N je pravzaprav spremenljivka |D| iz poglavja 3. Tako pri tej enačbi uporabljamo dejansko spremenljivko |D|, ki je shranjena v tabeli t_num_all. Za spremenljivko N_{1k} pa smo ustvarili novo tabelo, imenovano t_obrtniki_podobrt, v katero smo shranili v eno polje id podkategorije, v drugo pa število samostojnih podjetnikov, ki pripada tej podkategoriji. Polja teh dveh tabel se posodobijo v primeru, ko se doda novega samostojnega podjetnika, v primeru, da se samostojnega podjetnika odstrani iz baze, ali v primeru, ko prestavimo samostojnega podjetnika iz ene podkategorije v drugo.

4. Opis okolja in aplikacije

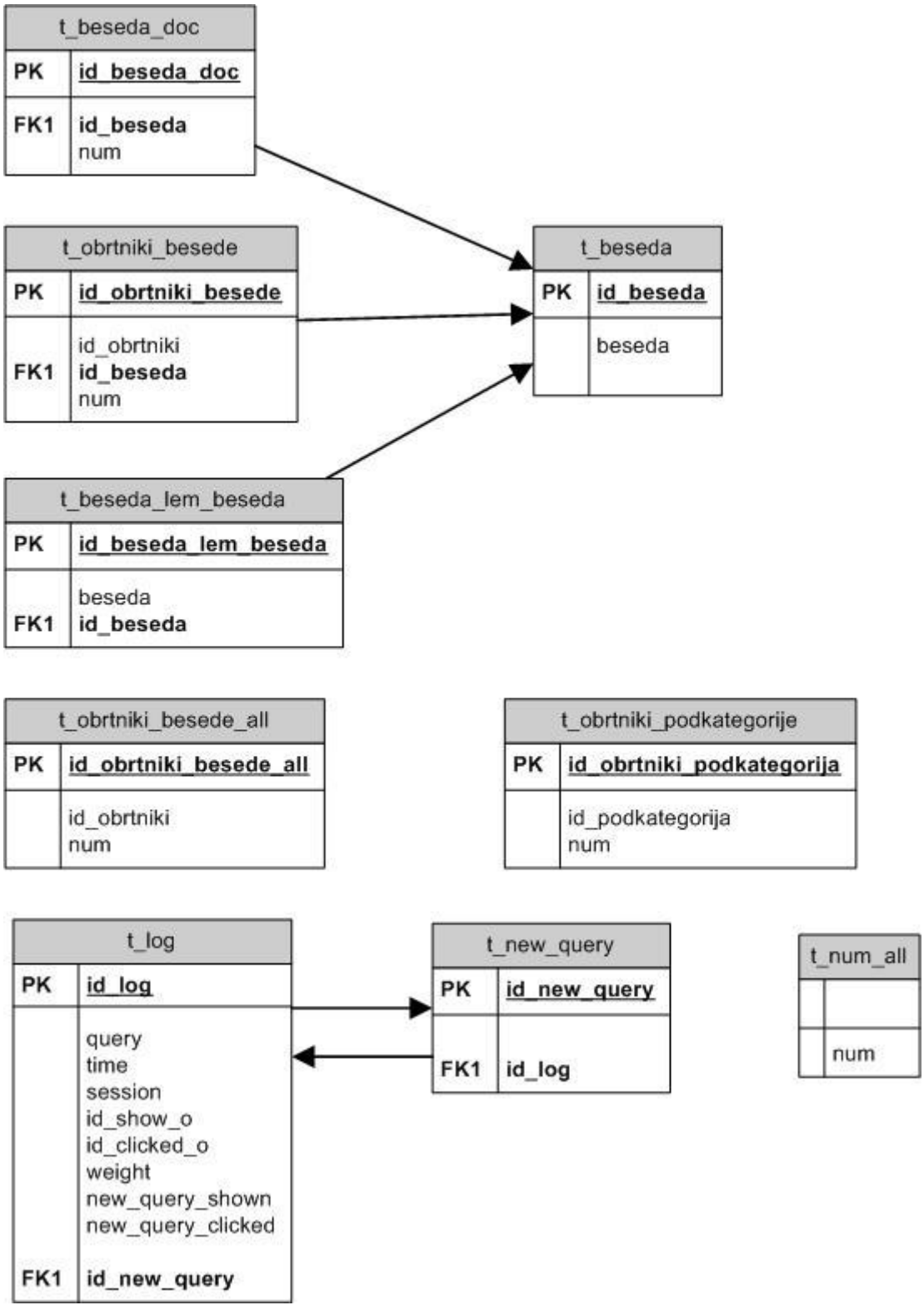
V tem poglavju smo navedli razvojna okolja in njihove verzije, s pomočjo katerih smo razvili aplikacijo. V nadaljevanju poglavja smo navedli nekaj statistike spletne strani najdiObrtnika.si. Nato smo predstavili tabele, ki so relevante za diplomsko nalogo, in relacije med njimi. Na koncu poglavja smo prestavili še funkcijo suggestion_query, ki obiskovalcem predlaga nov predlagani iskalni niz.

Celotna aplikacije je razvita v PHP programskem jeziku, in sicer verzija 5.1.2. Za ustvarjanje baze smo uporabili MySQL verzijo 5.0.22. Za direktni dostop do baze najdiobrtnika.si smo uporabili phpMyAdmin verzijo 2.8.2.4. Za razvoj aplikacije smo uporabljali Microsoftov izdelek Microsoft Expression Web 2.0. V diplomski nalogi smo želeli uporabiti tudi slovenski semantični leksikon, imenovan sloWNet, avtorice Darje Fišer. Verzija, ki nam jo je avtorica poslala in s katero smo si želeli pomagati, je bila 2.1. Za lematizacijo smo uporabili spletno storitev, imenovano JOS ToTaLe text analyser, na spletni strani http://nl.ijs.si/jos/analyse/.

Spletna stran je v času pisanja (marec, 2010) aktivna skoraj eno leto. Prvi samostojni podjetnik se je prijavil 11. aprila 2009. V marcu leta 2010 je bilo na strani oz. v bazi prijavljenih 250 samostojnih podjetnikov. Največ samostojnih podjetnikov je prijavljenih v podkategoriji *Zaključna gradbena dela* in sicer, 25, sledi podkategorija *Mizarstvo* z 22 samostojnimi podjetniki. Največ samostojnih podjetnikov prihaja iz regije *Ljubljana z okolico*, in sicer 90. Glede na spletno aplikacijo za merjenje obiska spletnih strani Google Analytics je bilo v celotnem obdobju, od kar deluje spletna stran v času pisanja, št. obiskov spletne strani najdiObrtnika.si 4680, kar znese malo manj kot 13 obiskov na dan. Število obiskovalcev je bilo 2837.

Tabele, ki so relevante za diplomsko nalogo, so naslednje: t_beseda_doc, t_obrtniki_besede, t_beseda_lem_beseda, t_obrtniki_besede_all, t_beseda, t_obrtniki_podkategorije, t_log, t_new_query, t_num_all. Tabela t_beseda vsebuje vse lematizirane besede, ki se nahajajo v nazivih in opisih dejavnostih samostojnih podjetnikov. Prav tako vsebuje vse lematizirane besede, ki so jih obiskovalci vpisali v obliki iskalnih nizov. Tabela t_beseda_doc vsebuje podatek, pri koliko različnih samostojnih podjetnikih se nahaja določena beseda. Id_beseda predstavlja tuj ključ, saj je to id lematizirane besede, ki se nahaja v tabeli t_beseda. Tabeli sta preko tujega ključa v relaciji ena proti ena. Tabela t_obrtniki_besede vsebuje za vsakega samostojnega podjetnika posebej, katere in koliko posameznih lematiziranih besed vsebujeta njegovi polji naziva in opisa dejavnosti. Tudi tukaj id_beseda nastopa kot tuj ključ in povezuje to tabelo s tabelo t_beseda. Povezava med njima

je mnogo proti ena. Zadnji dve omenjeni tabeli uporabljamo pri formuli TF-IDF. Sledi tabela t_beseda_lem_beseda, ki vsebuje nelematizirano različico besede iskalnega niza in njeno lematizirano različico. Lematizirana različica je podana v obliki tujega ključa id_beseda, ki to tabelo povezuje s tabelo t_beseda. V to tabelo se shranjujejo besede iskalnih nizov, da jih v prihodnosti, če še kateri obiskovalec vpiše enako besedo, ni potrebno ponovno lematizirati. V tabeli t_obrtniki_besede_all je shranjeno, koliko različnih lematiziranih besed vsebuje naziv in opis dejavnosti posameznega samostojnega podjetnika. Tabelo rabimo pri računanju TF-IDF. V tabeli t_obrtniki_podkategorije je zabeleženo, koliko samostojnih podjetnikov pripada določeni podkategoriji. To tabelo rabimo pri predlaganju novih podkategorij obiskovalcem. Tabela t_num_all vsebuje število vseh obrtnikov. Potrebujemo jo tako za računanje formule TF-IDF, kot za predlaganje novih podkategorij obiskovalcem. Zadnji dve tabeli sta t_log in t_new_query, ki sta povezani v obe smeri v razmerju ena proti ena s tujim ključen id_new_query v tabeli t_log in s tujim ključem id_log v tabeli t_new_query. V tabeli t_log beležimo vse iskalne nize, pri katerih je čas med dvema sosednjima nizoma večji od 2s. V tej tabeli hranimo iskalni niz obiskovalca, čas vpisa iskalnega niza, id PHP seje obiskovalca, id-je prikazanih samostojnih podjetnikov, id-je samostojnih podjetnikov, katerih spletne povezave je obiskovalec kliknil, utež iskalnega niza, kolikokrat je bil novi predlagani niz prikazan in kolikokrat je bil novi predlagan niz uporabljen oz. kliknjen. Zadnje polje je id_new_query. Tabela t_new_query vsebuje id-je iskalnih nizov tabele t_log. Ti dve tabeli potrebujemo za predlaganje novih iskalnih nizov obiskovalcem. Tabele in njihove relacije so prikazane na sliki 6.



Slika 6: Za diplomsko nalogo relevantne tabele v bazi najdiObrtnika.si.

```php
<?php
function suggestion_query($vrstica, $set_hr)
{
    global  $default_weight, $new_query_weight, $click_new_query_weight, $clicks_weight, $new_query_shown, $shown_border;
    $i = TRUE;
    $j = 0;
    $result2 = mysql_query("SELECT id_log FROM t_new_query WHERE id_new_query IN (SELECT id_new_query FROM t_log WHERE query='$vrstica' and id_new_query != 0)");

    while($row2 = mysql_fetch_array($result2))
    {
        $query_id = $row2['query_id'];
        $result = mysql_query( "SELECT query, weight, id_log, new_query_shown, new_query_clicked, id_show_o FROM t_log WHERE id_log = '$query_id'");

        while($row = mysql_fetch_array($result))
        {
            if($row['new_query_shown'] < $new_query_shown)
                $razmerje = $click_new_query_weight;
            else
                $razmerje = $row['new_query_clicked']/$row['new_query_shown'];

            if(($row['weight']*$razmerje) > $shown_border and !empty($row['id_show_o']))
            {
                $j++;
                if($i)
                {
                    $text .="<div style='font-weight:bold; font-size:small; margin-top:8px; margin-bottom:8px;'>Predlagamo, da izberete tudi po sledečih frazah:<br />";
                    $i = FALSE;
                }

                $text .= "<a href='search.php?vrstica=". $row['query'] ."&pokrajina=Vse+regije' onclick='update_clicks(" . $row['id_log'] . ")'>".$row['query']."</a>" ." ";

                $prikazov = $row['new_query_shown'] + 1;
                $id_log = $row['id_log'];
                mysql_query("UPDATE t_log SET new_query_shown = '$prikazov' WHERE id_log = '$id_log'");
                if($j == 5)
                {
                    $text .='<br />';
                    $j = 0;
                }
            }
        }
    }
    if(!$i)
    {
        $text = substr_replace($text,"",-2);
        if($set_hr)
            $text .="<hr /></div>";
        else
            $text .="</div>";
        return $text;
    }
}
```

Slika 7: Koda funkcije suggestion_query.

Slika 7 prikazuje PHP funkcijo z imenom suggestion_query. Ta funkcija je namenjena predlaganju novih iskalnih nizov obiskovalcem glede na njihov predhodni niz. Funkcija kot argument sprejme predhodni iskalni niz obiskovala – $vrstica in spremenljivko tipa logična vrednost (boolean) – $set_hr. Funkcija vrne PHP spremenljivko v obliki niza ($text), ki vsebuje vse potrebne HTML oznake za prikaz predlaganih novih nizov in same nove

predlagane iskalne nize v obliki spletne povezave. Če je spremenljivka $set_hr enaka vrednosti resnično (true), se na koncu niza, imenovanega $text, doda vodoravno črto. To storimo tako, da dodamo oznako HTML </hr>. V 4. vrstici imamo podane globalne spremenljivke:

a) $default_weight (vrednost: 0,5);

b) $new_query_weight (vrednost: 0,1);

c) $click_new_query_weight (vrednost: 0,65);

d) $clicks_weight (vrednost: 0,9);

e) $show_border (vrednost: 0,13).

Vrednosti pod zaporednimi črkami a, b, c in d predstavljajo uteži, ki se pripišejo posameznemu iskalnemu nizu v dnevniku iskalnih nizov. Utež iskalnega niza ima lahko samo eno izmed zgoraj naštetih vrednosti. Utež, ki se določi posameznemu iskalnemu nizu, je odvisna od akcij, ki jih izvede obiskovalec. Vrednost $show_border predstavlja mejo, pri kateri se obiskovalcu predlaga nov iskalni niz, ko število prikazov novega iskalnega niza preseže število 20. Dokler je število prikazov manjše od 20, se nov iskalni niz predlaga vsakokrat. Kako smo določili vrednost 0,13, je razloženo v tretjem poglavju.

Spremenljivka $i je tipa logična vrednost (boolean) in se uporabi za tvorjenje elementa HTML skupaj z uvodnim tekstom (*Predlagamo, da iščete tudi po sledečih frazah:*) v vrstici 26. Spremenljivka $j je tipa celo število (integer) in jo uporabimo, da se v eno vrstico izpiše največ pet novih predlaganih iskalnih nizov. Če je nizov več kot pet, se te zapiše v novo vrstico. Ta del kode se nahaja v vrsticah od 35 do 39.

V vrstici 7 naredimo prvo poizvedbo SQL, ki jo lahko razdelimo na dva dela. V prvem delu izberemo id-je tabele t_log, ki vsebujejo tuje ključe za tabelo t_new_query, in sicer mora biti iskalni niz obiskovalca identičen iskalnemu nizu, ki je že zapisan v tabeli t_log. Poleg tega mora biti tuj ključ različen od 0. V drugem delu poizvedbe SQL izberemo id-je (id_log) v tabeli t_new_query, ki vsebujejo id-je iskalnih nizov v tabeli t_log. Sedaj imamo podane id-je iskalnih nizov in lahko v novi poizvedbi SQL izberemo tista polja, ki imajo id-je enake prvi poizvedbi SQL. V vrstici 12 s poizvedbo SQL dostopamo do naslednjih polj: query, weight, id_log, new_query_shown, new_query_clicked, id_show_o. V vrstici 16 preverimo, ali smo novi iskalni niz že prikazali dvajsetkrat. Če še ni bil prikazan dvajsetkrat, dobi spremenljivka $razmerje vrednost 0,65 oz. $click_new_query_weight, kar pomeni, da bo ta novi iskalni niz gotovo predlagan obiskovalcu. V nasprotnem primeru izračunamo razmerje med kliknjenimi predlaganimi iskalnimi nizi in prikazanimi predlaganimi iskalnimi nizi in to razmerje shranimo v spremenljivko $razmerje. Če je produkt uteži novega predlaganega niza in spremenljivke $razmerje večji od 0,13 oz. $show_border in če polje prikazanih obrtnikov (id_show_o) ni prazno, se ta novi predlagani iskalni niz predlaga obiskovalcu.

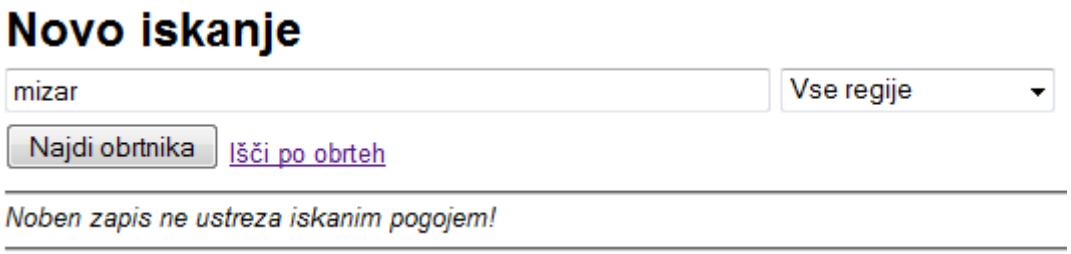
V vrsticah od 32 do 34 povečamo število prikazanih novih predlaganih iskalnih nizov (new_query_shown) za ena in to vrednost shranimo v nov predlagan iskalni niz. Preden

funkcija vrne spremenljivko $text, še v vrstici 45 odstranimo zadnji presledek in zadnjo vejico.

5. Rezultati diplomske naloge

V tem poglavju je predstavljeno delovanje aplikacije, in sicer na primeru iskalnih nizov *mizar* in *mizarstvo*. Primer prikazuje, na kakšne načine vse lahko algoritem pomaga obiskovalcu. To je preko predlaganih novih iskalnih nizov in preko predlaganja podkategorij, ki so najbolj zastopane med predstavljeni samostojnimi podjetniki. Na koncu smo še ročno izračunali in preverili vrednosti za formulo TF-IDF.

Oglejmo si sedaj še, kako algoritem deluje na primeru iskalnega niza *mizar*. Primer je zanimiv, ker nihče od samostojnih podjetnikov v svoji spletni predstavitvi ne uporablja besede *mizar* ali njene morfološke različice. Ko obiskovalec prvič vpiše iskalni niz *mizar*, med rezultati ni nobenega samostojnega podjetnika oz. njegove predstavitve. Rezultat tega iskalnega niza je prikazan na sliki 8. Za razliko od besede *mizar*, je beseda *mizarstvo* in njene morfološke različice dobro zastopana v predstavitvi mizarjev.

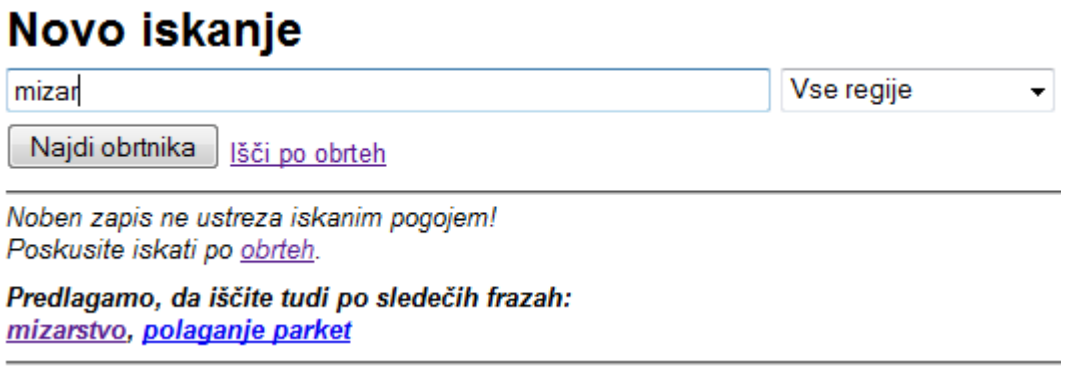


Slika 8: Za iskalni niz *mizar* ni zadetkov.

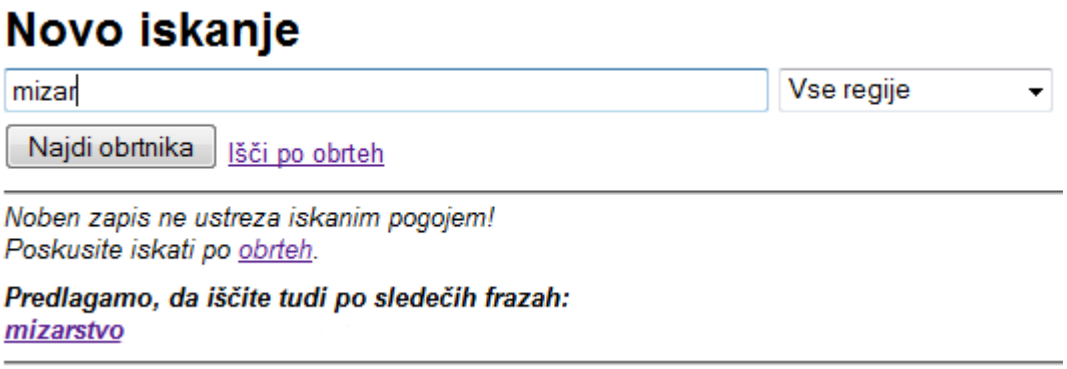
V nadaljevanju bomo pogledali, kaj se zgodi, če obiskovalec vpiše najprej iskalni niz *mizar* in nato *mizarstvo*. V drugem primeru bo obiskovalec vpisal iskalni niz *mizar* in *polaganje parketa*.

Predvidevamo naslednje: naključnemu obiskovalcu, ki bo v nadaljevanju še kdaj vpisal iskalni niz *mizar*, se bosta najprej pojavila dva predloga za nov iskalni niz, saj algoritem še ne bo razločeval med ustreznim in neustreznim novim predlaganim iskalnim nizom. Iskalna niza bosta *mizarstvo* in *polaganje parketa*. Ker se v tabelo baze shranjujejo lematizirani iskalni nizi, se bosta obiskovalcu dejansko predlagali njuni lematizirani različici. Na sliki 9 vidimo, kako se obiskovalcu predlagata oba nova iskalna niza. Ker je *mizarstvo* bolj smiselni izraz kot *polaganje parketa*, predvidevamo, da bo po 20 osnovnih prikazih kot nov iskalni niz predlagan le še niz *mizarstvo*. Ker predlagani iskalni niz *polaganje parketa* ne bo zadostil enačbi, se bo ta predlog opustil. Spomnimo se, da se glede na predhodni iskalni

niz predlog za nov iskalni niz prikaže obiskovalcu najmanj 20 krat. Po prvih 20 prikazih se izračuna, ali predlagan novi iskalni niz po formuli iz poglavja 3 presega prag za prikaz. Slika 11 prikazuje, kako se obiskovalcu predlaga le še en nov iskalni niz, in sicer *mizarstvo*. Neodvisno od predlaganja novega iskalnega niza se bo obiskvalcu ponudilo ob vpisu iskalnega niza *mizarstvo*, da lahko pogleda tudi podkategorijo *Mizarstvo*, saj sklepamo, da bo največ samostojnih podjetnikov iz te podkategorije.



Slika 9: Obiskovalcu se na začetku učenja algoritma predlagata dva nova iskalna niza.



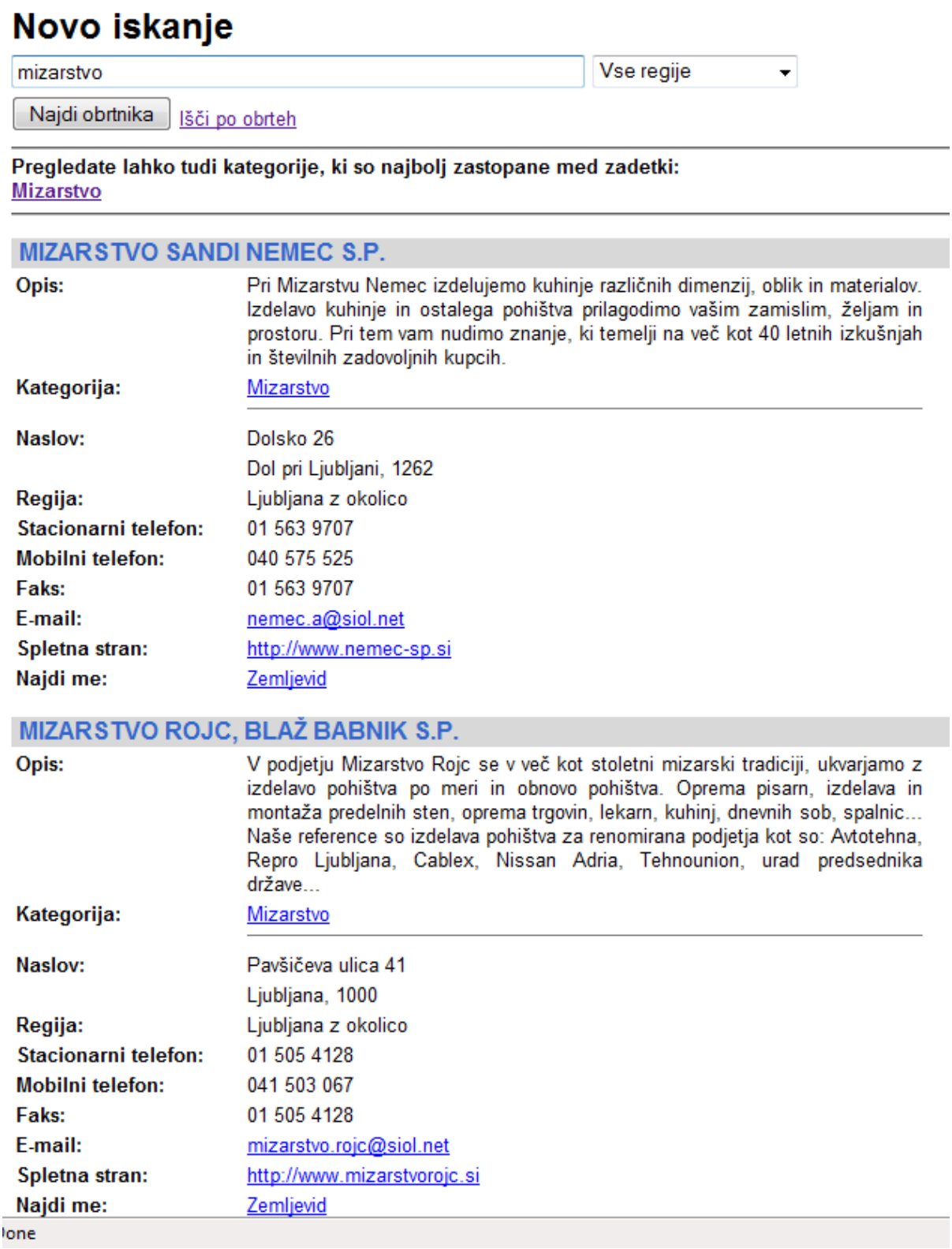
Slika 10: Že »naučeni« algoritem predlaga tisti iskalni niz, ki je ustreznejši.

V zgornjem primeru smo na primeru *mizar, mizarstvo* in *polaganje parketa* pokazali, kako algoritem deluje. S primerom smo pokazali, kako lahko na podlagi preteklih iskalnih nizov obiskovalcev novemu obiskovalcu na podlagi njegovega iskalnega niza svetujemo, kaj lahko še vpiše, da bo lahko prišel do podatkov samostojnega podjetnika, ki ga išče. Na sliki 11 vidimo, kako se obiskovalcu glede na njegov iskalni niz predlagajo podkategorije. Slika tudi prikazuje predstavitev prvih dveh od petih samostojnih podjetnikov na prvi strani. Na vsaki strani je namreč predstavljeno največ pet samostojnih podjetnikov.

Drugi predlagani iskalni niz *polaganje parketa* na sliki 9 prikazuje uspešen primer lematizacije: parketa → parket.

Za zaključek tega poglavja se posvetimo še formuli TF-IDF in izračunajmo TF-IDF vrednost za oba mizarja na sliki 11, ter tako preverimo, če sprogramirana formula opravlja svoje delo. Vseh samostojnih podjetnikov je 244 (D). Število samostojnih podjetnikov, ki imajo v svojem nazivu ali predstavitvi dejavnosti besedo *mizarstvo*, je 11 ($f_{wi,D}$). Oba

samostojna podjetnika imata besedo *mizarstvo* omenjeno 2 krat ($f'_{wi,d}$). Prvi ima v svojem nazivu in opisu dejavnosti 35 besed, drugi 52 ($f'_{ai,d}$). Pri tem moramo vedeti, da štejemo lematizirane besede samo enkrat. Izračunajmo sedaj TF-IDF $w_{mizarstvo}$ vrednost s sledečo enačbo: $w_{mizarstvo} = \frac{f'_{wi,d}}{f'_{ai,d}} * \ln\left(\frac{|D|}{f_{wi,D}}\right)$. Za prvega dobimo $w_{mizarstvo,1} = 0{,}18$. Za drugega pa $w_{mizarstvo,2} = 0{,}12$. Iz tega primera lahko sklepamo, da formula TF-IDF deluje pravilno.



Slika 11: Prikaz dela izpisa samostojnih podjetnikov in predlaganje obiskovalcu, da pogleda podkategorijo *mizarstvo*.

6. Zaključek

Za sodobne spletne strani je pozitivna uporabniška izkušnja vse bolj pomembna. Ni dovolj, da je spletna stran samo lično oblikovana in da vsebuje vse informacije, ki obiskovalca zanimajo. Zato smo v diplomski nalogi izdelali algoritem, ki obiskovalcem spletne strani najdiObrtnika.si izboljša uporabniško izkušnjo.

Algoritem omogoča obiskovalcem spletne strani boljše in bolj natančno iskanje samostojnih podjetnikov. To smo dosegli z vpeljavo postopka lematizacije iskalnega niza ter nazivov samostojnih podjetnikov in njihovih opisov dejavnosti. Z lematizacijo smo odstranili morfološko raznolikosti besed ter tako dobili manjši obseg besed, ohranili pa smo informacijo. Nato smo vzeli lematiziran iskalni niz obiskovalca in njegove besede določili kot ključne besede za iskanje po lematiziranih nazivih in opisih dejavnosti samostojnih podjetnikov. V ta namen smo uporabili formulo TF-IDF. Formula TF-IDF vrne vrednost, ki pove, kako pomembna je beseda ali več besed v besedilu. Sestavljena je iz produkta frekvence besede v izbranem besedilu in inverzne frekvence besede v vseh besedilih. Inverzna frekvenca besede v vseh besedilih deluje kot utež na frekvenco besede v izbranem besedilu. Bolj kot je beseda zastopana v vseh besedilih, manjša je njena teža za tisto besedilo in obratno.

Drugi sklop zaobjema neke vrste učenje algoritma na podlagi shranjevanja v dnevnik vpisanih iskalnih nizov obiskovalcev in njihovo vrednotenje. Dva zaporedna iskalna niza smo obravnavali kot par v obliki predhodni in novi iskalni niz. Če je naključni obiskovalec v prihodnosti vpisal predhodni iskalni, niz se mu je, če so bili pogoji izpolnjeni, predlagal tudi novi, tj. predlagani iskalni niz. Glede na odziv obiskovalcev spletne strani smo sproti vrednotili uporabnost predlaganega iskalnega niza. V kolikor predlagani iskalni niz ni dosegal želenih vrednosti smo njegovo predlaganje opustili.

Algoritem obiskovalcem tudi predlaga, da pregledajo samostojne podjetnike v podkategorijah, ki so trenutno najbolj zastopane med zadetki iskalnega niza.

V zadnjem delu diplomske naloge smo želeli uporabiti slovenski semantični leksikon, imenovan sloWNet, avtorice Darje Fišer. Z njim smo želeli poiskati sopomenke za besede iskalnega niza obiskovalcev ter nato tudi sopomenke iskati v nazivu in opisu dejavnosti samostojnih podjetnikov. Žal besedišča slovenskega semantičnega leksikona nismo mogli uporabiti, saj se je premalo pokrivalo z besediščem nazivov in opisov dejavnosti samostojnih podjetnikov. Zato se nam njegova uporaba ni zdela smiselna.

Glede na prikazan primer v poglavju 5. smatramo, da smo dosegli zastavljene cilje diplomske naloge. Menimo, da so obiskovalci spletne strani bolj zadovoljni z rezultati

algoritma za iskanje samostojnih podjetnikov v bazi in da hitreje najdejo podatke o želenem samostojnem podjetniku. Za izboljšanje uporabniške izkušnje se nam zdi ključno, da smo uporabili lematizacijo in nato TF- IDF. Zaradi omejenega obiska spletne strani predlaganje iskalnih nizov še ne pride do izraza.

7. Seznam slik

8. Seznam uporabljenih virov

[1] Wikipedia, PageRank, dostopna na: http://en.wikipedia.org/wiki/PageRank

[2] Sašo Džeroski, Tomaž Erjavec, Strojno učenje lematizacije neznanih slovenskih besed. *Jezikovne tehnologije: zbornik konference: proceedings of the conference*, Ljubljana, 2000.

[3] Joël Plisson, Dunja Mladenić, Nada Lavrač, Tomaž Erjavec, A Lemmatization Web Service Basedon Machine Learning Techniques. *Proceedings of 2nd Language and Technology Conference*, Poznan, 2005.

[4] Queries Juan Ramos, Using TF-IDF to Determine Word Relevance in Document, iCML 2003 Language Workshop, dostopno na: http://www.cs.rutgers.edu/~mlittman/courses/ml03/iCML03/papers/ramos.pdf

[5] Matjaž Juršič, *Implementacija učinkovitega sistema za gradnjo, uporabo in evaluacijo lematizatorjev tipa RDR*. Diplomsko delo, Fakulteta za računalništvo in informatiko, Ljubljana, 2007.

[6] Darja Fišer, *Izdelava slovenskega semantičnega leksikona z uporabo eno- in večjezičnih jezikovnih virov*. Doktorska disertacija, Filozofska fakulteta, Ljubljana, 2009

[7] PHP: fsockopen – Manual, dostopno na:

http://www.php.net/manual/en/function.fsockopen.php#49938

[8] ToTaLe analyser, dostopno na: http://nl2.ijs.si/analyze/

[9] Tomaž Erjavec, Simon Krek: Oblikoskladenjske specifikacije in označeni korpusi JOS. *Zbornik Šeste konference Jezikovne tehnologije*, 2008, Ljubljana.

[10] George A. Miller, WordNet: A Lexical Database for English. *Communications of the ACM Vol. 38, No. 11: 39-41*, New York, 1995.