Laboratorij za analizu teksta i inženjerstvo znanja – TakeLab

Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva
Zavod za elektroniku, mikrolektroniku, računalne i inteligentne sustave
Unska 3, 10000 Zagreb, Hrvatska

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 461

Primjena metoda strojnog učenja za poboljšanje pretraživanja dokumenata

Ivan Krišto

Zagreb, lipanj 2012.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA
ODBOR ZA DIPLOMSKI RAD PROFILA

Zagreb, 5. ožujka 2012.

DIPLOMSKI ZADATAK br. 461

Pristupnik: **Ivan Krišto**
Studij: Računarstvo
Profil: Računarska znanost

Zadatak: **Primjena metoda strojnog učenja za poboljšanje pretraživanja dokumenata**

Opis zadatka:

Sustavi za pretraživanje informacija nezaobilazan su dio suvremene informacijsko-komunikacijske infrastrukture. Razvijene su mnoge tehnike za poboljšanje rezultata pretraživanja kojima je cilj bolje odgovoriti na informacijsku potrebu korisnika, na način da se poveća odziv tražilice te da relevantni dokumenti budu visoko rangirani. U posljednje su vrijeme istraživanja usredotočena na primjenu metoda strojnog učenja na temelju zapisa o pretraživanju (engl. search logs), uključivo i podataka o klikovima (engl. click data).
U okviru diplomskog rada potrebno je proučiti osnovne modele za pretraživanje tekstnih dokumenata, standardne načine i mjere za vrednovanje uspješnosti pretraživanja, tehnike za poboljšanje rezultata pretraživanja te tehnike za učenje rangiranja na temelju zapisa o pretraživanju. Razviti metodu za predlaganje modificiranog upita temeljem upita koji je unio korisnik i zapisa o pretaživanju drugih korisnika. Razviti metodu za učenje rangiranja na temelju zapisa o pretraživanju. Implementirati metode u sustavu Apache Lucene i ispitati njihov rad na odgovarajućim skupovima podataka (zapisima tražilica AOL, CADIAL, search-lucene.com ili search-hadoop.com). Razmotriti uporabu tehnologije MapReduce za obradu velike količine tekstnih podataka. Provesti vrednovanje razvijenih metoda na temelju usporedne analize ili analize na ispitnoj zbirci. Po potrebi razviti manju ispitnu zbirku s ocjenama relevantnosti za dokumente koje pretražuje tražilica CADIAL. Radu priložiti izvorni programski kod, programsku dokumentaciju i korištene skupove podataka.

Zadatak uručen pristupniku: 9. ožujka 2012.
Rok za predaju rada: 21. lipnja 2012.

Mentor:

Doc.dr. sc. Jan Šnajder

Djelovođa:

Prof.dr.sc. Domagoj Jakobović

Predsjednik odbora za diplomski rad profila:

Prof.dr.sc. Siniša Srbljić

Sadržaj

1. Uvod

U svrhu zadovoljavanja informacijske potrebe korisnici računala najčešće pretražuju velike baze tekstnih dokumenata. Zbog nejednoznačnosti sintakse i semantike prirodnih jezika, pretraživanje je često otežano. Primjerice, zbog istoznačnica (sinonima) korisnik može izgraditi upit koristeći jedan oblik traženog pojma dok baza tražilice taj pojam ima spremljen pod drugim oblikom. Primjer su par riječi "jezikoslovlje" i "lingvistika."

Sustavi za pretraživanje informacija (engl. *Information Retrieval,* IR) često se zasnivaju na funkcijama rangiranja dokumenata po relevantnosti koje nemaju mogućnost prepoznavanja istoznačnica (sinonima), višeznačnica i ostalih semantičkih odnosa među leksemima. Današnja brzina stvaranja novih informacija dovodi do izražaja nemogućnost tražilica da se nose s nejednoznačnošću sintakse i semantike prirodnog jezika. Navedeni problem tražilica očituje se na primjeru istoznačnica i višeznačnica: većim brojem raspoloživih dokumenata pokriva se veći broj istoznačnica (ali, također se uvode nove višeznačnice) te kvaliteta tražilica opada.

U svrhu obogaćivanja funkcija rangiranja dokumenata po relevantnosti mogu se iskoristiti metode strojnog učenja (engl. *machine learning*) i obrade prirodnog jezika (engl. *natural language processing*). Cilj je ukomponirati u funkciju rangiranja vezu između semantički sličnih pojmova (ili naglasiti semantičku različitost nekih pojava višeznačnica). Automatsko povezivanje semantički sličnih pojmova izuzetno je težak posao: problem predstavlja činjenica da se svakodnevno pojavljuju novi pojmovi i nove veze između njih. Zbog toga je nastala ideja da se te nove veze otkrivaju automatski iz indirektnog djelovanja velikog broja korisnika.

U ovom radu istražuje se mogućnost korištenja zapisa dnevnika upita pretraživača dokumenata (u literaturi poznatog kao *search engine query log*, odnosno *click-data* ili *clickthrough data*) za potrebe poboljšavanja pretraživanja dokumenata. Poboljšavanje pretraživanja manifestira se predlaganjem boljih upita koji iskorištavaju novonaučne veze među pojmovima te brže zadovoljavaju korisnikovu

informacijsku potrebu, kao i izravnim poboljšavanjem rangiranja bez potrebe za korisnikovom intervencijom.

U ovom radu naglasak je na dva zadatka: (1) predlaganje boljeg upita, te (2) poboljšavanje rangiranja. Oba zadatka temelje se na informacijama dobivenim iz dnevnika upita tražilice. Za potrebe ovog rada, smatra se da zapis dnevnika upita tražilice sadrži sljedeće podatke:

- korisnički upit,
- identifikacijska oznaka odabranog dokumenta (ili ništa ako dokument nije odabran),
- vrijeme nastanka upita (engl. *query timestamp*),
- identifikacijska oznaka korisnika.

Pri analizi pristupa poboljšanju pretraživanja bitno je uzeti u obzir tip podataka nad kojima se metode pristupa primjenjuju. Za potrebe metoda poboljšanja pretraživanja uglavnom se koriste *povratne informacije* korisnika ili eksperta. Pri tome možemo razlikovati: (1) *eksplicitnu povratnu informaciju* i (2) *implicitnu povratnu informaciju*. *Eksplicitna povratna informacija* se skuplja na način da se izravno pita korisnika koji dokumenti su relevantni za njegovu informacijsku potrebu ili zapošljavanjem stručnjaka da odredi relevantnost skupa dokumenata za skup informacijskih potreba. Prikupljanje eksplicitnih povratnih informacija izuzetno je skup proces, a često može doći do niskog slaganja među korisnicima ili među stručnjacima. *Implicitna povratna informacija* se dobiva iz korisnikove interakcije sa sustavom pri čemu nema potrebe za izravnim postavljanjem pitanja o relevantnosti dokumenata. Primjer implicitne povratne informacije je kupovina knjiga u Internetskoj knjižari. Ako korisnik kupi neku knjigu tada možemo pretpostaviti da ju je kupio jer mu se sviđa (zbog autora, teme, žanra, preporuke i slično). Na temelju skupa kupljenih knjiga (skupa implicitnih povratnih informacija) možemo definirati mjeru sličnosti dvaju korisnika te izgraditi sustav predlaganja knjiga koji bi korisniku predlagao knjige koje je kupio njemu sličan korisnik.

Kod pretraživača dokumenata implicitnu povratnu informaciju možemo crpiti iz dnevnika upita pretraživača, pri čemu se možemo pretpostaviti da vrijedi: *ako je korisnik za zadani upit odabrao neki dokument, tada je taj dokument relevantan za zadani upit*. Budući da dnevnik upita može sadržavati veliku količinu šuma, te da je obrazac korištenja pretraživača dokumenata puno složeniji od onoga što navedena pretpostavka implicira, dokumente na koje nas upućuje implicitna povratna informacija ne možemo proglasiti potpuno relevantnima već ih prozivamo

pseudo-relevantnim.

Joachims et al. (2005) navodi da čest obrazac korištenja tražilica uključuje čitanje rezultata s vrha prema dnu te da korisnici obično ne čitaju sve rezultate. Navedeni obrazac korištenja tražilica navodi da korisnikovim odabirom dokumenta s rangom k, dokumente s rangom od 1 do $k-1$ možemo smatrati manje relevantnima od odabranog dokumenta, ali za dokumente s rangom većim od k nemamo dovoljno informacija. Implicitnu povratnu informaciju, u slučaju tražilica, moramo smatrati relativnom mjerom. Stoga, koristimo termin *implicitna relevantna povratna informacija* te njenom primjenom ne dobivamo relevantne dokumente već pseudo-relevantne dokumente.

Tražilica ne mora biti ograničena na tekstne dokumente, ovdje je dokument apstraktan pojam te se pod njim smatra sve što je moguće pretraživati (tekstni dokumenti, slike, zvuk, video, web dokumenti, i sl.).

Cilj rada jest napraviti implementacije sustava za predlaganje upita te sustava za poboljšavanje rangiranja koji su iskoristivi u biblioteci Apache Lucene.[1] Lucene je biblioteka za pretraživanje dokumenata. Radi se o projektu otvorenog koda pod organizacijom Apache.

U nastavku rada opisani su tradicionalni modeli pretraživanja dokumenata. U trećem poglavlju opisan je problem učenja rangiranja uz navođenje postignuća u navedenom području i opis razvijenih metoda. U četvrtom poglavlju opisan je problem predlaganja upita i metode razvijene u sklopu ovog rada. Poglavlje 5 opisuje korištene skupove podataka. Slijede evaluacija te zaključak i prijedlozi daljni rad.

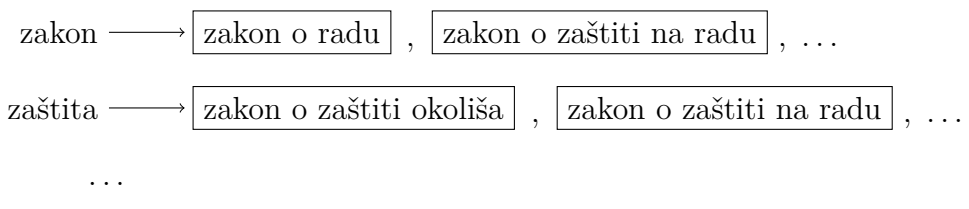
[1] `http://lucene.apache.org/`

2. Osnovni modeli pretraživanja dokumenata

Zbog široke primjenjivosti i korisnosti područje pretraživanja dokumenata se dugo i intenzivno istražuje, tome svjedoče rezultati i brojnost razvijenih metoda. Opširan i detaljan uvid u područje pretraživanja informacija dan je u (Manning et al., 2008).[1] Danas najveći izazov donosi masivnost podataka, odnosno problem skaliranja sustava za pretraživanje informacija na velike količine podataka. Nakon svake inovacije u području pretraživanja potrebno je postaviti pitanje "*Hoće li metoda raditi na velikoj količini podataka?*" Opširna razrada problematike rada s velikim količinama podataka dana je u knjigama (Rajaraman i Ullman, 2011)[2] i (Lin i Dyer, 2010).[3]

U nastavku slijedi sažet opis osnovne podjele IR modela. Treba imati na umu da je svaki model podložan velikom broju modifikacija i proširenja te da postoji mogućnost kombiniranja više modela ili dodavanje odvojenih sustava koji poboljšavaju pretraživanje.

Veliki dio modela kao osnovnu podatkovnu strukturu koristi strukturu pod nazivom *invertirani indeks* (engl. *inverted index*). Radi se o strukturi koja preslikava pojmove na dokumente u kojima se oni pojavljuju. Skica takve strukture dana je na slici 2.1.



Slika 2.1: Invertirani indeks (engl. *inverted index*).

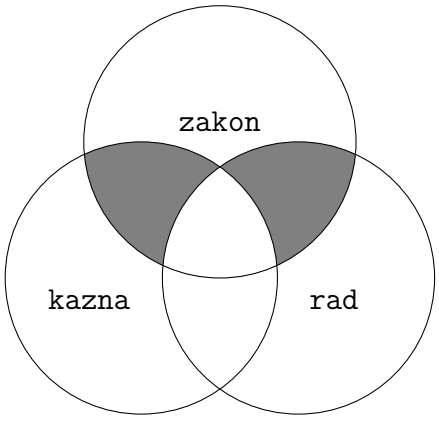
[1] Dostupna bez naknade na: `http://nlp.stanford.edu/IR-book/`.
[2] Dostupna bez naknade na: `http://infolab.stanford.edu/~ullman/mmds.html`.
[3] Dostupna bez naknade na: `http://lintool.github.com/MapReduceAlgorithms/`.

2.1. Booleov model

Booleov model pretraživanja informacija zasniva se na teoriji skupova. Njegova teorijska osnova čini ga pogodnim za korištenje u okolini koja zahtijeva egzaktni odgovor dobiven složenim upitom. Primjerice:

```
(zakon AND rad OR zakon AND kazna) AND NOT (rad AND kazna)
```

Navedeni upit dohvaća sve dokumente koji sadrže pojmove "zakon" i "rad" ili "zakon" i "kazna," osim ako ne sadrže oba pojma "rad" i "kazna." Vennov dijagram za navedeni upit prikazan je na slici 2.2. Za korištenje IR sustava temeljenih na teoriji skupova potrebno je poznavati osnove propozicijske logike, što takve sustave čini neprilagođenima široj javnosti — ovo se ne odnosi na složenost primjene propozicijske logike već na neusklađenost njenih operatora s njihovim ekvivalentima prirodnog jezika. Prirodni jezik ne poznaje prednost operatora, često zamjenjuje operatore `AND` i `OR` – `zakon I rad` se u prirodnom jeziku mogu shvatiti kao upit `zakon OR rad` te ponekad koristi veznik "*ili*" kao operator "isključivo ili" — "`hoćeš kavu ILI čaj`."



Slika 2.2: Vennov dijagram za upit iz primjera korištenja Booleovog modela.

Bitan problem Booleovog modela jest nemogućnost rangiranja dohvaćenih dokumenata. Rangiranje se može ostvariti korištenjem neizrazite logike (engl. *fuzzy logic*) umjesto propozicijske, odnosno proširenjem modela. Često proširenje modela je dodavanje operatora *blizine* (engl. *proximity*) – primjerice, "`zakon NEAR rad`," što vraća samo one dokumente kod kojih se pojmovi `zakon` i `rad` nalaze u neposrednoj blizini jedan od drugoga.

2.2. Statistički model

Statistički modeli, odnosno modeli vektorskog prostora (engl. *vector space models,* VSM) polaze od pretpostavke da je relevantnost dokumenta i upita u uskoj vezi sa frekvencijom pojavljivanja pojmova iz upita unutar dokumenta (Salton et al., 1975). Kako upit i dokument mogu dijeliti više pojmova, za upit i svaki dokument u bazi grade se vektori, pri čemu su frekvencije pojmova ukomponirane u komponente vektora. Vektori se ne grade od samih frekvencija pojmova jer bi tada duljine dokumenata i distribucija pojmova po dokumentima nepravedno utjecala na rezultate pretraživanja. Radi toga se komponente grade od težinskih mjera poput mjere *tfidf*:

$$\text{tfidf}(t, \mathbf{d}) = \text{tf}(t, \mathbf{d}) \cdot \text{idf}(t), \tag{2.1}$$

pri čemu su t i $\mathbf{d}$ pojam i dokument za koje računamo mjeru, $\text{tf}(t, \mathbf{d})$ je frekvencija pojavljivanja pojma t u dokumentu $\mathbf{d}$ (engl. *term frequency*), a $\text{idf}(t)$ je inverzna frekvencija dokumenata za pojam t (engl. *inverse document frequency*). Inverzna frekvencija dokumenata za t definirana je sa:

$$\text{idf}(t) = \log \frac{N}{\text{df}(t)}, \tag{2.2}$$

pri čemu je N ukupan broj dokumenata u bazi, a $\text{df}(t)$ broj dokumenata u kojima se pojavljuje pojam t. Baza logaritma u navedenom izrazu ne utječe na svojstva mjere.

Rangiranje se dobiva određivanjem sličnosti svakog dokumenta s upitom. Često korištena mjera sličnosti je kosinus kuta između vektora upita i dokumenta:

$$\text{sim}(\mathbf{q}, \mathbf{d}) = \cos\theta, \tag{2.3}$$

pri čemu je $\mathbf{q}$ vektor upita, $\mathbf{d}$ vektor dokumenta, a θ kut između ta dva vektora. Jednadžba (2.3) može se izračunati preko normaliziranog skalarnog produkta tih vektora:

$$\text{sim}(\mathbf{q}, \mathbf{d}) = \frac{\mathbf{q} \cdot \mathbf{d}}{|\mathbf{q}| \cdot |\mathbf{d}|}, \tag{2.4}$$

pri čemu je $|\cdot|$ L_2-norma vektora.

Statistički modeli često su korišteni u praksi. Za razliku od Booleovih modela, statistički modeli mogu primiti upit u obliku prirodnog jezika, ali taj upit ne može biti ekspresivan poput upita kod Booleovih modela. Primjer je težina implementacije operatora blizine kod statističkih modela. Proširenje statističkog modela

dodavanjem operatora bliskosti zahtijeva dodavanje informacije o udaljenosti među pojmovima u tekstu u vektor dokumenta te mijenjanje funkcije rangiranja. Problem nastaje zbog miješanja dvaju tipova vrijednosti unutar vektora dokumenta – vrijednosti temeljenih na frekvencijama pojavljivanja pojmova te vrijednosti temeljenih na udaljenosti među pojmovima. Da bi funkcija rangiranja razlikovala vrijednosti temeljene na frekvencijama, odnosno udaljenosti pojmova, potrebno ju je parametrizirati (pridijeliti različite težine različitim tipovima vrijednosti). Navedeni parametar povećava složenost modela jer ga je potrebno eksperimentalno odrediti ili strojno naučiti. Dodatni problem statističkih modela je ograničenost vokabulara upita – vokabular upita je vokabular unije dokumenata.

2.3. Vjerojatnosni model

Cilj vjerojatnosnog modela pretraživanja (engl. *probabilistic retrieval model*) je rangiranje dokumenata po vjerojatnosti relevantnosti za zadani upit. Ako označimo informacijsku potrebu korisnika sa I te dokument sa D, tada je vjerojatnost relevantnosti dokumenta za upit:

$$r(D) = P(I|D). \tag{2.5}$$

Ovdje se relevantnost iskazuje prema informacijskoj potrebi korisnika. Pri ovom modelu upit nema toliku ulogu kao kod statističkih modela. Moguća je situacija da upit i najrelevantniji dokument nemaju niti jedan zajednički pojam (primjerice ako se koristi *latentno semantičko indeksiranje* (engl. *latent semantic indexing,* LSI)).

Vjerojatnost relevantnosti na neograničenom skupu ne može se egzaktno izračunati. Zbog toga ne možemo reći "ako je $r(D) \geq 0.5$, dokument je relevantan." Potrebno je dodatno znanje kako bismo mogli definirati vjerojatnosti $P(R \mid \mathbf{q}, \mathbf{d})$ ($R \in \{0, 1\}$) koja određuje kolika je vjerojatnost da je dokument relevantan za upit ($R = 1$), odnosno nerelevantan (bitno je primijetiti da zbog nemogućnosti egzaktnog izračuna vjerojatnosti općenito ne vrijedi $P(R = 1 \mid \mathbf{q}, \mathbf{d}) + P(R = 0 \mid \mathbf{q}, \mathbf{d}) = 1$). Time se može definirati da je $\mathbf{d}$ relevantan ako i samo ako vrijedi:

$$P(R = 1 \mid \mathbf{q}, \mathbf{d}) > P(R = 0 \mid \mathbf{q}, \mathbf{d}). \tag{2.6}$$

Primjer vjerojatnosnih modela su jezični modeli (engl. *language models*). Opis jezičnih modela nalazi se u nastavku.

2.3.1. Jezični modeli

Pri modeliranju upita za pretraživanje dokumenata tražimo riječi i fraze koje će se vrlo vjerojatno pojaviti u relevantnom dokumentu. Jezični modeli (engl. *language models,* LM) kreću od toga na način da pretpostavljaju da je dokument relevantan ako postoji velika vjerojatnost da generira zadani upit (Ponte i Croft, 1998).

Rangiranje se vrši na način da svakom dokumentu odredimo njegov jezični model M te pomoću njega generiramo upit:

$$p(\mathbf{q}, \mathbf{d}) = \prod_i p_{\mathbf{d}}(\mathbf{q}_i), \tag{2.7}$$

pri čemu je $\mathbf{q}_i$ i-ti pojam upita, a $p_{\mathbf{d}}(\mathbf{q}_i)$ vjerojatnost da jezični model M generira $\mathbf{q}_i$. Vjerojatnost da M generira pojam t može jednostavno biti omjer frekvencije pojavljivanja pojma t u dokumentu i ukupnog broja pojmova u dokumentu.

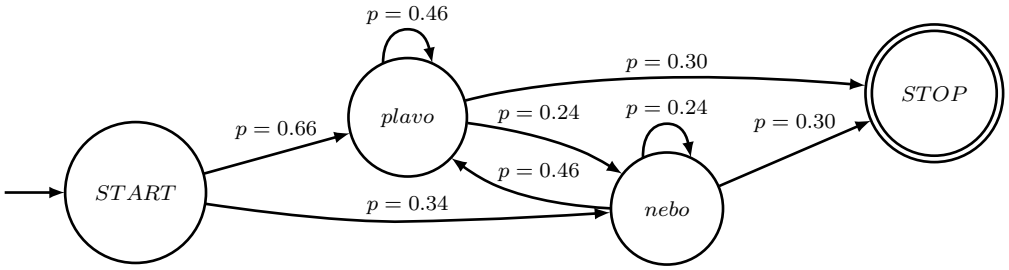
Problem koji se ističe u (2.7) je mogućnost da neki dokument ne sadrži pojam iz upita. Time je $p(\mathbf{q}, \mathbf{d}) = 0$ iako $\mathbf{d}$ može biti relevantan (primjerice, ako se samo jedan pojam iz upita ne pojavljuje). Zbog raspodijele riječi u prirodnom jeziku, ta situacija je relativno česta te se zbog toga uvodi *zaglađivanje* (engl. *smoothing*) modela. Radi se o dodjeljivanju malih vjerojatnosti pojmovima koji se ne pojavljuju u dokumentu (Zhai i Lafferty, 2004; Chen i Goodman, 1999).

Teorijska osnova jezičnih modela zasniva se na teoriji automata. Ako se automatima modelira gramatika koja generira jezik, tada je potrebno definirati kriterije zaustavljanja generiranja (prihvatljiva stanja). Kod jezičnih modela može se odrediti vjerojatnost zaustavljanja p_s te tada vjerojatnost da dokument generira upit duljine n iznosi:

$$p(\mathbf{q}, \mathbf{d}) = \left(\prod_{i=1}^{n-1} p_{\mathbf{d}}(\mathbf{q}_i) \cdot (1 - p_s) \right) \cdot p_{\mathbf{d}}(\mathbf{q}_n) \cdot p_s. \tag{2.8}$$

Ukratko, dokument generira $n-1$ pojam uz vjerojatnost $p_{\mathbf{d}}(\mathbf{q}_i) \cdot (1 - p_s)$, te zadnji pojam uz vjerojatnost $p_{\mathbf{d}}(\mathbf{q}_n) \cdot p_s$. Vjerojatnost zaustavljanja u jezične modele uvodi ograničenje duljine generiranog niza pojmova, odnosno pristranost prema nekim duljinama nizova. Jezični model se može prikazati kao automat sa početnim stanjem koje vodi u sva ostala stanja automata te jednim prihvatljivim stanjem u koje vode sva stanja uz vjerojatnost p_s. Ako se zanemari vjerojatnost zaustavljanja, takav automat nikad ne bi završio generiranje, a samim time niti jedan konačni niz ne bi bio prihvatljiv. Primjer automata koji predstavlja jezični model prikazan je na slici 2.3. Navedeni automat može generirati dokument koji sadrži pojmove "plavo" i "nebo." Automat generira pojam "plavo" uz vjerojatnost $0.66 \cdot (1 - p_s)$, te

pojam "nebo" uz vjerojatnost $0.34 \cdot (1 - p_s)$. Vjerojatnost zaustavljanja generiranja je $p_s = 0.30$, što znači da je prosječna duljina generiranih dokumenata tri pojma. U praksi, jezični modeli sva stanja proglašavaju prihvatljivima te se vjerojatnost zaustavljanja često zanemaruje (Manning et al., 2008).



Slika 2.3: Jezični model prikazan pomoću automata.

Izraz (2.7) prikazuje generiranje upita od strane *unigramnog* modela, odnosno modela koji pretpostavlja da su svi pojmovi međusobno nezavisni. Zbog nezavisnosti takav model nije u mogućnosti koristiti kontekst, odnosno prepoznavati fraze, kolokacije i ostale grupacije pojmova. Uvođenjem više-gramskih modela proširuje se ekspresivnost modela, ali se uvodi dodatna greška u model jer, osim procjene vjerojatnosti da dokument generira jedan pojam, više-gramski model mora procijeniti vjerojatnost da se taj pojam generira unutar određenog konteksta. Radi navedenog, unigramski model je precizniji od više-gramskog. Primjer računanja vjerojatnosti generiranja upita kod *bigramskog* modela bez stop-vjerojatnosti:

$$p(\mathbf{q}, \mathbf{d}) = p_{\mathbf{d}}(\mathbf{q}_1) \cdot \prod_{i=2}^{n} p_{\mathbf{d}}(\mathbf{q}_i | \mathbf{q}_{i-1}). \tag{2.9}$$

2.4. Model korišten u Lucenu

Lucene koristi statistički model (VSM) uz mjeru sličnosti (Hatcher et al., 2010):

$$\begin{aligned} \mathrm{sim}(\mathbf{q}, \mathbf{d}) = \sum_{t \in \mathbf{q}} (&\mathrm{tf}(t, \mathbf{d}) \cdot \mathrm{idf}(t)^2 \cdot \mathrm{boost}(t.field \in \mathbf{d}) \\ &\cdot \mathrm{lenNorm}(t.field \in \mathbf{d})) \\ &\cdot \mathrm{coord}(\mathbf{q}, \mathbf{d}) \cdot \mathrm{queryNorm}(\mathbf{q}), \end{aligned} \tag{2.10}$$

pri čemu je $\mathbf{q}$ vektor upita, $\mathbf{d}$ vektor dokumenta čiju sličnost s upitom tražimo, $\mathrm{tf}(t, \mathbf{d})$ frekventnost pojma t u dokumentu $\mathbf{d}$, $\mathrm{boost}(t.field \in \mathbf{d})$ pojačanje vrijednosti polja (svako polje, odnosno zona ili regija, može imati statičnu vrijednost

pojačanja pri indeksiranju tako da vrijedi više, odnosno manje, pri pretraživanju), lenNorm($t.field \in \mathbf{d}$) normalizirana duljina vrijednosti polja (polja s kraćom vrijednošću su važnija pri pretraživanju), coord($\mathbf{q}, \mathbf{d}$) koordinacijski faktor (dokumenti koji sadrže više pojmova iz upita dobivaju veću važnost pri pretraživanju), a queryNorm($\mathbf{q}$) normalizacija vrijednosti upita.

3. Učenje rangiranja

Učenje rangiranja (engl. *learning to rank*) je prisup automatskoj izgradnji funkcije rangiranja dokumenata po relevantnosti. Automatizacija izgradnje funkcije rangiranja ostvaruje se korištenjem metoda strojnog učenja. Podatci nad kojima se uči funkcija rangiranja mogu biti dokumenti s ocjenama relevantnosti, dnevnik upita tražilice, kombinacija dokumenata i vanjskih resursa s opisima semantičkih veza (primjerice, iskorištavanje veza na Wikipediji[1] za povezivanje visoko rangiranih dokumenata i njihovih semantički sličnih dokumenata) te ostali resursi koji nose informaciju o relevantnosti dokumenata ili semantičkim vezama među njima.

Potreba za učenjem rangiranja proizlazi iz dva problema:

- relevantnost se mijenja s vremenom (primjerice, upit "*predsjednik Republike Hrvatske*" ne može uvijek vraćati istu osobu),
- metode pretraživanja informacija opisane u 2. poglavlju nisu optimalne za svaki skup dokumenata i svaki skup korisnika; učenjem rangiranja možemo optimirati pretraživanje informacija za jedan skup dokumenata i pripadni skup korisnika.

Možemo izdvojiti tri tipa naučene funkcije rangiranja:

- funkcija $f(d, q)$ koja izražava relevantnost dokumenta d za upit q (*pointwise* pristup),
- funkcija $f(d_1, d_2, q)$ koja izražava odnos $f(d_1, q) \gtreqless f(d_2, q)$ (*pairwise* pristup),
- funkcija $f(l, q)$ koja za upit q i rangiranu listu dokumenata l vraća permutiranu listu l' (*listwise* pristup).

Liu (2011) navodi mogućnost uklanjanja ovisnosti funkcije rangiranja od upita.

Objavljen je znatan broj radova koji obrađuju prednosti i nedostatke te mogućnosti zapisa dnevnika upita tražilica kao izvora za rangiranje (Singla i White,

[1] `http://www.wikipedia.org/`

2010; Dou et al., 2008; Ji et al., 2009; Joachims, 2002; Radlinski i Joachims, 2007; Cao et al., 2010).

Radlinski i Joachims (2007) bave se problemom grupiranosti zapisa dnevnika upita na prve rezultate tražilice (korisnici rijetko kad pregledavaju veći broj rezultata). Problem učenja rangiranja svode na traženje funkcije koja izražava relevantnost para dokumenata (*pairwise* pristup). Problem pristranosti korisnika prvim rezultatima tražilice rješavaju postavljanjem različitih dokumenata na prva dva ranga tražilice. Prilikom skupljanja podataka za učenje funkcije rangiranja, odabiru se dva dokumenta (koriste nekoliko metoda, primjerice slučajni odabir, odabir para dokumenata za koje je greška učenja najveća). Odabrani dokumenti postave se na prva dva ranga liste rezultata tražilice. Time se, koristeći korisnikovu pristranost prvim rezultatima tražilice, dobiva informacija o relativnoj relevantnosti odabranih dokumenata. Osnovni doprinos rada jest metodologija bržeg skupljanja podataka za učenje te uklanjanje dijela šuma kod podataka za učenje. Problem opisanih metoda jest potreba za modificiranjem rezultata tražilice zbog skupljanja podataka za učenje.

Joachims (2002) koristi zapise dnevnika upita kombiniranih rezultata više tražilica te uči funkciju rangiranja korištenjem Ranking SVM-a.

Singla i White (2010) tvrde da su zapisi dnevnika upita previše nehomogeni po pitanju parova "(upit, dokument)." Po njihovim podacima 90% parova se pojavljuje samo jednom te 5% dva puta. Kao rješenje problema nehomogenosti predlažu filtriranje kvalitetnih parova "(upit, dokument)." Predložene metode uključuju značajke poput entropije upita, prosječnog zadržavanje korisnika na odabranom dokumentu te duljine navigiranja korisnika po odabranom dokumentu (tj. duljina puta po dokumentima koji korisnik napravi počevši od odabranog dokumenta). Sve navedene značajke, osim entropije upita, zahtijevaju praćenje korisnika nakon napuštanja tražilice (što često nije poželjno ponašanje sustava). Entropija upita se računa na sljedeći način:

$$\log_2 e(q) = -\sum_d \bar{c}(q,d) \cdot \log(\bar{c}(q,d)), \tag{3.1}$$

pri čemu je q upit, $\bar{c}(q,d)$ normalizirani broj pojavljivanja para (q,d) u dnevniku upita tražilice. Ako je $c(q)$ broj pojavljivanja upita q u dnevniku upita tražilice, tada vrijedi: $\bar{c}(q,d) = \frac{c(q,d)}{c(q)}$. Singla i White (2010) navode da među navedenim značajkama samo entropija upita daje dobre rezultate (pri filtriranju koriste prag $e(q) = 1$). No, navode da kombiniranjem svih značajki postižu poboljšanje u preciznosti od 20%.

Joachims et al. (2005) iznosi detaljno istraživanje kvalitete zapisa dnevnika upita za potrebe učenja rangiranja. Glavni zaključci koje iznosi su:

- korisnici rezultate tražilice pregledavaju od vrha prema dnu;
- korisnici češće odabiru visoko rangirane dokumente bez obzira na njihovu relevantnost – fenomenen koji nazivamo pristranost zbog povjerenja (engl. *trust bias*);
- ukoliko je kvaliteta rangiranja tražilice loša, zapisi dnevnika upita će također biti niske kvalitete (engl. *quality bias*);
- zapisi dnevnika upita su implicitna relativna povratna informacija;
- ako se zapisi dnevnika upita smatraju implicitnom *relativnom* povratnom informacijom, tada je slaganje s eksplicitnom povratnom informacijom 80.8% (što je jako dobar rezultat jer je slaganje među označivačima pri izradi eksplicitne povratne informacije 89.5%).

Prilikom razmatranja radova na temu korištenja dnevnika upita tražilica kao implicitne povratne informacije potrebno je obratiti pažnju koristi li metoda informacije dobivene iz meta-tražilica ili od ljudskih označivača. Metode koje koriste meta-tražilice često uzimaju značajke specifične samo za meta-tražilice, poput "rang dokumenta na 1. tražilici," "dokument se pojavio unutar top 10 dokumenata na bar tri tražilice" i slično. Takve značajke nisu primjenjive u općenitom slučaju. Cilj ovog rada je razviti metodu potpuno neovisnu o ljudskom faktoru ili drugim tražilicama, stoga se radovi koji uključuju druge tražilice ili ljudske označivače samo površno uzimaju u razmatranje.

Logovi tražilica pokazali su se korisnima za poboljšavanje rangiranja čak i kad se koriste relativno jednostavne metode. Primjer tomu je komercijalni dodatak za poboljšavanje rangiranja Apache Solra[2] koji je razvijen od strane Lucid Imaginationsa (Białecki, 2011). Iz dnevnika upita tražilice izdvojili su karakteristične fraze unutar upita te njima proširili odabrane dokumente (dodali su nova polja u indeks tražilice za te dokumente). Zbog toga će novi upit koji sadrži frazu kojom su prošireni neki dokumenti favorizirati proširene dokumente.

Białecki (2011) navodi da je značaj zapisa dnevnika upita potrebno promatrati kroz vrijeme, odnosno da mu je potrebno postupno smanjivati utjecaj. Time se može izbjeći stacioniranje dokumenata na vrhu i omogućiti prodiranje novih, također relevantnih dokumenata na bolje pozicije u rang listi.

[2] `http://lucene.apache.org/solr/`

3.1. Učenje na temelju ocjena relevantnosti dokumenata

Ako postoji dovoljno označenih podataka moguće je naučiti funkciju rangiranja za takav tip podataka. Općeniti postupak učenja funkcije rangiranja temeljem ocjena relevantnosti je sljedeći:

a) raspolažemo skupom upita $\mathcal{Q}$;

b) za svaki upit iz $\mathcal{Q}$ poznata je ocjena relevantnosti svih dokumenata;

c) svaki dokument i svaki upit prikažemo pomoću vektora značajki [3] (Liu (2011) predlaže korištenje značajki dokumenta bez značajki upita – time pokušava izbjeći prenaučenost funkcije rangiranja, odnosno pokušava generalizirati funkciju rangiranja da je primjenjiva na neviđene upite);

d) odabranom metodom strojnog učenja (primjerice SVM-om) učimo funkciju rangiranja f.

Problem nastaje ako se poveća raznolikost dokumenata u bazi tražilice (što će se vjerojatno dogoditi ako se baza često mijenja). Time se može poremetiti raspored razreda u vektorskom prostoru te je potrebno ponovo provesti učenje funkcije rangiranja. Ponovno učenje zahtijeva novi skup upita s ocjenama relevantnosti, što je izuzetno skupo.

Liu (2011) identificira tri različita skupa metoda za učenje funkcije rangiranja:

***Pointwise* pristup:** na ulaz metode za učenje funkcije rangiranja dovode se pojedinačni dokumenti s njihovim ocjenama relevantnosti. Naučena funkcija može odgovoriti koliko je neki dokument relevantan. Bitno je primijetiti da Liu (2011) navodi da skup *pointwise* metoda ne koristi upit kao informaciju, čime mu je ekspresivnost drastično smanjena. Ako se upit uključi kao informacija, tada dobivamo pristup sličan tradicionalnim IR metodama s tim da funkciju rangiranja ne određujemo heuristički već je učimo na temelju dokumenata s ocjenama relevantnosti.

***Pairwise* pristup:** na ulaz metode dovode se parovi dokumenata s njihovim relativnim relevantnostima. Naučena funkcija odgovara na pitanje koji je od dva dokumenta relevantniji. Kao i kod *pointwise* pristupa, skup metoda ne mora koristiti upit kao informaciju već može biti potpuno općenit.

[3] *vidi:* odjeljak 5.1

***Listwise* pristup:** na ulaz se dovode upit i lista rangiranih dokumenata te ručno podešena lista rangova. Cilj je naučiti funkciju koja će dobivenu listu rangova permutirati u listu što sličniju ručno podešenoj listi rangova. Takve metode su ovisne o upitu i najčešće korištene.

Li (2011) i Liu (2011) dali su opširan pregled metoda i dobar pregled teorijske pozadine učenja na temelju ocjena relevantnosti. Skupovi potrebni za navedene metode opisani su u odjeljku 5.1.

Uzevši u obzir visoku cijenu izrade dovoljno velikog skupa s ocjenama relevantnosti, navedene metode nisu primjerene za široku uporabu. Stoga niti nisu predmetom razmatranja ovog rada. Za akademske potrebe postoji veći broj skupova upita s ocjenama relevantnosti dostupnih bez naknade. Više o njima bit će riječi u odjeljku 5.1.

3.2. Podešavanje rangiranja omjerima

Metoda podešavanja rangiranja omjerima spada u *pointwise* pristup učenju funkcije rangiranja dokumenata. Za razliku od općenitih metoda rangiranja, navedena metoda nije u mogućnosti rangirati potpuni skup dokumenata već služi samo kao poboljšanje postojećim funkcijama rangiranja.

Metoda koristi zapise dnevnika upita tražilice te iskorištava frekvencije pojavljivanja parova "(upit, dokument)." Navedena metoda je ovisna o upitu pri rangiranju te nije u mogućnosti provesti rangiranje nad neviđenim upitima.

Metoda kreće od pretpostavke da je relevantnost dokumenta moguće dobiti iz omjera:

$$\mathrm{rel}(q, d) = \frac{c(q, d)}{c(q)}, \tag{3.2}$$

pri čemu je q upit, d dokument, $c(q, d)$ broj odabira dokumenta d za upit q te $c(q)$ broj postavljanja upita q.

Ideja je uzeti osnovnu listu koju za upit q vraća nemodificirana tražilica. Svakom dokumentu u toj listi podijeliti vjerojatnost odabira (αp_r) temeljem ranga u listi – ako razmatramo prvih k dokumenata (uzeto je $k = 20$), prvi dokument dobiva vjerojatnost $k \cdot p_r$, drugi $(k-1) \cdot p_r$ i tako do zadnjeg koji dobiva vjerojatnost p_r. Vrijedi:

$$p_r = \frac{1}{\sum_{i=1}^{k} i}. \tag{3.3}$$

Sad možemo dokumentima liste dodati novu ocjenu temeljem "osnovne vjerojatnosti" (αp_r) i njihove frekvencije u dnevniku upita tražilice ($\mathrm{rel}(q, d)$):

$$s(q, d) = \mathrm{rel}(q, d) + \alpha p_r. \tag{3.4}$$

Dokumenti se sortiraju po $s(q, d)$ te su spremni za prikaz korisniku.

U obzir treba uzeti pristranost korisnika prema prvim rezultatima tražilice (Joachims et al., 2005). To se može heuristički riješiti uvođenjem parametra za redukciju broja klikova kod najviše rangiranih dokumenata (naročito prvog, drugog i trećeg).

3.3. Podešavanje rangiranja logaritamskim korakom

Metoda kreće od pretpostavke da bi broj odabira nekog dokumenta logaritamski utječe na poboljšanje njegovog ranga. Odnosno, ako je dokument d rangiran pozicijom k za upit q i ako u dnevniku upita tražilice postoji m parova (q, d), tada bi se d za q trebao naći na poziciji $k + \lfloor \log_b m \rfloor$.

Pri izvedbi moguće je poslužiti se "osnovnom vjerojatnošću tražilice" koja je opisana u 3.2. Time se i-ti dokument u odabranih k dokumenata može bodovati na sljedeći način:

$$s(q, d_i) = p_r \cdot (k - i + 1) + p_r \cdot \log_b m. \tag{3.5}$$

Baza logaritma može biti eksperimentalno odabrana, primjerice $b = 10$. Znatan utjecaj baze osjeća se samo pri manjim pomacima ranga (za jedno ili dva mjesta, odnosno, za mjesta za koja je potrebno b, odnosno b^2 zapisa iz dnevnika upita tražilice).

Metoda također pati od pristranosti korisnika prema prvim rezultatima. Problem se može riješiti na način opisan u odjeljku 3.2.

3.4. Učenje relevantnosti temeljeno na Kullback-Leiblerovoj udaljenosti

Metoda se temelji na proširivanju jezičnog modela upita s jezičnim modelima relevantnih dokumenata (u konkretnom slučaju, pseudo-relevantnih jer njihovu

relevantnost samo pretpostavljamo na temelju zapisa u dnevniku upita tražilice). Radi se o svojevrsnoj prilagodbi metode Rocchio (Manning et al., 2008) za korištenje u probabilističkim modelima (Zhai, 2008).

Kod probabilističkih modela ocjenu relevantnosti možemo izraziti preko Kullback-Leiblerove (KL) udaljenosti:

$$\begin{aligned} s(q,d) &= -\operatorname{KL}(\theta_q||\theta_d) \\ &= -\sum_{\omega \in V} p(\omega|\theta_q) \log \frac{p(\omega|\theta_q)}{p(\omega|\theta_d)}, \end{aligned} \tag{3.6}$$

pri čemu je q upit, d dokument, θ jezični model, a V vokabular upita i dokumenta te vrijedi:

$$p(\omega||\theta_q) = \frac{c(\omega, q)}{|q|}, \tag{3.7}$$

pri čemu je $c(\omega, q)$ broj pojavljivanja pojma ω u upitu te $|q|$ broj pojmova u upitu.

Dodatnu informaciju o relevantnosti možemo ukomponirati u model tako da jezični model upita proširimo relevantnim dokumentima ($F = \{d_1, \ldots, d_n\}$). Bitno je napomenuti da se radi o metodi ovisnoj o upitu, odnosno da metoda ne generalizira poboljšavanje ranga na neviđene upite. Novi jezični model upita θ'_q je:

$$p(\omega|\theta'_q) = (1-\alpha)p(\omega|\theta_q) + \alpha p(\omega|\hat{\theta}_F), \tag{3.8}$$

pri čemu je $\alpha \in [0, 1]$ parametar za kontrolu jačine informacije koju primamo od relevantnih dokumenata, a $\hat{\theta}_F$ podešeni jezični model kolekcije F. Da bismo izračunali (3.8), potrebno je odrediti $\hat{\theta}_F$. Prilikom određivanja $\hat{\theta}_F$ cilj je naglasiti riječi koje čine informaciju koju opisuje q te filtrirati "pozadinske riječi" (riječi karakteristične za ostatak dokumenata) – $\hat{\theta}_F$ je jezični model blizak svim dokumentima u F, ali dalek ostatku dokumenata. Za njegovo određivanje potrebno je riješiti optimizacijski problem:

$$\hat{\theta}_F = \arg\min_{\theta} \left(\frac{1}{n} \sum_{i=1}^{n} \operatorname{KL}(\theta||\theta_i) - \lambda \operatorname{KL}(\theta||\theta_C) \right), \tag{3.9}$$

pri čemu je θ_i jezični model dokumenta $d_i \in F$, θ_C jezični model kompletne kolekcije dokumenata te $\lambda \in [0, 1)$ parametar za kontrolu udaljenosti između $\hat{\theta}_F$ i θ_C. Navedeni optimizacijski problem ima analitičko rješenje:

$$p(\omega|\hat{\theta}_F) \propto \exp\left(\frac{1}{1-\lambda} \cdot \frac{1}{n} \sum_{i=1}^{n} \log p(\omega|\theta_i) - \frac{1}{1-\lambda} \log p(w|C) \right). \tag{3.10}$$

Budući da se kod problema u ovom radu neki dokumenti više puta pojavljuju kao relevantni, izraz (3.10) proširen je prebrojavanjem tih dokumenata:

$$p(\omega|\hat{\theta}_F) \propto \exp\left(\frac{1}{1-\lambda} \cdot \frac{1}{n} \sum_{i=1}^{m} c(d_i) \cdot \log p(\omega|\theta_i) - \frac{1}{1-\lambda} \log p(w|C) \right), \tag{3.11}$$

pri čemu je $c(d_i)$ broj pojavljivanja dokumenta d_i među relevantnim dokumentima, te vrijedi:

$$n = \sum_{i=1}^{m} c(q_i). \tag{3.12}$$

Za zaglađivanje jezičnih modela koristi se vjerojatnost pojave jedne riječi u nekom tekstu (pretpostavlja se da se sve riječi jednako vjerojatne): $p_s = c_d/c_t$ (c_d je broj dokumenata, a c_t broj riječi u kolekciji). Takvo zaglađivanje navedeno je u (Zhai, 2008), no moguće je koristiti i sofisticiranije metode zaglađivanja, primjerice Good-Turingovu metodu (Good, 1953). Postupak primjene metode učenja rangiranja temeljene na KL udaljenosti:

1. izvršava se upit q koji vraća listu l;
2. uzima se k ($k = 20$) prvih rezultata osnovne liste;
3. dohvaća se prošireni jezični model za upit q, θ'_q (ako ne postoji, postupak je gotov);
4. za svaki od k rezultata (dokumenata) računa se odstupanje jezičnog modela dokumenta i θ'_q korištenjem KL udaljenosti;
5. dobiveno odstupanje dodaje se postojećoj listi l (modificiramo samo prvih k rangova):

   $$l_i = s(q, l_i(d)) \cdot p_r \cdot (k - i + 1), \tag{3.13}$$

   pri čemu je $l_i(d)$ dokument na i-tom mjestu liste, a p_r vjerojatnost relevantnosti preuzeta iz osnovnog rezultata liste:

   $$p_r = \frac{1}{\sum_{i=1}^{k} i}, \tag{3.14}$$

   time prvorangirani dokument ima osnovnu vjerojatnost $p_r \cdot k$, a k-rangirani p_r;
6. nakon podešavanja, listu l je potrebno sortirati te se može prikazati korisniku.

3.5. Implementacija

Implementacija opisanih metoda izvedena je koristeći programski jezik Java.[4] Razlog odabira Jave kao programskog jezika implementacije leži u ciljanoj integraciji programskog rješenja sa Apache Luceneom koji je napisan u Javi.

[4] `http://www.oracle.com/us/technologies/java/overview/index.html`

Programsko rješenje nalazi se u Eclipse[5] projektu "LearningToRank." Svaka implementacija nalazi se iza `hr.fer.takelab.mlr.BoostFunction` sučelja:

```
1 public interface BoostFunction {
2   void learn(QueryLogParser logParser);
3   BoostDescriptor[] boost(String query, BoostDescriptor ...
        boostedDocs);
4 }
```

Metode opisane u odjeljcima 3.2, 3.3 i 3.4 implementirane su razredima `RatioBoostFunction`, `LogBoostFunction`, odnosno `LMBoost`.

Pri razvoju korištena je IR biblioteka *Terrier* (Ounis et al., 2006). Razlog korištenja Terriera naspram Lucenea za potrebe razvoja leži u većoj ponudi IR modela. Primjer korištenja `BoostFunction` sučelja u kombinaciji sa *Terrier* bibliotekom:

```
1 public static BoostDescriptor[] query(BoostFunction f, String q,
      QueryLogParser logParser) throws IOException {
2   MultiModelRetreival mmr = new MultiModelRetreival();
3   int[] res = mmr.query(q);
4   int resNum = Math.min(res.length, MAX_RESULTS);
5   BoostDescriptor[] resDocs = new BoostDescriptor[resNum];
6
7   // linear probabilities - r[0]->max p; r[resNum]->min p; sum(p)=1
8   double p1 = BoostDescriptor.pQuant(resNum); // probability for one
        rank
9
10  for (int i = 0; i < resNum; i++) {
11    String doc = mmr.getFilename(res[i]);
12    int lastSl = doc.lastIndexOf('/');
13    int lastDot = doc.lastIndexOf('.');
14    String filename = doc.substring(lastSl+1, lastDot);
15    resDocs[i] = new BoostDescriptor(res[i],filename,(resNum-i)*p1);
16  }
17
18  f.learn(logParser);
19  BoostDescriptor[] bstDcs = f.boost(q, resDocs);
20
21  return bstDcs;
22 }
```

Razred `BoostDescriptor` ima sljedeće članove:

[5]Razvojno okruženje za Javu: `http://eclipse.org/`

```
1 public class BoostDescriptor {
2   public final int docid;
3   public final String doc;
4   public final double boost;
5 }
```

pri čemu je `docid` ID dokumenta u Terrier indeksu, `doc` naziv dokumenta te `boost` ocjena relevantnosti dokumenta.

4. Predlaganje upita

Prilikom pretraživanja, korisnik može postaviti upit koji je neadekvatan za bazu dokumenata tražilice. Neadekvatnost se manifestira pravopisnim greškama ili "lošim" izborom riječi upita – primjerice, korisnik postavi upit "*avion*," a u bazi tražilice to je prijevozno sredstvo navedeno kao "*zrakoplov*."

Cilj predlaganja upita je, na temelju korisnikovog upita, ponuditi drugi upit koji daje bolje rezultate. Za razliku od poboljšavanja rangiranja, pri predlaganju upita pokušava se uz rangiranje poboljšati preciznost i odziv.[1] Bolji upit može dohvatiti više relevantnih dokumenata i njime se može ostvariti bolje rangiranje rezultata.

Prilikom predlaganja upita bitno je da brojnost rezultata bude dovoljno velika. Stoga, prije predlaganja korisno je postaviti tražilici novi upit i vidjeti koliko rezultata on donosi. Ako novi upit vrati broj rezultata koji je za više redova veličine manji od broja rezultata početnog upita, tada je dobro taj novi upit zanemariti (odnosno, ne ponuditi prijedlog).

U nastavku slijedi opis razvijenih metoda s njihovim prednostima i nedostacima te formalna evaluacija svake metode.

4.1. Metoda temeljena na grupiranju

Metoda temeljena na grupiranju opisana je u radu (Beeferman i Berger, 2000). Ideja se temelji na pretpostavci da su upiti semantički slični ako je korisnik za njih odabrao semantički slične URL-ove. Za metodu nije potreban potpuni sadržaj dnevnika upita već samo parovi "(upit, URL)" — što povlači da metoda koristi isključivo one zapise dnevnika upita tražilice koji su rezultirali klikom.

Za izvedbu je potrebno napraviti bipartitni graf kojem su na jednoj strani skupovi upita, a na drugoj skupovi URL-ova (u prvoj iteraciji svaki skup ima točno jedan element). Inicijalne veze među čvorovima odgovaraju parovima iz dnevnika

[1] *vidi:* Dodatak: B

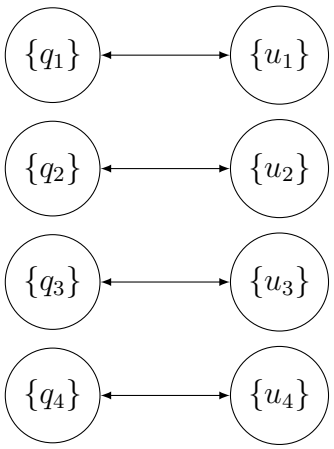
upita. Nakon toga se vrši grupiranje iterativnim spajanjem sličnih čvorova (prvo se spajaju čvorovi upita pa čvorovi URL-ova). Prilikom spajanja pamti se broj pojavljivanja pojedinog upita. Spajanje se vrši do kriterija zaustavljanja. Kao mjera sličnosti koristi se Jaccardijeva sličnost skupova (Rajaraman i Ullman, 2011):

$$\text{sim}(S_1, S_2) = \frac{S_1 \cap S_2}{S_1 \cup S_2}. \tag{4.1}$$

No, mjera sličnosti se ne primjenjuje direktno na skupove već na drugu stranu grafa. Odnosno, ako računamo sličnost između dva čvora upita (lijeva strana bipartitnog grafa) u mjeru ubacujemo njihove odgovarajuće čvorove URL-ova (desna strana grafa). Primjer računanja sličnosti dva čvora URL-ova:

$$\text{sim}(U_1, U_2) = \frac{Q_1 \cap Q_2}{Q_1 \cup Q_2}, \tag{4.2}$$

pri čemu su Q_i čvorovi upita (lijeva strana), a U_i čvorovi URL-ova (desna strana).



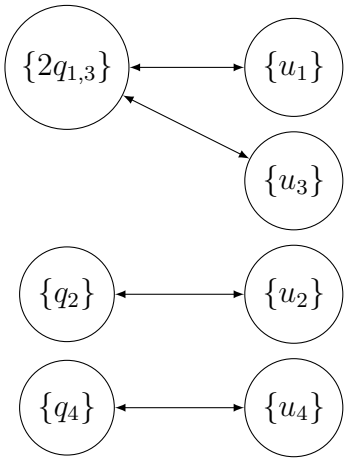
Slika 4.1: Bipartitni graf za grupiranje upita (q_i) i URL-ova (u_i), prva iteracija.

Beeferman i Berger (2000) koriste dva kriterija zaustavljanja.

$$\max_{Q_i, Q_j \in \mathcal{Q}} \text{sim}(Q_i, Q_j) = 0 \quad \text{i} \quad \max_{U_i, U_j \in \mathcal{U}} \text{sim}(U_i, U_j) = 0, \tag{4.3}$$

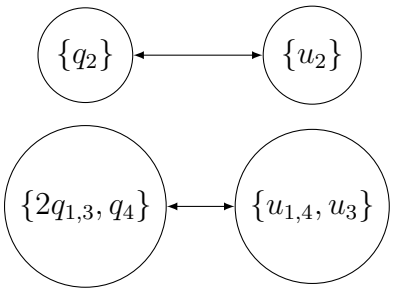
pri čemu su $\mathcal{Q}$ i $\mathcal{U}$ skupovi svih čvorova upita, odnosno URL-ova. Primijetimo da, ako se koristi ovakav kriterij zaustavljanja, dva su čvora slična ako dijele bar jedan element.

Primjer: koristimo kriterij (4.3), inicijalni graf sa slike 4.1, te vrijedi: $q_1 = q_3$ i $u_1 = u_4$. U drugoj iteraciji spajaju se čvorovi upita q_1 i q_3. Budući da se radi o istom upitu taj upit ćemo označiti sa $q_{1,3}$ te ćemo navesti njegovu brojnost



Slika 4.2: Druga iteracija grupiranja, spajanje čvorova upita.

u novom čvoru (brojnost se koristi pri odabiru reprezentativnog upita grupe). Primjer druge iteracije nalazi se na slici 4.2. U trećoj iteraciji dolazi do grupiranja čvorova URL-ova te se spajaju čvorovi u_1 i u_3 (jer dijele isti upit) te u_1 i u_4 (jer su jednaki). Time dobivamo čvor URL-ova koji sadrži URL-ove $\{u_{1,4}, u_3\}$ (kod URL-ova brojnost nam nije bitna te ju ne bilježimo). Kraj grupiranja dolazi u četvrtoj iteraciji kad se vrši zadnje spajanje čvorova upita. Rezultat grupiranja se može vidjeti na slici 4.3.



Slika 4.3: Rezultat primjera grupiranja (treća iteracija).

Kad korisnik postavi upit, prvo se traži grupa u kojoj se taj upit nalazi. Ako takva grupa postoji, tada se korisniku vraća najfrekventniji upit te grupe. Za izgrađene grupe na slici 4.3 i korisnikov upit q_4, sustav bi mu predložio upit q_1 jer se on dva puta pojavio u toj grupi (prvo kao q_1, a onda kao q_3).

Ako se kao kriterij zaustavljanja odabere (4.3), tada je moguće izvesti grupiranje direktno u tablici raspršenog adresiranja čime se sve operacije, osim operacije spajanja grupa, mogu izvesti u logaritamskom vremenu. No, takav kriterij omogućava izradu prevelikih grupa. Stoga je potrebno ručno podesiti restriktivniji kriterij, a uvođenje restriktivnijeg kriterija povlači potrebu za potpunim računanjem mjere

sličnosti skupova, što je izuzetno skupa operacija.

4.2. Metode temeljene na ispravljaču pravopisa

Ako je na raspolaganju ispravljač pravopisa, poput *Spellchecker* komponente Apache Lucenea (Hatcher et al., 2010), moguće je pomoću njega izvesti predlaganje upita na način da mu se baza riječi napuni upitima iz dnevnika upita tražilice.

Razvijene su dvije metode koje se temelje na ispravljaču pravopisa:

a) Temelj metode je pretpostavka da se dobar upit češće pojavljuje u dnevnika upita. Kao baza "riječi" ispravljača pravopisa koriste se svi upiti iz dnevnika upita tražilice koji su rezultirali odabirom dokumenta. Dodatno, u obzir se uzima njihova brojnost. Odnosno, pri predlaganju upita, među rezultatima koje ispravljač pravopisa vrati, odabire se onaj upit koji je najfrekventniji u bazi.

b) Metoda se temelji na pretpostavci da korisnik mijenja upit dok ne dođe do zadovoljavajućeg rezultata. Tu pretpostavku dodatno ograničavamo te kažemo da su n-ti i $(n+1)$-vi upit u vezi uz uvjete (svi uvjeti moraju biti zadovoljeni):

   (a) zadao ih je isti korisnik;

   (b) vremenska razlika zadavanja upita je manja ili jednaka t_{min} ($t_{min} = 120s$);

   (c) Levenshteinova udaljenost (engl. *Levenshtein distance; edit distance*) (Levenshtein, 1966) među upitima je manja od zadanog praga (prag je postavljen na 50%).

   Svaki upit za koji smatramo da je ispravak nekog drugog upita, dodajemo u bazu ispravljača pravopisa (također pamtimo njegovu frekvenciju pojavljivanja).

Navedena metoda uspijeva ponuditi poboljšani upit čak i kad se novi upit (upit koji želimo poboljšati) ne nalazi u dnevniku upita tražilice.

Problem metode je slabo prepoznavanje semantičkih veza među upitima koji nisu morfološki slični. Pozitivna strana metode jest mogućnost generiranja prijedloga za dosad neviđene upite.

O kvaliteti pristupa govori činjenica da je krajem četvrtog mjeseca 2012. godine Twitter[2] iskoristio pristup predlaganju upita za njihovu tražilicu sličan ovdje opisanoj metodi (b) (Twitter, 2012). Twitter je jedna od najvećih socijalnih mreža te broji preko 140 milijuna korisnika[3] koji izvršavaju $1,6$ milijardi upita na Twitterovoj tražilici po danu.[4] Budući da se radi o privatnom rješenju zatvorenog koda nije moguće dati detaljnu usporedbu njihove metode predlaganja upita i metode opisane u ovom odjeljku. Twitter je u javnost pustio informaciju da koriste Lucene i njegovu Spellchecker komponentu (ista ideja je predložena u ovom radu) te da kao mjeru sličnosti upita koriste modificiranu Levensteinovu udaljenost (modifikacija sličnosti zbog domene pretraživanja koja uključuje "Twitter tagove" – kratke riječi koje počinju oznakom "#"; detalji nisu poznati).

4.3. Metode temeljene na kratkoj povijesti pretraživanja

Temelj ove metode je pretpostavka da korisnik mijenja upit dok ne dobije zadovoljavajući rezultat. Upiti do zadovoljavajućeg se proglašavaju "lošim." Ako drugi korisnik zada "loš" upit, sustav mu treba ponuditi upit koji je drugog korisnika doveo do zadovoljavajućeg rezultata.

Metoda koristi potpuni zapis dnevnika upita tražilice te se sastoji od tri dijela koji se mogu varirati:

Okidač za gledanje unatrag: odlučuje treba li na trenutnom zapisu dnevnika upita stati i pogledati prethodne zapise radi dopunjavanja baze prijedloga.

Strategija za gledanje unatrag: definira koliko daleko unatrag smijemo ići da bismo dopunili bazu prijedloga.

Uvjet prihvata: odlučuje prihvaća li se povijesni zapis u bazu prijedloga.

Algoritam izgradnje baze prijedloga je sljedeći:

1. iteriraj po dnevniku upita dok zapis (e_g) ne aktivira *okidač za gledanje unatrag*;

[2] `https://twitter.com/`

[3] `http://blog.twitter.com/2012/03/twitter-turns-six.html`

[4] `http://engineering.twitter.com/2011/05/engineering-behind-twitters-new-search.html`

2. koristeći *strategiju za gledanje unatrag* iteriraj po prošlim zapisima (od novijih prema starijim);

3. svaki zapis koji zadovolji *uvjet prihvata* dodaj u listu loših upita ($\mathbf{e}_b$);

4. kad završi gledanje unatrag, dodaj u bazu prijedloga: $\mathbf{q}_b \rightarrow q_g$ (lista $\mathbf{q}_b$ je lista upita svih zapisa $\mathbf{e}_b$, a q_g je upit zapisa e_g; baza prijedloga sad za svaki loš zapis ($q \in \mathbf{q}_b$) kao prijedlog može vratiti q_g – pravi prijedlog je onaj koji se najviše puta nađe u bazi prijedloga);

5. vrati se na prvi korak.

Bitno je naglasiti da se prijedlozi promatraju isključivo među upitima jednog korisnika. Odnosno, strategija za gledanje unatrag ne smije doći do upita drugog korisnika. Na taj način osiguravamo da se traže samo "loši" upiti koje je zadnji upit "popravio."

Moguće su sljedeće varijacije glavnih dijelova algoritma:

- Okidač za gledanje unatrag:
  - gledaj unatrag za svaki zapis,
  - gledaj unatrag samo ako je korisnik odabrao neki rezultat.
- Strategija za gledanje unatrag:
  - gleda se najviše n prethodnih zapisa,
  - gledaju se svi prethodni zapisi koji nisu stariji od t sekundi u odnosu na zapis koji je pokrenuo okidač.
- Uvjet prihvata:
  - prihvati svaki zapis,
  - prihvati svaki zapis ako vrijedi:

$$\text{dist}(q_g, q_{cur}) \leq p, \tag{4.4}$$

ili (druga varijanta):

$$\text{dist}(q_g, q_{cur}) \geq p, \tag{4.5}$$

pri čemu je q_g upit zapisa koji je pokrenuo okidač, q_{cur} upit zapisa koji trenutno promatramo, $\text{dist}(q_g, q_{cur})$ njihova udaljenost, primjerice Levenshteinova udaljenost (Levenshtein, 1966), a p prag. Varijante koristimo u ovisnosti o tome želimo li prihvaćati znakovno sličnije ili

raznolikije upite. Primijetite da varijante nisu ekvivalentne u odnosu na interval praga. Ako vrijedi $p \in [0, 1]$, nekome bi se moglo učiniti da od prve varijante možemo dobiti drugu tako da prvoj varijanti postavimo prag $1 - p$. Prva varijanta za prag p vraća sve iz intervala $\text{dist}(\cdot) \in [0, p]$, dok druga za p vraća sve iz $\text{dist}(\cdot) \in [p, 1]$, odnosno radi se o različitim stranama intervala.

4.4. Implementacija

Implementacije navedenih metoda izvedene su korištenjem programskog jezika Java te se nalaze u paketu `org.apache.lucene.didyoumean` unutar Eclipse projekta `lucene_solr`. Projekt `lucene_solr` je repozitorij Apache projekata Lucenea (`http://lucene.apache.org/`) i Solra (`http://lucene.apache.org/solr/`). Metode su izvedene kao Luceneovi `contrib` moduli uz cilj jednostavne integracije sustava za predlaganje upita s Luceneom.

Implementacija metode opisane u 4.1 nalazi se u razredu `ClusteringDym`. Radi se o brzoj izvedbi bez mogućnosti promjene praga sličnosti skupova. Implementacija ovisi o Redis KVS[5] poslužitelju te Jedis klijentu[6] za pristup poslužitelju Redis iz Jave. Ako je potrebno, poslužitelj Redis je moguće izbaciti i koristiti neki drugi KVS sustav. Izgradnja grupa za predlaganje upita vrši se metodom:

```
1 void prepareDB(QueryLogParser parser);
```

a dohvaćanje prijedloga boljeg upita:

```
1 String suggest(String query);
```

Za čitanje dnevnika upita tražilice pripremljeni su razredi `IterCadialLogParser` za čitanje dnevnika upita Cadial tražilice te `IterQueryLogParser` za čitanje dnevnika upita AOL tražilice.

Također je razvijena implementacija kod koje je moguće podešavati prag sličnosti. Radi se o razredu `ClusteringBasedDym` (također koristi poslužitelj Redis). Ta implementacija je nepraktična zbog velike vremenske i prostorne složenosti obrade.

Implementacije metoda opisanih u 4.2 nalaze se u razredima `QueryLogProcessor` i `PrepareIndex`. Obje ovise o Apache Luceneu i njegovom modulu Spellchecker.

[5] Redis key-value-store, `http://redis.io/` – raspodijeljena, skalabilna tablica raspršenog adresiranja

[6] `https://github.com/xetorthio/jedis`

Inicijalizacija i predlaganje upita vrše se metodama:

```
static void processLog(QueryLogParser parser, StringDistance
    distance, float distThreshold, IndexWriter writer);
static String getSuggestion(String query, SpellChecker
    dymRecommender, IndexSearcher is);
```

odnosno:

```
static void indexIt(Directory dir, Directory spellDir, String path);
static String getSuggestion(String query, SpellChecker
    dymRecommender, IndexSearcher is);
```

Prva metoda kao ulaz prima potpuni dnevnik upita servera,[7] a druga tekstnu datoteku s parovima (n, q) pri čemu je q upit, a n broj pojava upita q u dnevniku upita tražilice (n i q su odvojeni razmakom).

Primjer korištenja `QueryLogProcessora`:

```
private Directory dir;
private Directory spellDir;
private SpellChecker dymRecommender;
private IndexSearcher is;

/**
 * Inicijalizacija komponente.
 * Mora se pozvati prije predlaganja upita!
 */
public void init() throws IOException {
  dir = new RAMDirectory();
  spellDir = new RAMDirectory();
  IndexWriter writer = new IndexWriter(dir, new IndexWriterConfig(
      Version.LUCENE_40, new StandardAnalyzer(Version.LUCENE_40)));
  dymRecommender = new SpellChecker(spellDir);

  QueryLogParser parser = new IterCadialLogParser("./cadial-all");
  QueryLogProcessor.processLog(parser, dymRecommender.
      getStringDistance(), dymRecommender.getAccuracy(), writer);
  writer.commit();

  IndexReader ir = IndexReader.open(dir);

  dymRecommender.indexDictionary(new LuceneDictionary(ir,
      QueryLogProcessor.REC_FIELD_NAME));
  is = new IndexSearcher(dir, true);
```

[7] *vidi:* Poglavlje 5

```
24
25    ir.close();
26    writer.close();
27  }
28
29  /** Predlaganje upita. */
30  public String suggest(String query) throws IOException {
31    return QueryLogProcessor.getSuggestion(query, dymRecommender, is);
32  }
33
34  /** Oslobađanje resursa. */
35  public void destroy() throws IOException {
36    is.close();
37    dir.close();
38    dymRecommender.close();
39    spellDir.close();
40  }
```

Primjer korištenja `PrepareIndex`-a:

```
1  private Directory dir;
2  private Directory spellDir;
3  private SpellChecker dymRecommender;
4  private IndexSearcher is;
5
6  /**
7   * Inicijalizacija komponente.
8   * Mora se pozvati prije predlaganja upita!
9   */
10 public void init() throws IOException {
11   dir = new RAMDirectory();
12   spellDir = new RAMDirectory();
13
14   // cut -f3,4 cadial-all | grep -v '^0' | cut -f2 | sort | uniq -c
       > cadial-hqc
15   PrepareIndex.indexIt(dir, spellDir, "./cadial-hqc");
16
17   is = new IndexSearcher(dir, true);
18   dymRecommender = new SpellChecker(spellDir);
19 }
20
21 /** Predlaganje upita. */
22 public String suggest(String query) throws IOException {
23   return PrepareIndex.getSuggestion(query, dymRecommender, is);
```

```
24 }
25
26 /** Oslobađanje resursa. */
27 public void destroy() throws IOException {
28   is.close();
29   dir.close();
30   dymRecommender.close();
31   spellDir.close();
32 }
```

Zadnja metoda, `QuerySeqProcessorDym`, također je ovisna o Redisu i Jedisu. Uz to, izuzetno je konfigurabilna te korisnik mora odrediti konfiguraciju koja najbolje odgovara njegovoj ideji poboljšanog upita. Priprema i predlaganje upita vrše se metodama:

```
1 void prepareDB(QueryLogParser parser);
2 String suggestString(String query);
```

Primjer korištenja:

```
1 private QuerySeqProcessorDym seqDym;
2 private String queryLogPath;
3
4 /** Konstruktor primjera s prikazom podešavanja parametara. */
5 public QSPDymUsageExample(String queryLogPath, LookBackStrategy
      lookBackStrategy, LookBackTrigger trigger, EntryAcceptCond
      accCond) {
6   seqDym = new QuerySeqProcessorDym(lookBackStrategy, trigger,
        accCond);
7   this.queryLogPath = queryLogPath;
8 }
9
10 /**
11  * Inicijalizacija komponente.
12  * Mora se pozvati prije predlaganja upita!
13  */
14 public void init() throws IOException {
15   QueryLogParser parser = new IterCadialLogParser(queryLogPath);
16   seqDym.prepareDB(parser);
17 }
18
19 /** Predlaganje upita. */
20 public String suggest(String query) throws IOException {
21   return seqDym.suggestString(query);
22 }
```

```
23
24 /** Oslobađanje resursa. */
25 public void destroy() throws IOException {
26   seqDym.flush();
27   seqDym.clean();
28 }
```

Konfiguracija metode sastoji se od odabira strategije za gledanje unatrag, okidača za gledanje unatrag i uvjeta prihvata. Konfiguracije koje su korištene pri evaluaciji su sljedeće:

```
1 EntryAcceptCond conds[] = {
2   null,
3   new SimilarityCond(new LevensteinDistance(), 0.3f, true),
4   new SimilarityCond(new LevensteinDistance(), 0.6f, true),
5   new SimilarityCond(new LevensteinDistance(), 0.9f, true),
6   new SimilarityCond(new LevensteinDistance(), 0.3f, false),
7   new SimilarityCond(new LevensteinDistance(), 0.6f, false),
8   new SimilarityCond(new LevensteinDistance(), 0.9f, false)
9 };
10
11 LookBackStrategy strategies[] = {
12   new TimeBasedLookBack(420000),
13   new CountBasedLookBack(5)
14 };
15
16 LookBackTrigger triggers[] = {
17   null,
18   new ClickLookBackTrigger(),
19 };
```

Konfiguracija je detaljnije opisana u odjeljku 4.3.

5. Skupovi podataka

Skupovi podataka potrebni za razvijene metode mogu se dobiti izravno iz zapisa dnevnika upita tražilice. Bitno je da tražilica bilježi sljedeće podatke:

- korisnički upit,
- identifikacijska oznaka (ID) odabranog dokumenta (ili ništa ako dokument nije odabran),
- vrijeme nastanka upita (engl. *query timestamp*),
- identifikacijska oznaka (ID) korisnika.

Zabilježeno vrijeme i ID korisnika moraju biti tek relativno točni. Odnosno, vrijeme mora nositi informaciju o poretku upita, a ID korisnika se ne mora direktno vezati uz fizičku osobu već je samo potrebna mogućnost da se za skup upita može reći da su postavljeni od istog korisnika za vrijeme trajanja jedne pretrage.

Prilikom rada sa skupovima temeljenim na dnevniku upita tražilica potrebno je biti svjestan karakterističnih problema tog tipa podataka i karakteristika skupa podataka. Tako u obzir treba uzeti sljedeće:

- zapis dnevnika upita tražilice sadrži samo implicitnu povratnu informaciju;
- metode zahtijevaju veliku količinu podataka (reda veličine 10^6), što, iako se to može činiti ostvarivim, često nije slučaj;
- parovi upita i dokumenata dosta su neravnomjerno raspoređeni, odnosno većina dokumenata uopće se ne pojavljuje u dnevniku upita;
- postoji velika količina šuma u podacima. Dio šuma tvore sami korisnici, no u slučaju web-tražilica veliki dio šuma tvore pobirači (engl. *crawlers*) drugih tražilica (u slučaju skupa podataka Cadial, radilo se o čak 50% zapisa – ti su zapisi unaprijed uklonjeni). Takve zapise je potrebno filtrirati (to je često moguće napraviti po IP adresi);
- dio metoda zahtijeva indeksirane dokumente, to često zna biti problem (zbog nedostupnosti ili količine podataka),

– neke metode zahtijevaju pristup tražilici od koje je dnevnik upita potekao, a korištenje drugih alata (primjerice biblioteke Terrier) može utjecati na točnost evaluacije (gubi se pristranost korisnika prema prvo rangiranim dokumentima).

5.1. Skupovi za učenje rangiranja s ocjenama relevantnosti

Postoji veći broj javno dostupnih skupova za učenje rangiranja s ocjenama relevantnosti. Takvi skupovi imaju preračunate značajke te nemaju tekstni zapis upita i dokumenata. No, korisni su za potrebe pristupa opisanog u 3.1. Primjeri dostupnih skupova su:

LETOR: `http://research.microsoft.com/~letor`,

YAHOO: `http://learningtorankchallenge.yahoo.com/`.

Skup LETOR kao bazu dokumenata koristi TREC 2003 i 2003 "Gov," "OHSUMED" i TREC 2007 "Gov2" korpuse. Korpusi "Gov" i "Gov2" sadrže HTML stranice sa .gov domene, a "OHSUMED" je korpus članaka iz područja medicine. Statistike pojedinih korpusa:

Gov: 1 053 110 dokumenata, 575 upita;

Gov2: 25 000 000 dokumenata, 2 500 upita;

OHSUMED: 348 566 dokumenata, 106 upita i 16 140 parova upita i dokumenata.

Točan broj parova upita i dokumenata nije poznat. Parovima upita i dokumenata je određena relevantnost u tri stupnja. Skup LETOR se sastoji od unaprijed izračunatih značajki. Liu (2011) navodi puni popis korištenih značajki. Upiti i dokumenti u tekstnom obliku nisu dostupni.

Skup YAHOO je podijeljen na dva podskupa – mali skup sa 1 266 upita te 34 815 dokumenata, te veliki skup sa 19 944 upita i 473 134 dokumenata. Parovima upit-dokument je određena relevantnost u pet stupnjeva (od nerelevantnog para do savršeno relevantnog para). Skup sadrži dokumente i upite iskazane preko 700 unaprijed izračunatih značajki (same značajke nisu poznate).

Skupovi su detaljno opisani u (Liu, 2011).

5.2. Skup podataka AOL

Skup podataka AOL je skup temeljen na dnevniku upita AOL tražilice koji je tvrtka AOL[1] objavila 2006. godine. Skup sadrži 32 milijuna zapisa dnevnika upita koji su skupljani u vremenskom periodu od tri mjeseca. Objavljivanje skupa proglašeno je napadom na privatnost te je skup maknut sa AOL-ovih stranica. Detaljniji opis skupa može se naći na: `http://www.researchpipeline.com/mediawiki/index.php?title=AOL_Search_Query_Logs`, a sam se skup može skinuti sa: `http://www.gregsadetsky.com/aol-data/` ili preuzeti iz direktorija: "`datasets/aol`."

Zbog nedostatka dokumenata na temelju kojih je skup nastao te preopćenitost domene koju skup pokriva skup AOL nije korišten za evaluaciju razvijenih metoda. No, zbog veličine, skup AOL je bio vrijedan resurs za ispitivanje metoda prilikom razvoja (radi se o ispitivanju metoda pri radu s velikim količinama podataka te neformalnoj evaluaciji).

5.3. Skupovi podataka search-lucene i search-hadoop

Search-lucene i search-hadoop skupovi podataka potječe od dviju tražilica `http://search-lucene.com/` i `http://search-hadoop.com/`, koje su u vlasništvu tvrtke Sematext International.[2] Skup broji 64000 zapisa.

5.4. Skup podataka CADIAL

Cadial dnevnik upita sadrži zapise dobivene iz tražilice `http://cadial.hidra.hr/search.php`. Radi se o tražilici baze pravnih propisa Republike Hrvatske. Skup broji 346035 zapisa, od kojih 315544 zapis ima odabran dokument. Za 27 najfrekventnijih upita skupa podataka CADIAL[3] određena je relevantnost parova upit-dokument. Ovaj skup podataka korišten je za evaluaciju svih postupaka.

Dokumenti na kojima se temelji dnevnik upita mogu se naći u filtriranom i originalnom obliku, u direktorijima na DVD mediju priloženom uz rad: "`datasets/cadial/documents/txt`" (filtrirana verzija) i "`datasets/cadial/documents/xml`."

[1] `http://www.aol.com/`

[2] `http://www.sematext.com/`

[3] *vidi:* Dodatak A

Uz to, bitna je veza između opisnika dokumenta u dnevniku upita i samog dokumenta. Ta veza je navedena u datotekama "`datasets/cadial/documents/titles`" (filtrirana verzija) i "`datasets/cadial/ documents/nn_documents.xml`" (original). Programski, tu vezu je moguće iskoristiti putem razreda `hr.fer.takelab.mlr.dataset.Cadia` (funkcija `buildDesc()` prima putanju do `titles` datoteke) iz projekta "Learning-ToRank."

Za filtriranje dokumenata korištena je sljedeća skripta:

```
for f in *.xml; do cat $f | sed -e 's/<HEAD>.*<\/HEAD>//' -e 's/<
    styles>.*<\/styles>//' -e 's/<[^>]*>/\n/g' | recode HTML_4.0 >
    ../txt/${f:0:-4}.txt; done
```

umjesto recode može se koristiti:

```
lynx -dump -stdin
```

ili

```
w3m -dump -T text/html
```

oni dodatno formatiraju ispis. Primjer neobrađenog dnevnika upita tražilice CADIAL:

```
0 [23/Aug/2011:07:21:00 +0200] 206507 GET /searchdoc.php?query=Pravilnik+o+za%C5
    %A1titi+na+radu+pri+proizvodnji+i+preradi+te%C5%A1kih+i+lakih&searchText=on&
    searchTitle=on&filteracttype=all&filtereuchapter=all&filterfields=all&
    resultlimitnum=10&resultdetails=basic&lang=hr&resultoffset=0&annotate=on&bid
    =9KeCMTMgxEP8Cglbd6dC7g%3D%3D HTTP/1.1 "Mozilla/5.0 (Windows; U; Windows NT
    6.1; en-US; rv:1.9.2.20) Gecko/20110803 Firefox/3.6.20"
0 [23/Aug/2011:07:21:17 +0200] 206857 GET /search.php?action=search&lang=hr&
    query=%3Cu%3EZakon%20o%20vlasni%C5%A1tvu%20i%20drugim%20stvarnim%20pravima%3
    C/u%3E&searchText=on&searchTitle=on&resultdetails=titles&displayOptions=on&
    filteracttype=all&filtereuchapter=all&filterfields=all&resultlimit=on&
    resultlimitnum=10 HTTP/1.1 "Mozilla/5.0 (compatible; MSIE 9.0; Windows NT
    6.1; Trident/5.0)"
1 [23/Aug/2011:07:21:17 +0200] 206857 GET /style.css HTTP/1.1 "Mozilla/5.0 (
    compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)"
1 [23/Aug/2011:07:21:18 +0200] 206857 GET /media/minus12.png HTTP/1.1 "Mozilla
    /5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)"
1 [23/Aug/2011:07:21:21 +0200] 206857 GET /tree/_lib/jquery.js HTTP/1.1 "Mozilla
    /5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)"
1 [23/Aug/2011:07:21:22 +0200] 206857 GET /favicon.ico HTTP/1.1 "Mozilla/5.0 (
    compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)"
0 [23/Aug/2011:07:23:45 +0200] 164795 GET /searchdoc.php?lang=hr&query=azbest&
    searchText=on&searchTitle=on&searchDescriptors=on&resultlimitnum=10&
    resultoffset=&action=search&filteracttype=34116&bid=D%2FSOIPQ3mGq97B5dgLdhKw
    %3D%3D HTTP/1.1 "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)"
```

Iz takvog zapisa moguće je dobiti slijed podataka s informacijama (vrijeme, korisnik, pogodak, upit, dokument). Primjerice (radi se o drugom segmentu dnevnika upita):

```
1 23/Aug/2011:08:11:33  1512  1 izvod iz uredbe o unutarnjem ustrojstvu ureda drž
      avne uprave u županijama bid=Dk5NFqJnRRz9IG7vAsKfXQ%3d%3d
2 23/Aug/2011:08:11:41  13539 1 zakon o prostornom uređenju i gradnji id_doc=
      ORsIO1e%2f0q6mTAMBBCm5Sw%3d%3d
3 23/Aug/2011:08:12:33  1512  1 izvod iz uredbe o unutarnjem ustrojstvu ureda drž
      avne uprave u županijama bid=%2bWbXKBnTzyGRbwosO4PVnA%3d%3d
4 23/Aug/2011:08:13:28  206871  1 azbest  bid=KII6dGQFf15O0vC5Sj2M7Q%3D%3D
5 23/Aug/2011:08:13:34  206826  1 pravilnik o zaštiti na radu za radne i pomočne
      prostorije  id_doc=Bj0bkzeEv7KTuA2a6jlUwQ%3d%3d
6 23/Aug/2011:08:13:39  206826  1 pravilnik o zaštiti na radu u građevinarstvu
      id_doc=VwjqvPw7jqOxEGyt%2bONnKg%3d%3d
7 23/Aug/2011:08:13:39  206826  1 pravilnik o zaštiti na radu u građevinarstvu
      bid=ddaWJSIzGW4BkOX3BHVYvw%3d%3d
```

Za čišćenje dnevnika upita (dobivanje drugog oblika iz prvog) napisane su skripte: `swap.pl, format_log.py, clean.sed, combine.sh, format.sh`. Skripte se nalaze u direktoriju "`datasets/cadial/scripts`." Cadial dnevnik upita je dobiven kao skup manjih datoteka koje je potrebno filtrirati i spojiti. Prvo se vrši filtriranje:

```
1 for f in *.LOG; do ./format.sh "$f"; done
```

time se za svaku LOG datoteku dobije filtrirana LOG.flt datoteka. Sljedeći korak je kombiniranje tih datoteka naredbom:

```
1 ./combine.sh
```

Rezultat je datoteka `combined.sort` (u tekstu se spominje kao `cadial-all`).

6. Evaluacija

Pretraživanje informacija je pretežno empirijska disciplina te se kvaliteta različitih IR metoda uspoređuju postupkom evaluacije. Postupak evaluacije zahtijeva skup informacijskih potreba i skup dokumenata s označenim ocjenama relevantnosti za pojednu informacijsku potrebu. Evaluacija se svodi na uspoređivanje liste dokumenata dobivene IR metodom koju evaluiramo i liste dokumenata složene na temelju oznaka ocjena relevantnosti. Usporedba se vrši mjerama poput Kendallove τ mjere, srednjeg recipročnog ranga (engl. *mean reciprocal rank,* MRR) i srednje prosječne preciznosti (engl. *mean average precision,* MAP). Kendallova τ mjera je opisana u odjeljku 6.1. Srednji recipročni rang je definiran izrazom:

$$MRR(\mathcal{Q}) = \frac{1}{|\mathcal{Q}|} \cdot \sum_{i=1}^{|\mathcal{Q}|} \frac{1}{r(d_{i,rel})}, \tag{6.1}$$

pri čemu je $\mathcal{Q}$ skup informacijskih potreba, a $r(d_{i,rel})$ rang prvog relevantnog dokumenta u i-oj informacijskoj potrebi. Vrijednosti srednjeg recipročnog ranga se kreću u intervalu $(0, 1]$, pri čemu vrijedi $MRR(\mathcal{Q}) = 1$ ako IR metoda za svaku informacijsku potrebu kao prvi rezultat vraća relevantan dokument. Srednja prosječna preciznost je definirana izrazom:

$$MAP(\mathcal{Q}) = \frac{1}{|\mathcal{Q}|} \cdot \sum_{i=1}^{|\mathcal{Q}|} \left(\frac{1}{m_i} \cdot \sum_{j=1}^{m_i} \text{preciznost}(R_{i,j}) \right), \tag{6.2}$$

pri čemu je $\mathcal{Q}$ skup informacijskih potreba, m_i broj relevantnih dokumenata i-te informacijske potrebe, $\text{preciznost}(R_{i,j})$ mjera preciznosti liste dokumenata dobivene IR metodom ograničene pojavom j-og relevantnog dokumenta i-te informacijske potrebe – ako IR metoda za i-tu informacijsku potrebu vrati listu dokumenata (a, b, d, h, k), pri čemu su relevantni dokumenti a i h, mjera MAP će računati preciznost na dvije liste: (a) i (a, b, d, h). Preciznost je definirana u dodatku B.

Da bi se omogućila usporedba različitih IR metoda izgrađeni su standardni skupovi za evaluaciju (Manning et al., 2008). Skup za evaluaciju IR metoda mora

sadržavati dokumente, skup informacijskih potreba izražen kao skup upita te relevantnosti dokumenata za svaki pojedini upit.

Budući da se radi o milijunskim kolekcijama dokumenata, označavanje svih dokumenata za pojedini upit je preskupo. Zato se koristi strategija "izvlačenja upita" (engl. *pooling strategy*) (Voorhees, 2002). Raspolažemo kolekcijom dokumenata i skupom upita iz skupa podataka. Algoritam odabira parova upita i dokumenata je sljedeći:

1. indeksiramo kolekciju dokumenata;
2. odabiremo m različitih IR modela;
3. za svaki upit q iz kolekcije upita izvrši sljedeće:
   (a) dohvati listu dokumenata l_i koristeći upit q i IR model M_i;
   (b) napravi uniju listi dokumenata $l_1, l_2, \ldots, l_m$;
   (c) od dobivene unije listi dokumenata odaberi prvih k dokumenata;
   (d) za svaki od odabranih k dokumenata te upit q stvori par upit-dokument;
4. svakom paru upit-dokument odredi relevantnost.

Nažalost, standardni skupovi ne postoje za svaki problem pretraživanja informacija. Standardni skup temeljen na dnevniku upita tražilice ne postoji. Zato je bilo potrebno izgraditi vlastiti skup podataka.

Kao temelj skupa podataka korištenog u ovom radu uzet je skup podataka Cadial. Za potrebe evaluacije odabrano je 27 upita iz dnevnika upita tražilice Cadial.[1] Koristeći navedene upite označeno je 2 700 parova upit-dokument. Za potrebu indeksiranje dokumenata koristila se biblioteka Terrier, a kao modeli odabrani su Terrierovi standardni modeli: "PL2," "TF_IDF," "BM25," "DLH13" (Ounis et al., 2006). Potpuni postupak označavanja je sljedeći:

a) Iz Cadial skupa izdvojeni su najfrekventniji upiti:

```
cut -f3,4 cadial-all | grep -v '^0' | cut -f2 | sort | uniq -
    c | sort -n -r
```

   Ti upiti su naknadno filtrirani radi lakšeg ocjenjivanja relevantnosti (primjerice, izbačen je upit "zakon" jer pokriva prevelik dio Cadial skupa). Odabrano je 27 najpogodnijih upita.

[1] Popis upita nalazi se u dodatku A

b) Dokumenti tražilice Cadial indeksirani su koristeći biblioteku Terrier (Ounis et al., 2006).

c) Koristeći Terrier razvijena je unija modela dohvata podataka. Implementacija se nalazi u projektu "LearningToRank" kao razred `MultiModelRetreival`. Model tvore Terrierovi modeli "PL2," "TF_IDF," "BM25" i "DLH13" spojeni primjenom *Borda count* algoritma (Elektorama, 2010). Borda count algoritam koristi se pri spajanju rezultata glasovanja na izborima. Njegov cilj je svakoj listi glasatelja pridjeliti odgovarajuću težinu (kod glasanja na izborima, težina je broj glasatelja koji tvore pojedinu listu). Borda count algoritam odabran je zbog svoje jednostavnosti, provjerene kvalitete i karakteristike da svaki model jednako vrednuje (budući da svaki model vraća jednako dugu listu dokumenata).

   Svaki pojedinačan upit izvršava se četiri puta (pojednom za svaki model). Svaka od lista ograničava se na 100 upita (radi jednostavnije evaluacije). Nakon toga se svakom dokumentu u svakoj listi dodijeli ocjena temeljena na njegovom rangu u listi. Ako dokument ima k-ti rang u listi od n dokumenata, tada on dobiva ocjenu $n - k + 1$. Od svih se lista radi unija dokumenata na način da se zbroje ocjene istog dokumenta u više lista. Dobivena lista se sortira i ponudi kao rezultat.

d) U svakoj listi se za svaki dokument označava njegova relevantnost s obzirom na upit. Za označavanje relevantnosti koriste se tri ocjene: relevantan (2), djelomično relevantan (1) te nije relevantan (0).

Popis upita korištenih za evaluaciju nalazi se u dodatku A. Ovakve ocjene relevantnosti nisu u skladu s načinom pripreme skupa za evaluaciju koji opisuje Voorhees (2002). Relevantnost se određuje na temelju upita (nepotpunog modela informacijske potrebe) dok Voorhees (2002) navodi da bi se relevantnost trebala određivati na temelju teme (informacijske potrebe). Problem nastaje ako se relevantnost parova upit-dokument određuje temeljem prespecifičnih ili preopćenitih upita. Prespecifičan upit može previše ograničiti relevantan skup dokumenata što nam ne ostavlja dovoljno prostora za gradaciju rezultata (ako postoji 15 relevantnih dokumenata te metoda koju evaluiramo relevantnima označi 10 dokumenata to govori puno više nego ako postoje dva relevantna dokumenta, a metoda koju evaluiramo relevantnima označi samo jedan dokument). S druge strane, preopćenit upit (primjerice, upit "zakon" za skup Cadial) može stvarati probleme pri ocjenjivanju

relevantnosti dokumenata.

Kvalitetan skup podataka za evaluaciju metoda učenja rangiranja te predlaganja upita trebao bi se sastojati od:

- skupa dokumenata tražilice;
- dnevnika upita tražilice;
- skupa "informacijskih potreba" iskazanih rečenicama (primjerice, "Koja su pravila postavljanja klima uređaja na stambene zgrade?");
- skupa umjetno stvorenih upita koji modeliraju svaku navedenu informacijsku potrebu;
- skupa parova umjetno stvorenih upita i dokumenata te pripadnih ocjena relevantnosti (skupovi parova upit-dokument se grade po prethodno opisanom algoritmu);
- popisa veza upita iz dnevnika tražilice (upiti koje su napravili korisnici) te umjetno stvorenih upita – radi se o semantičkoj vezi, odnosno, ako je korisnikov upit poopćenje ili specifičniji oblik umjetnog upita, tada se stvara veza (korisnikov upit, umjetni upit).

Time bi slijedili preporuke o evaluaciji iznesene u (Voorhees, 2002) te omogućili kvalitetniju evaluaciju metoda učenja rangiranja i predlaganja upita. Prednost proizlazi iz činjenice da relevantnost više ne vežemo uz nepotpuni model korisnikove informacijske potrebe već uz pravu informacijsku potrebu. Potencijalni problemi proizlaze iz moguće pojave nejednoznačnih upita (odnosno, dva korisnika mogu koristiti isti upit za različite informacijske potrebe – radi se o problemu nejednoznačnosti prirodnog jezika).

Ocjene relevantnosti nalaze se u direktoriju "`datasets/cadial/anotirani`" u obliku *csv* datoteka. Primjer učitavanja ocjena relevantnosti za upit "kazneni zakon:"

```
String[] relJug = EvalUtils.loadRelevanceJudgement("datasets/cadial/
    anotirani", "kazneni zakon");
```

Rezultat je lista dokumenata poredana po relevantnosti. Ako neki dokumenti imaju jednaku relevantnost, tada im je zadržan originalni (relativni) poredak (time se smanjuje greška kod evaluacije). Primjena mjere sličnosti između rang listi:

```
double score = EvalUtils.kendallTau(relJug, docs);
```

Lista docs sadrži poredane nazive dokumenata. Ako imamo staru listu `org` i novu `newLst`, razlika kvalitete nove i stare liste je:

```
double scoreOrg = EvalUtils.kendallTau(relJug, org);
double scoreNew = EvalUtils.kendallTau(relJug, newLst);
double r = scoreNew - scoreOrg;
```

Ako je $r > 0$, metoda je poboljšala rezultat pretraživanja.

6.1. Kendallova τ mjera

Kendallova τ mjera je besparametarski korelacijski koeficijent između dvije varijable $\mathbf{X}$ i $\mathbf{Y}$ (Nelsen, 2011). Izražava se preko suglasnih (engl. *concordant*) i nesuglasnih (engl. *discordant*) parova. Parovi (x_j, y_j) i (x_k, y_k) su suglasni ako vrijedi:

$$x_j < x_k \ \wedge \ y_j < y_k \quad \vee \quad x_j > x_k \ \wedge \ y_j > y_k, \tag{6.3}$$

odnosno, nesuglasni ako vrijedi:

$$x_j < x_k \ \wedge \ y_j > y_k \quad \vee \quad x_j > x_k \ \wedge \ y_j < y_k, \tag{6.4}$$

pri čemu je $x_i \in \mathbf{X}$, a $y_i \in \mathbf{Y}$.

Nakon određivanja suglasnosti svih $\binom{n}{2}$ parova ($n = \min(|\mathbf{X}|, |\mathbf{Y}|)$), mjera se definira sa:

$$\tau = \frac{c - d}{c + d}, \tag{6.5}$$

pri čemu je c broj suglasnih parova, a d broj nesuglasnih parova.

U slučaju rangiranja, varijable su rang liste. Za potrebe evaluacije zanima nas korelacija ručno ocjenjene rang liste $\mathbf{T}$ i nove liste $\mathbf{X}$. Ako rang liste sadrže parove (`dokument`, `rang`) i ako je primjena uvjeta (6.3) neprikladna, tada možemo napraviti novu listu $\mathbf{Y}$ koja sadržava samo rangove iz $\mathbf{T}$ ali poredane u redoslijedu koji definira $\mathbf{X}$:

$$y_i = \text{rang}_{\mathbf{T}}(\text{dokument}(x_i)), \tag{6.6}$$

pri čemu je $y_i \in \mathbf{Y}$, $x_i \in \mathbf{X}$, $\text{dokument}(x_i)$ vraća dokument iz x_i, a $\text{rang}_{\mathbf{T}}(d)$ vraća rang iz $\mathbf{T}$ za dokument d. Sada se određivanje suglasnosti može izračunati preko liste $\mathbf{Y}$:

$$c = |\{y_i \mid y_i < y_j, \ i > j, \ y \in \mathbf{Y}\}|. \tag{6.7}$$

Za nesuglasne parove vrijedi $d = \binom{n}{2} - c$.

Kendallova τ mjera prima vrijednosti iz intervala $[-1, 1]$. Vrijednost $\tau = -1$ označava da su svi parovi nesuglasni, dok vrijednost $\tau = 1$ označava da su svi parovi suglasni.

6.2. Evaluacija metoda učenja rangiranja

Evaluacija predlaganja upita je napravljena korištenjem 27 označenih upita. Postupak evaluacije je sljedeći:

a) funkcija rangiranja f naučena je na skupu podataka Cadial (pri učenju funkcije koriste se isključivo podatci iz dnevnika upita tražilice, ocjene relevantnosti se ne koriste);

b) za svaki označeni upit q, korištenjem Terriera dohvaćena je lista dokumenata, l_q;

c) lista dokumenata permutirana je funkcijom f, $l'_q = f(l_q)$,

d) između ručno podešene liste (dobivene iz ocjena relevantnosti) te lista l_q i l'_q izračunate su evaluacijske mjere: Kendallova τ mjera, recipročni rang (RR) i prosječna preciznost (AP).

Ocjena metoda može se vidjeti u tablici 6.1. Poredak upita odgovara onome u dodatku A, navedeni dodatak sadrži sve korištene upite u tekstnom obliku. Na dnu tablice nalaze se agregirane vrijednosti. U slučaju recipročnog ranga i prosječne preciznosti, agregirane vrijednosti označavaju mjeru MRR, odnosno MAP. Evaluacija je vršena nad četiri liste:

Originalna lista: lista dobivena tražilicom bez dodatnog podešavanja rangiranja.

Rangiranje omjerima: originalna lista nakon podešavanja rangiranja metodom opisanom u odjeljku 3.2.

Rangiranje log. korakom: originalna lista podešena metodom opisanom u odjeljku 3.3.

Učenje uz KL udaljenost: originalna lista na koju je primjenjena metoda opisana u odjeljku 3.4.

Rezultati pokazuju blago poboljšanje rangiranja pri korištenju metoda učenja rangiranja opisanih u odjeljcima 3.2 i 3.3 te znatno pogoršanje rangiranja pri korištenju metode učenja rangiranja opisane u odjeljku 3.4. Zbog niske kvalitete informacija nad kojima metode uče, nije realno za očekivati znatno poboljšanje rangiranja.

Evaluacija je pokazala da jednostavnije metode (podešavanje rangiranja omjerima ili logaritamskim korakom) daju bolje rezultate od složenije metode (učenje

relevantnosti temeljeno na KL udaljenosti). Možemo pretpostaviti dva razloga nadmoći jednostavnijih metoda nad složenijom:

a) implicitna povratna informacija sadrži znatnu količinu šuma i slabu informaciju o relevantnosti;

b) dokumenti skupa Cadial pisani su "pravnim jezikom," odnosno svi su slično modelirani te sadrže veliki broj jednakih pojmova.

Opisane jednostavne metode su stabilnije (imaju manju varijancu) te se bolje nose sa šumom u podatcima. Navedena karakteristika skupa Cadial stvara problem metodi učenja relevantnosti na temelju KL udaljenosti jer ona pokušava modelirati "pozadinu" kolekcije dokumenata da bi istaknula relevantne dokumente. Kod skupa Cadial, dokumenti su relativno homogeni te je teško napraviti navedeno isticanje pojedinih dokumenata. Navedeni razlozi su pretpostavke koje se mogu provjeriti evaluacijom metoda nad drugim skupovima podataka.

Tablica 6.1: Evaluacija metoda učenja rangiranja.

	Originalna lista			Rangiranje omjerima			Rangiranje log. korakom			Učenje uz KL udaljenost		
Upit	τ	RR	AP	τ	RR	AP	τ	RR	AP	τ	RR	AP
q_1	$-0,55$	$0,11$	$0,11$	$-0,55$	$0,11$	$0,11$	$-0,55$	$0,11$	$0,11$	$-0,55$	$0,10$	$0,10$
q_2	$0,89$	$1,00$	$1,00$	$0,89$	$1,00$	$1,00$	$0,89$	$1,00$	$1,00$	$0,43$	$0,33$	$0,33$
q_3	$1,00$	$1,00$	$1,00$	$1,00$	$1,00$	$1,00$	$1,00$	$1,00$	$1,00$	$0,96$	$1,00$	$1,00$
q_4	$0,54$	$0,50$	$0,50$	$0,52$	$0,33$	$0,33$	$0,54$	$0,50$	$0,50$	$0,39$	$0,17$	$0,17$
q_5	$0,80$	$1,00$	$1,00$	$0,80$	$1,00$	$1,00$	$0,77$	$1,00$	$1,00$	$0,69$	$1,00$	$1,00$
q_6	$0,66$	$1,00$	$1,00$	$0,64$	$0,33$	$0,33$	$0,66$	$1,00$	$1,00$	$0,66$	$1,00$	$1,00$
q_7	$0,97$	$1,00$	$1,00$	$0,97$	$1,00$	$1,00$	$0,97$	$1,00$	$1,00$	$0,47$	$0,05$	$0,05$
q_8	$0,84$	$1,00$	$1,00$	$0,84$	$1,00$	$1,00$	$0,84$	$1,00$	$1,00$	$0,84$	$1,00$	$1,00$
q_9	$0,29$	$1,00$	$1,00$	$0,29$	$1,00$	$1,00$	$0,29$	$1,00$	$1,00$	$0,13$	$1,00$	$1,00$
q_{10}	$0,99$	$1,00$	$1,00$	$0,99$	$1,00$	$1,00$	$0,99$	$1,00$	$1,00$	$0,83$	$1,00$	$1,00$
q_{11}	$0,83$	$0,50$	$0,50$	$0,84$	$1,00$	$1,00$	$0,84$	$1,00$	$1,00$	$0,78$	$0,50$	$0,50$
q_{12}	$0,68$	$1,00$	$1,00$	$0,68$	$1,00$	$1,00$	$0,67$	$1,00$	$1,00$	$0,33$	$0,05$	$0,05$
q_{13}	$0,96$	$1,00$	$1,00$	$0,97$	$1,00$	$1,00$	$0,96$	$1,00$	$1,00$	$0,96$	$1,00$	$1,00$
q_{14}	$0,85$	$1,00$	$1,00$	$0,85$	$1,00$	$1,00$	$0,79$	$1,00$	$1,00$	$0,62$	$0,50$	$0,50$
q_{15}	$-0,53$	$0,05$	$0,05$	$-0,53$	$0,05$	$0,05$	$-0,53$	$0,05$	$0,05$	$-0,19$	$0,05$	$0,05$
q_{16}	$0,40$	$1,00$	$1,00$	$0,39$	$1,00$	$1,00$	$0,42$	$1,00$	$1,00$	$0,42$	$1,00$	$1,00$
q_{17}	$0,65$	$1,00$	$1,00$	$0,64$	$1,00$	$1,00$	$0,64$	$1,00$	$1,00$	$0,42$	$1,00$	$1,00$
q_{18}	$0,26$	$0,17$	$0,17$	$0,26$	$0,17$	$0,17$	$0,27$	$0,17$	$0,17$	$0,12$	$0,20$	$0,20$
q_{19}	$0,66$	$0,25$	$0,25$	$0,78$	$1,00$	$1,00$	$0,71$	$0,50$	$0,50$	$0,61$	$0,25$	$0,25$
q_{20}	$0,17$	$1,00$	$1,00$	$0,28$	$1,00$	$1,00$	$0,19$	$1,00$	$1,00$	$0,16$	$1,00$	$1,00$
q_{21}	$0,55$	$1,00$	$1,00$	$0,61$	$1,00$	$1,00$	$0,55$	$1,00$	$1,00$	$0,56$	$1,00$	$1,00$
q_{22}	$0,76$	$1,00$	$1,00$	$0,72$	$0,50$	$0,50$	$0,77$	$1,00$	$1,00$	$0,55$	$1,00$	$1,00$
q_{23}	$0,76$	$0,50$	$0,50$	$0,82$	$0,50$	$0,50$	$0,78$	$0,50$	$0,50$	$0,41$	$0,09$	$0,09$
q_{24}	$0,21$	$0,50$	$0,50$	$0,32$	$1,00$	$1,00$	$0,24$	$1,00$	$1,00$	$0,28$	$1,00$	$1,00$
q_{25}	$0,27$	$1,00$	$1,00$	$0,42$	$1,00$	$1,00$	$0,32$	$1,00$	$1,00$	$0,26$	$1,00$	$1,00$
q_{26}	$0,32$	$1,00$	$1,00$	$0,32$	$1,00$	$1,00$	$0,32$	$1,00$	$1,00$	$-0,27$	$0,12$	$0,12$
q_{27}	$0,84$	$0,06$	$0,06$	$0,84$	$0,06$	$0,06$	$0,84$	$0,06$	$0,06$	$0,61$	$0,05$	$0,05$
Prosjek	$0,56$	$0,76$	$0,76$	$\mathbf{0,58}$	$0,78$	$0,78$	$0,56$	$\mathbf{0,81}$	$\mathbf{0,81}$	$0,43$	$0,61$	$0,61$

6.3. Evaluacija metoda predlaganja upita

Evaluacija predlaganja upita napravljena je korištenjem 27 označenih upita. Postupak je sljedeći:

a) za svaki je upit dohvaćen prijedlog $q_s = \text{suggest}(q)$;

b) za upit q dohvaćene su ocjene relevantnosti (upit q_s nema ocjene relevantnosti);

c) korištenjem Terriera dohvaćena je lista rezultata za q te za q_s (koriste se isti modeli kao kod izgradnje parova upit-dokument s ocjenama relevantnosti);

d) izračunata je Kendallova τ mjera između ručno podešene liste (dobivene iz ocjena relevantnosti) te liste za q, odnosno za q_s;

e) uspješnost predlaganja upita izračunata je kao razlika dobivenih dviju vrijednosti Kendallove τ mjere.

Kako ocjena relevantnosti za q_s ne mora postojati, kao savršen rezultat uzima se lista dobivena iz ocjena relevantnosti za q. Postupak uvodi grešku u evaluaciju jer ni q ni q_s ne moraju dobro opisivati korisnikovu informacijsku potrebu.

Dodatan problem je odabir upita za evaluaciju. Odabrani su najfrekventniji upiti što ih čini dobrim kandidatima za predlaganje (za druge upite ne postoje ocjene relevantnosti).

Rezultati evaluacije prikazani su u tablicama 6.2, 6.3, 6.4, 6.5 i 6.6. Svaka tablica sadrži originalni upit, prijedlog "poboljšanog" upita, mjeru kvalitete originalnog upita, mjeru kvalitete "poboljšanog" upita te razliku (poboljšani minus originalni). Neke metode za neke upite nisu dale prijedlog upita ili je prijedlog jednak originalnom upitu. Ti zapisi su izostavljeni.

Kod metode temeljene na kratkoj povijesti pretraživanja prikazane su samo tri kombinacije. Ostale se mogu smatrati lošijima. Iz evaluacije je izostavljena metoda predlaganja upita temeljena na grupiranju. Navedena metoda je za svaki postavljeni upit vraćala prijedlog "zakon o posebnim porezima na osobne automobile, ostala motorna vozila, plovila i zrakoplove." Radi se o problemu prevelikih grupa zbog preniskog praga. Uz pomicanje praga i korištenje većeg skupa podataka, navedena metoda mogla bi dati bolje rezultate. Zaključak slijedi iz eksperimentiranja s metodom nad skupom AOL, uz ručno razbijanje velikih grupa. Iz rezultata evaluacije vidljivo je da niti jedna metoda ne popravlja rezultate pretraživanja.

Tablica 6.2: Evaluacija predlaganja upita metode temeljene na ispravljaču pravopisa (a).

Originalni upit (q)	Predloženi upit (q_s)	$\tau(q)$	$\tau(q_s)$	$\tau(q_s) - \tau(q)$
odluka o proglašenju crne mlake	odlika o proglašenju bijeli potoci_kamensko	$0,99$	$-0,96$	$-1,95$
pravilnik o izradi procjene	pravilniki o izradu procjene	$0,92$	$0,21$	$-0,71$
pravilnik o kartografskim znakovima	pravilnik o kartogramskim prikazima	$0,87$	$-0,36$	$-1,23$
pravilnik o mjerama zaštite odelementarnih nepogoda	pravilnik o mjerama zaštite od elementarnih nepogoda	$0,96$	$0,32$	$-0,64$
pravilnik o mjerama zaštite od požara pri izvođenju radova zavarivanja	pravilnik o mjerama zaštite od požara pri izvođenju radova zavarivanja, rezanja	$0,99$	$0,46$	$-0,53$
pravilnik o tehničkim mjerama i uvjetima za izgradnju prostora i uređeja za prikupljanje i odnošenje otpadnog materijala	pravilnik o tehničkim mjerama i uvjetima za izgradnju prostora i uređaja za prikupljanje i odnošenje otpadnog materijala iz stambenih zgrada	$0,90$	$-0,26$	$-1,16$
viktor lenac	viktor+lenac	$0,85$	$0,73$	$-0,13$
zakon o državnim službenicima	zakonu o državnim službenicima	$0,77$	$-1,00$	$-1,77$
zakon o gradnji	zakon o gradnji 1992	$0,80$	$-0,58$	$-1,38$
zakon o hrani	zakon o obrani	$0,65$	$-1,00$	$-1,65$
zakon o izmjenama i dopunama kaznenog zakona	zakon o izmjenama i dopunama ovršnog zakona	$0,91$	$0,86$	$-0,05$
zakon o kaznenom postupku	zakono o kaznenom postupku	$0,76$	$0,42$	$-0,34$
zakon o kemikalijama	zakon	$0,97$	$0,51$	$-0,46$
zakon o naseljima	zakon o nasljeđivanju	$0,99$	$-0,02$	$-1,01$
zakon o osnovama sigurnosti transporta naftovodima	zakon o osnovama sigurnosti transporta naftovodima i plinovodima	$0,98$	$0,82$	$-0,15$
zakon o otpadu	zakona o otpadu	$0,70$	$0,69$	$-0,01$
zakon o prostornom uređenju i gradnji	zakona o prostornom uređenju i gradnji	$0,88$	$0,66$	$-0,22$
Prosjek		0.88	0.09	−0.79

Tablica 6.3: Evaluacija predlaganja upita metode temeljene na ispravljaču pravopisa (b).

Originalni upit (q)	Predloženi upit (q_s)	$\tau(q)$	$\tau(q_s)$	$\tau(q_s) - \tau(q)$
pravilnik o izradi procjene	pravilnik o kvaliteti stočne hrane	$0,92$	$-1,00$	$-1,92$
pravilnik o tehničkim normativima i uvjetima za projektiranje i gradnju tunela na cestama	• pravilnik o tehničkim normativima za ventilacijske i klimatizacijske sisteme	$0,93$	$-0,76$	$-1,68$
pravilnik o tehničkim normativima za projektiranje, gradnju, pogon i održavanje plinskih kotlovnica	pravilnik o tehničkim normativima za zaštitu skladišta od požara i eksplozija	$1,00$	$-0,75$	$-1,75$
pravilnik o tehničkim normativima za projektiranje i izvođenje završnih radova u građevinarstvu	pravilnik o tehničkim normativima za nosive čelične konstrukcije	$0,98$	$-0,20$	$-1,19$
pravilnik o tehničkim propisima za pregled i ispitivanje nosećih čeličnih konstrukcija	pravilnik o tehničkim normativima za nosive čelične konstrukcije	$0,86$	$-0,50$	$-1,36$
prvilnik o tehničkim normativima za zaštitu visokih objekata	• pravilnik o tehničkim normativima za zaštitu visokih objekata od požara	$0,95$	$0,95$	$0,00$
zakon o gradnji	zakon o prometu	$0,80$	$-0,90$	$-1,69$
zakon o hrani	zakon o vodama	$0,65$	$-0,91$	$-1,56$
zakon o javnoj nabavi	zakon o vodama	$0,81$	$-0,81$	$-1,62$
zakon o kemikalijama	uvod o kemikalijama	$0,97$	$0,23$	$-0,73$
zakon o naseljima	zakon o vodama	$0,99$	$-0,69$	$-1,68$
zakon o otpadu	zakon o vodama	$0,70$	$-0,71$	$-1,41$
zakon o zaštiti od požara	zakon o zaštiti okoliša	$0,99$	$0,99$	$0,00$
Prosjek		0.89	-0.39	-1.28

Tablica 6.4: Evaluacija predlaganja upita metode temeljene na kratkoj povijesti (gledanje unatrag na klik, 5 prethodnih upita, prihvaća se uz $\text{dist}(\cdot) \geq 0.6$).

Originalni upit (q)	Predloženi upit (q_s)	$\tau(q)$	$\tau(q_s)$	$\tau(q_s) - \tau(q)$
kazneni zakon	zakon o međunarodnj pravnoj pomoći u kaznenim stvarima	$0,62$	$-0,43$	$-1,05$
pravilnik o izradi procjene	osporavanje punomoći	$0,64$	$-1,00$	$-1,64$
pravilnik o kartografskim znakovima	državni pedagoški standard srednjoškolskog sustava odgoja i obrazovanja	$0,93$	$-0,97$	$-1,91$
pravilnik o mjerama zaštite odelementarnih nepogoda	pravilnik 400 kv	$0,96$	$-0,99$	$-1,95$
pravilnik o mjerama zaštite od požara pri izvođenju radova zavarivanja	prekršajni zakoni	$0,99$	$-0,97$	$-1,97$
pravilnik o tehničkim normativima i uvjetima za projektiranje i gradnju tunela na cestama	pravilnik o tehničkim normativima za izgradnju nadzemnih elektroenergetskih vodova nazivnog napona	$0,93$	$-0,81$	$-1,74$
pravilnik o tehničkim normativima za projektiranje, gradnju, pogon i održavanje plinskih kotlovnica	pri izgradnji i rekonstrukciji mlinova	$1,00$	$-0,97$	$-1,97$
pravilnik o tehničkim normativima za projektiranje i izvođenje završnih radova u građevinarstvu	tehnički propisi o kvaliteti zavarenih spojeva za nosive čelične konstrukcije	$0,98$	$-0,60$	$-1,58$
pravilnik o tehničkim propisima za pregled i ispitivanje nosećih čeličnih konstrukcija	upravljanje i raspolaganje+ nekretninama+	$0,86$	$-1,00$	$-1,86$
prvilnik o tehničkim normativima za zaštitu visokih objekata	nakladnička djel	$1,00$	$-1,00$	$-2,00$
viktor lenac	pravilnik o parcelacijskim i drugim elaboratima	$0,85$	$-1,00$	$-1,85$
zakon o državnim službenicima	uredba o načinima i uvjetima napredovanja	$0,77$	$-0,99$	$-1,76$
zakon o izmjenama i dopunama kaznenog zakona	za toksikologiju	$0,80$	$-1,00$	$-1,80$
zakon o javnoj nabavi	uredba o postupku javne nabave male vrijednosi	$0,81$	$0,08$	$-0,73$
zakon o kaznenom postupku	pravilnik o standardima smještaja i prehrane zatvorenika	$0,76$	$-0,91$	$-1,66$
zakon o kemikalijama	7647	$0,38$	$-1,00$	$-1,38$
zakon o naseljima	pravilnik o evidentiranju prostornih jedinica	$0,99$	$-0,78$	$-1,77$
zakon o osnovama sigurnosti transporta naftovodima	zakon o izmjenama i dopunama zakona o zaštiti i spašavanju	$0,98$	$-0,96$	$-1,94$
zakon o otpadu	zakon o subvencioniranju i državnom jamstvu	$0,70$	$-0,73$	$-1,43$
zakon o prostornom uređenju i gradnji	poslovi na željeznici posebni uvjeti	$0,88$	$-1,00$	$-1,87$
zakon o sigurnosti prometa na cestama	gospodarska vozila	$0,84$	$-1,00$	$-1,84$
zakon o zaštiti od požara	zakon o odvjetnicima	$0,99$	$0,18$	$-0,81$
Prosjek		0.85	−0.81	−1.66

Tablica 6.5: Evaluacija predlaganja upita metode temeljene na kratkoj povijesti (gledanje unatrag na klik, najveća vremenska razlika među upitima je $420s$, prihvaća se uz $\text{dist}(\cdot) \geq 0.3$).

Originalni upit (q)	Predloženi upit (q_s)	$\tau(q)$	$\tau(q_s)$	$\tau(q_s) - \tau(q)$
obiteljski zakon	zakon o rješavanju sukoba zakona s propisima drugih zemalja u određenim odnosima	$0,94$	$-0,75$	$-1,69$
pravilnik o kartografskim znakovima	rješavnju sukoba	$0,93$	$-1,00$	$-1,93$
pravilnik o tehničkim normativima i uvjetima za projektiranje i gradnju tunela na cestama	izjava kojom se operator očituje da će aerodrom u svako vrijeme kada je otvoren za uzlijetanje i slijetanje zrakoplova biti dostupan svim osobama pod jednakim uvjetima i pravima	$0,93$	$-0,97$	$-1,90$
pravilnik o tehničkim normativima za projektiranje i izvođenje završnih radova u građevinarstvu	uredba o postupcima službenog obavješćivanja u području norma, tehničkih propisa, te propisa o uslugama informacijskog društva	$0,98$	$-0,95$	$-1,94$
pravilnik o tehničkim propisima za pregled i ispitivanje nosećih čeličnih konstrukcija	stublina	$0,00$	$-1,00$	$-1,00$
prvilnik o tehničkim normativima za zaštitu visokih objekata	tumač znakova u geodetskim elaboratima	$0,95$	$-0,97$	$-1,92$
zakon o gradnji	pravilnik o projektima potrebnim za osiguranje pristupačnosti građevina osobama s invaliditetom i smanjene pokretljivosti	$0,80$	$-0,98$	$-1,77$
zakon o hrani	pravilnik o uvjetima glede stručnih djelatnika, prostorija i opreme koje moraju ispunjavati zdravstvene i druge pravne osoe za obavljanje analiza i superanaliza namirnica	$0,65$	$-0,90$	$-1,55$
zakon o izmjenama i dopunama kaznenog zakona	pravilnik o prehrambenim aditivima	$0,91$	$-1,00$	$-1,91$
zakon o javnoj nabavi	uredba o nabavi roba i usluga male vrijednosti	$0,81$	$-0,26$	$-1,07$
zakon o kaznenom postupku	pravilnik o standardima smještaja i prehrane zatvorenika	$0,76$	$-0,91$	$-1,66$
zakon o kemikalijama	zakon o registraciji i odobravanju objekta u kojima posluju subjekti u poslovanju hranom za životinje	$0,97$	$-0,97$	$-1,94$
zakon o naseljima	pravilnik o evidentiranju prostornih jedinica	$0,99$	$-0,78$	$-1,77$
zakon o otpadu	zakon o nacionalnom parku i spomen području brijuni	$0,70$	$-0,96$	$-1,66$
zakon o sigurnosti prometa na cestama	pravilnik o izvršavanju zaštitnih mjera zabrane upravljanja motornim vozilima	$0,84$	$-0,73$	$-1,57$
Prosjek		0.81	-0.88	-1.69

Tablica 6.6: Evaluacija predlaganja upita metode temeljene na kratkoj povijesti (uvijek se gleda unatrag, 5 prethodnih upita, uvijek se prihvaća).

Originalni upit (q)	Predloženi upit (q_s)	$\tau(q)$	$\tau(q_s)$	$\tau(q_s) - \tau(q)$
kazneni zakon	zakon o međunarodnj pravnoj pomoći u kaznenim stvarima	$0,62$	$-0,43$	$-1,05$
pravilnik o izradi procjene	osporavanje punomoći	$0,64$	$-1,00$	$-1,64$
pravilnik o kartografskim znakovima	državni pedagoški standard srednjoškolskog sustava odgoja i obrazovanja	$0,93$	$-0,97$	$-1,91$
pravilnik o mjerama zaštite odelementarnih nepogoda	pravilnik 400 kv	$0,96$	$-0,99$	$-1,95$
pravilnik o mjerama zaštite od požara pri izvođenju radova zavarivanja	prekršajni zakoni	$0,99$	$-0,97$	$-1,97$
pravilnik o tehničkim normativima za projektiranje, gradnju, pogon i održavanje plinskih kotlovnica	pri izgradnji i rekonstrukciji mlinova	$1,00$	$-0,97$	$-1,97$
pravilnik o tehničkim normativima za projektiranje i izvođenje završnih radova u građevinarstvu	tehnički propisi o kvaliteti zavarenih spojeva za nosive čelične konstrukcije	$0,98$	$-0,60$	$-1,58$
pravilnik o tehničkim propisima za pregled i ispitivanje nosećih čeličnih konstrukcija	pravilnik o zaštiti na radu za radne i pomočna prostore	$0,86$	$-0,96$	$-1,83$
prvilnik o tehničkim normativima za zaštitu visokih objekata	nakladnička djel	$1,00$	$-1,00$	$-2,00$
zakon o državnim službenicima	uredba o načinima i uvjetima napredovanja	$0,77$	$-0,99$	$-1,76$
zakon o gradnji	pravilnik o projektima potrebnim za osiguranje pristupačnosti građevina osobama s invaliditetom i smanjene pokretljivosti	$0,80$	$-0,98$	$-1,77$
zakon o hrani	voda za piće	$0,65$	$-1,00$	$-1,65$
zakon o izmjenama i dopunama kaznenog zakona	za toksikologiju	$0,80$	$-1,00$	$-1,80$
zakon o javnoj nabavi	uredba o postupku javne nabave male vrijednosi	$0,81$	$0,08$	$-0,73$
zakon o kaznenom postupku	pravilnik o standardima smještaja i prehrane zatvorenika	$0,76$	$-0,91$	$-1,66$
zakon o kemikalijama	7647	$0,38$	$-1,00$	$-1,38$
zakon o naseljima	pravilnik o evidentiranju prostornih jedinica	$0,99$	$-0,78$	$-1,77$
zakon o osnovama sigurnosti transporta naftovodima	zakon o izmjenama i dopunama zakona o zaštiti i spašavanju	$0,98$	$-0,96$	$-1,94$
zakon o otpadu	zakon o subvencioniranju i državnom jamstvu	$0,70$	$-0,73$	$-1,43$
zakon o prostornom uređenju i gradnji	poslovi na željeznici posebni uvjeti	$0,88$	$-1,00$	$-1,87$
zakon o sigurnosti prometa na cestama	gospodarska vozila	$0,84$	$-1,00$	$-1,84$
zakon o zaštiti od požara	zakon o odvjetnicima	$0,99$	$0,18$	$-0,81$
Prosjek		0.83	−0.82	−1.65

7. Zaključak

Ovim radom istražena je mogućnost korištenja zapisa dnevnika upita tražilice u svrhu predlaganja upita i učenja rangiranja. Dnevnik upita tražilice je jeftin resurs te je ideja o njegovom iskorištavanju za poboljšavanje pretraživanja vrlo privlačna. U radu su opisane tri metode predlaganja upita i tri metode učenja rangiranja.

U radu nije ostvaren značajan napredak u iskorištavanju loga tražilica za poboljšanje pretraživanja. Rad navodi veći broj problema koji se susreću pri pokušajima iskorištavanja dnevnika upita tražilica. Većina tih problema tiče se samih podataka, odnosno njihovog nedostatka, rijetkosti zapisa i količine šuma.

U daljnjem radu potrebno je nabaviti veći skup podataka te osigurati pristup tražilici od koje su potekli ti podatci. Potrebno je razviti bolju metodu evaluacije metoda za predlaganje upita te napraviti kvalitetniji skup dokumenata s ocjenama relevantnosti po uputama iz (Voorhees, 2002). Za metode poboljšavanja rangiranja potrebno je proučiti latentne veze između dokumenata te ukloniti ovisnost o upitu.

Literatura

D. Beeferman i A. Berger. Agglomerative clustering of a search engine query log. U *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, stranice 407–416. ACM, 2000.

Andrzej Białecki. Implementing click-through relevance ranking in solr and lucidworks enterprise. `http://www.lucidimagination.com/sites/default/files/Bialecki_Andrzej_Click_through_relevance_ranking_in_Solr_LucidWorks_Enterprise.pdf`, 2011.

B. Cao, D. Shen, K. Wang, i Q. Yang. Clickthrough log analysis by collaborative ranking. *Proc. of AAAI'10*, 2010.

S.F. Chen i J. Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–393, 1999.

Z. Dou, R. Song, X. Yuan, i J.R. Wen. Are click-through data adequate for learning web search rankings? U *Proceeding of the 17th ACM conference on Information and knowledge management*, stranice 73–82. ACM, 2008.

Elektorama. Elektorama: Borda count. `http://wiki.electorama.com/wiki/Borda_count`, 2010. Pristupljeno: 5. ožujka 2012.

I.J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3-4):237–264, 1953.

Erik Hatcher, Otis Gospodnetić, i Mike McCandless. *Lucene in Action*. Manning, 2. izdanje, 2010. ISBN 9781933988177.

S. Ji, K. Zhou, C. Liao, Z. Zheng, G.R. Xue, O. Chapelle, G. Sun, i H. Zha. Global ranking by exploiting user clicks. U *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, stranice 35–42. ACM, 2009.

T. Joachims. Optimizing search engines using clickthrough data. U *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, stranice 133–142. ACM, 2002.

T. Joachims, L. Granka, B. Pan, H. Hembrooke, i G. Gay. Accurately interpreting clickthrough data as implicit feedback. U *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, stranice 154–161. ACM, 2005.

V.I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. U *Soviet physics doklady*, svezak 10, stranice 707–710, 1966.

Hang Li. Learning to rank for information retrieval and natural language processing. *Synthesis Lectures on Human Language Technologies*, 2011.

J. Lin i C. Dyer. *Data-intensive text processing with MapReduce*. Morgan & Claypool Publishers, 2010.

Tie-Yan Liu. *Learning to rank for information retrieval*. Springer-Verlag New York Inc, 2011.

C.D. Manning, P. Raghavan, i H. Schutze. *Introduction to information retrieval*, svezak 1. Cambridge University Press Cambridge, 2008.

J.H. McDonald. Handbook of biological statistics (2nd ed.): Spearman rank correlation. `http://udel.edu/~mcdonald/statspearman.html`, 2009. Pristupljeno: 18. svibnja 2012.

R.B. Nelsen. Encyclopedia of mathematics: Kendall tau metric. `http://www.encyclopediaofmath.org/index.php?title=Kendall_tau_metric&oldid=12869`, 2011. Pristupljeno: 20. svibnja 2012.

I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald, i C. Lioma. Terrier: A High Performance and Scalable Information Retrieval Platform. U *Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006)*, 2006.

Jay M. Ponte i W. Bruce Croft. A language modeling approach to information retrieval. U *Proceedings of the 21st annual international ACM SIGIR conference*

on Research and development in information retrieval, SIGIR '98, stranice 275–281, New York, NY, USA, 1998. ACM. ISBN 1-58113-015-5. doi: 10.1145/290941.291008. URL http://doi.acm.org/10.1145/290941.291008.

F. Radlinski i T. Joachims. Active exploration for learning rankings from clickthrough data. U *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, stranice 570–579. ACM, 2007.

A. Rajaraman i J.D. Ullman. *Mining of massive datasets*. Cambridge University Press, 2011.

G. Salton, A. Wong, i C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, Studeni 1975. ISSN 0001-0782. doi: 10.1145/361219.361220. URL http://doi.acm.org/10.1145/361219.361220.

A. Singla i R.W. White. Sampling high-quality clicks from noisy click data. U *Proceedings of the 19th international conference on World wide web*, stranice 1187–1188. ACM, 2010.

Twitter. Related queries and spelling corrections in search. http://engineering.twitter.com/2012/05/related-queries-and-spelling.html, 2012. Pristupljeno: 19. lipnja 2012.

E. Voorhees. The philosophy of information retrieval evaluation. U *Evaluation of cross-language information retrieval systems*, stranice 143–170. Springer, 2002.

C. Zhai i J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems (TOIS)*, 22(2):179–214, 2004.

C.X. Zhai. Statistical language models for information retrieval a critical review. *Foundations and Trends in Information Retrieval*, 2(3):137–213, 2008.

Dodatak A
Upiti za evaluaciju sa oznakama relevantnosti

Za potrebe evaluacije ručno je označeno 27 upita:

1. kazneni zakon
2. obiteljski zakon
3. odluka o proglašenju crne mlake
4. pravilnik o izradi procjene
5. pravilnik o kartografskim znakovima
6. pravilnik o mjerama zaštite odelementarnih nepogoda
7. pravilnik o mjerama zaštite od požara pri izvođenju radova zavarivanja
8. pravilnik o tehničkim mjerama i uvjetima za izgradnju prostora i uređeja za prikupljanje i odnošenje otpadnog materijala
9. pravilnik o tehničkim normativima i uvjetima za projektiranje i gradnju tunela na cestama
10. pravilnik o tehničkim normativima za projektiranje, gradnju, pogon i održavanje plinskih kotlovnica
11. pravilnik o tehničkim normativima za projektiranje i izvođenje završnih radova u građevinarstvu
12. pravilnik o tehničkim propisima za pregled i ispitivanje nosećih čeličnih konstrukcija

13. prvilnik o tehničkim normativima za zaštitu visokih objekata
14. viktor lenac
15. zakon o državnim službenicima
16. zakon o gradnji
17. zakon o hrani
18. zakon o izmjenama i dopunama kaznenog zakona
19. zakon o javnoj nabavi
20. zakon o kaznenom postupku
21. zakon o kemikalijama
22. zakon o naseljima
23. zakon o osnovama sigurnosti transporta naftovodima
24. zakon o otpadu
25. zakon o prostornom uređenju i gradnji
26. zakon o sigurnosti prometa na cestama
27. zakon o zaštiti od požara

Korištene su tri oznake relevantnosti:

2: relevantno,

1: djelomično relevantno,

0: nerelevantno.

Označavanje je vršeno na temelju upita, a ne općenite informacijske potrebe. Označeni skup se nalazi u direktoriju "*datasets/cadial/anotirani*" te sadrži 27 *csv* tekstnih datoteka u formatu:

`rang: <tab> relevantnost <tab> dokument <tab> naslov_dokumenta`

Rang je dobiven od tražilice i relativno je nebitan za potrebe evaluacije, a dokument je naziv datoteke (bez ekstenzije) u kojoj se nalazi dokument. Primjerice "NN012927" za dokument koji se nalazi u:
"`datasets/cadial/documents/txt/NN012927.txt`," odnosno u:
"`datasets/cadial/documents/xml/NN012927.xml`."

Dodatak B
Preciznost, odziv i F_1 mjera

U domeni tražilica, računanje preciznosti (engl. *precision*) i odziva (engl. *recall*) (a time i F_1 mjere) je nepraktično. No, definiranje kvalitete i njen opis u ovisnosti o preciznosti i odzivu je praktičan jer nam omogućava da jednostavnim jezikom opišemo rezultat. Za razliku od Kendallove τ ili Spearmanove mjere ρ (McDonald, 2009) koje opisuju odnos optimalnog i dobivenog rangiranja, preciznost i odziv se temelje na relevantnosti dohvaćenih dokumenata (Manning et al., 2008):

Preciznost (engl. *precision*) pokazuje koliki dio dohvaćenih dokumenata je relevantan:

$$P = \frac{\#\{\text{dohvaćeni relevantni dokumenti}\}}{\#\{\text{svi dohvaćeni dokumenti}\}} \tag{B.1}$$

Odziv (engl. *recall*) pokazuje koliko relevantnih dokumenata je dohvaćeno:

$$R = \frac{\#\{\text{dohvaćeni relevantni dokumenti}\}}{\#\{\text{svi relevantni dokumenti}\}} \tag{B.2}$$

Pri izračunu preciznosti i odziva često se koristi sljedeća notacija:

TP (*true positive*): broj dohvaćenih relevantnih dokumenata.

FP (*false positive*): broj dohvaćenih nerelevantnih dokumenata.

TN (*true negative*): broj nedohvaćenih relevantnih dokumenata.

FN (*false negative*): broj nedohvaćenih nerelevantnih dokumenata.

Za preciznost i odziv vrijedi:

Preciznost:

$$P = \frac{TP}{TP + FP} \tag{B.3}$$

Odziv:

$$R = \frac{TP}{TP + FN} \tag{B.4}$$

Mjera koja objedinjuje preciznost i odziv je F_1 mjera:

$$F_1 = \frac{2}{\frac{1}{P} + \frac{1}{R}} \tag{B.5}$$

Mjera F_1 daje jednak značaj preciznosti i odzivu. Ona je poseban slučaj F_β mjere za $\beta = 1$:

$$F_\beta = \left(1 + \beta^2\right) \cdot \frac{P \cdot R}{\beta^2 \cdot P + R} \tag{B.6}$$

Parametar β se nalazi u intervalu $[0, \infty]$. Za $\beta = 1$ preciznost i odziv imaju jednaku težinu, a za $\beta < 1$ naglašava se preciznost.

Pri pretraživanju dokumenata radi velike količine informacija čest je slučaj da korisnici više preferiraju preciznost naspram odziva. Stoga je dobra praksa razmatrati F_β mjeru sa $\beta < 1$ (primjerice, $\beta = 0.5$).

Primjena metoda strojnog učenja za poboljšanje pretraživanja dokumenata

Sažetak

Današnja brzina stvaranja novih informacija postavlja nove izazove pri izgradnji sustava za pretraživanje informacija. S velikim količinama informacija dolazi veliki broj informacijskih domena i njihova međusobna isprepletenost (veze između domena) što zahtijeva složenije funkcije rangiranja. Zbog velike brzine stvaranja novih informacija, ručno podešavanje funkcije rangiranja postaje izuzetno težak posao te su ideje o automatizaciji podešavanja funkcija rangiranja i općenitog poboljšavanja pretraživanja sve primamljivije. Poboljšavanje pretraživanja može se ostvariti izravnim podešavanjem funkcije rangiranja ili stvaranjem vanjskih mehanizama koji pomažu pri pretraživanju informacija. U ovom radu obrađene su metode izravnog podešavanja funkcije rangiranja kroz metode učenja rangiranja te metode predlaganja upita u svrhu pomoći korisnicima pri pretraživanju informacija.

U radu se istražuje se mogućnost korištenja zapisa dnevnika upita tražilica (u literaturi poznatog kao *click-data* ili *clickthrough data*) za poboljšavanja pretraživanja dokumenata. Proučene su metode predlaganja boljih upita te poboljšavanja rangiranja korištenjem isključivo podataka dobivenih iz dnevnika upita tražilica.

Razvijene su i evaluirane nove (no ne posebno učinkovite) metode predlaganja upita i poboljšavanja rangiranja te navedeni problemi koji se susreću pri radu s podatcima dobivenim iz dnevnika upita tražilica.

Ključne riječi: Pretraživanje informacija, učenje rangiranja, predlaganje upita, strojno učenje, tražilice dokumenata, tražilica.

Using Machine Learning Methods to Improve Document Retrieval

Abstract

Present speed of creating new information sets a new challenge at building of information retrieval systems. Large amount of information brings large number of information domains which requires more complex ranking functions. Due to high speed of creating new information, manual adjustment of ranking functions becomes extremly hard. That is why automatic adjustment of ranking functions and general information retrieval improvement becomes more attractive. Information retrieval improvement can be made by direct adjustment of ranking function or by creation of mechanisms which help users to search for information. This paper describes methods of direct adjustment of ranking function by learning to rank methods and query suggestion methods as a help for users to search for information.

This paper explores possibility of using search engine query logs data (known as *click-data* or *clickthrough data*) to improve document retrieval. It contains research of query suggestion and ranking improval methods which only use search engine query log data.

It presents implementation and evaluation of new (but not very successful) methods of query suggestion and ranking improval. It also notes problems which occur while working with data collected from search engine query log.

Keywords: Information retrieval, learning to rank, query suggestion, machine learning, search engine, query log.