TakeLab

Laboratorij za analizu teksta i inženjerstvo znanja
Text Analysis and Knowledge Engineering Lab
Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva
Unska 3, 10000 Zagreb, Hrvatska

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1153

Prepoznavanje logičke posljedice u tekstovima na hrvatskome jeziku

Lana Lisjak

Zagreb, srpanj 2015.

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

Zagreb, 6. ožujka 2015.

Predmet: **Analiza i pretraživanje teksta**

DIPLOMSKI ZADATAK br. 1153

Pristupnik: **Lana Lisjak (0036450501)**
Studij: Računarstvo
Profil: Računarska znanost

Zadatak: **Prepoznavanje logičke posljedice u tekstovima na hrvatskome jeziku**

Opis zadatka:

Logička posljedica u tekstu (engl. textual entailment) jest relacija koja vrijedi između dva fragmenta teksta ako istinitost prvog fragmenta implicira istinitost drugog. Mnogi se zadatci u obradi prirodnog jezika mogu operacionalizirati kao prepoznavanje logičke posljedice u tekstu, kao što su prepoznavanje parafraze, pretraživanje informacija ili odgovaranje na pitanja. No, zbog leksičke, sintaktičke i semantičke varijacije u jeziku, automatsko prepoznavanje logičke posljedice u tekstu izazovan je zadatak.
U okviru diplomskoga rada potrebno je proučiti postupke za prepoznavanje logičke posljedice u tekstu, s naglaskom na postupke temeljene na strojnom učenju. Razraditi model za prepoznavanje logičke posljedice u tekstovima na hrvatskome jeziku temeljen na nadziranome strojnom učenju. Osnovna inačica modela neka koristi usporedbu tekstnih fragmenata temeljenu na sličnosti znakovnih nizova, po uzoru na rad (Malakasiotis i Androutsopoulos, 2007). Izgraditi odgovarajući skup podataka na hrvatskome jeziku po uzoru na engleski skup RTE3. Razviti programsku implementaciju modela te provesti iscrpno vrednovanje na odgovarajućem skupu podataka, uključivo analizu značajki i usporedbu sa referentnim modelima. Radu priložiti izvorni i izvršni kod razvijenog sustava, označene skupove podataka i potrebnu dokumentaciju te citirati korištenu literaturu.

Zadatak uručen pristupniku: 13. ožujka 2015.
Rok za predaju rada: 30. lipnja 2015.

Mentor:

Doc. dr. sc. Jan Šnajder

Djelovođa:

Doc. dr. sc. Tomislav Hrkać

Predsjednik odbora za diplomski rad profila:

Prof. dr. sc. Siniša Srbljić

SADRŽAJ

1. Uvod

Godine 2006. Ido Dagan i Oren Glickman predstavljaju izazov prepoznavanje logičke posljedice između dva teksta, s ciljem promocije apstraktnoga problema koji se koristi u mnogim primjenama u području obrade prirodnog jezika. Razni sustavi u području obrade prirodnog jezika mogli bi se uvelike optimizirati koristeći bazično znanje jezika, koje bismo mogli dobiti od sustava koji prepoznaju logičke posljedice u tekstovima. Ako imamo tekst t i hipotezu h, problem je pronaći je li hipoteza h logička posljedica teksta t. Ako je h logička posljedica t, kažemo da je logička posljedica prepoznata i označavamo takav par sa DA, kao što bi interpretirali ljudi sa znanjem odgovarajućeg jezika i potrebnog općeg znanja, inače kažemo da je logička posljedica nije prepoznata i označavamo takav par sa NE. Automatski sustavi za prepoznavanje logičke posljedice između dva teksta temeljeni su na raznim metodama koji povezuju područja obrade prirodnog jezika i strojnog učenja. Dan danas ovaj problem je izazovan i mnogi su znanstvenici pokazali velik interes za njegovo rješavanje. Sustavi za prepoznavanje logičke posljedice u tekstovima sastoje se od ulaznih podataka, modela za učenje i testiranje, a izlaz sustava su predikcije nad neviđenim skupovima podataka. U cilju boljeg istraživanja postoje više izazova, te svaki od njih ima svoj skup podataka.

U ovom radu prikazat će se jedno rješenje za prepoznavanje logičke posljedice u tekstovima na hrvatskome jeziku koristeći treći skup podataka (RTE-3) koji je predložen 2007 godine. (Giampiccolo et al., 2007) Prije izrade samog modela za prepoznavanje logičke posljedice potrebno je pripremiti podatke. Podatci su prevedeni s engleskog na hrvatski jezik i podijeljeni u potrebne podskupove. Prikazat će se i oblikovanje modela, od prikupljanja značajki do odabira potrebnih parametara unakrsnom provjerom. U sklopu ovog rada implementirano je rješenje za prepoznavanje logičke posljedice u tekstovima na hrvatskome jeziku koristeći postojeće alate za treniranje i testiranje modela strojnog učenja i za obradu hrvatskog jezika. Na kraju rada prikazat ćemo rezultate i analizirati pogreške implementiranog sustava.

Ovaj rad odvojit ćemo u pet dijelova: objašnjenje samog zadatka prepoznavanje logičke posljedice u tekstovima, opis podataka odnosno ulaz samom sustava, opis mo-

dela, testiranje modela i evaluacija rezultata. U drugom poglavlju definiran je zadatak prepoznavanja logičke posljedice u tekstovima, objašnjenja je njezina složnost i prikazani su pristupi rješavanja istog zadatka. U trećem poglavlju objašnjen je postupak prikupljanja i označivanja podataka. U četvrtom poglavlju opisan je model koji je korišten za prepoznavanje logičke posljedice. U petom i šestom poglavlju prikazani su način testiranja i rezultati modela opisanog u četvrtom poglavlju.

Tablica 1. Primjer originalnoga podataka iz RTE-3 skupa podataka

ID	Tekst	Hipoteza	Oznaka
1	At the same time the Italian digital rights group, Electronic Frontiers Italy, has asked the nation's government to investigate Sony over its use of anti-piracy software.	Italy's government investigates Sony.	NE
2	A Pentagon committee and the congressionally chartered Iraq Study Group have been preparing reports for Bush, and Iran has asked the presidents of Iraq and Syria to meet in Tehran.	Bush will meet the presidents of Iraq and Syria in Tehran.	NE
3	Parviz Davudi was representing Iran at a meeting of the Shanghai Co-operation Organisation (SCO), the fledgling association that binds Russia, China and four former Soviet republics of central Asia together to fight terrorism.	China is a member of SCO.	DA

2. Prepoznavanje logičke posljedice u tekstovima

U ovom poglavlju opisat ćemo i objasniti zadatak prepoznavanje logičke posljedice u tekstovima, prikazati dobre i loše strane formulacije zadatka i složenost problema. Prodiskutirati ćemo i o raznim pristupima rješavanja ovog zadatka.

2.1. Definicija zadatka

Prepoznavanje logičke posljedice u tekstovima formalno je definirano u (Dagan et al., 2006). Takvu definiciju ćemo i mi razmatrati, a ona glasi:

Definicija 1. *Prepoznavanje logičke posljedice u tekstovima definirano je kao usmjeren odnos između dva tekstualna izraza, označena sa t - tekst, i h - hipoteza. Kažemo da je logičke posljedice u tekstovima prepoznata, ako značenje hipoteze h može biti izvedeno iz značenje teksta t, kako bi tipično zaključili ljudi.*

Ako bolje pogledamo definiciju, za određivanje je li logička posljedica prepoznata ili ne potrebno nam je znanje ljudskog jezika i osnovno zdravorazumsko znanje. Definirati ćemo i uređeni par *t–h*, koji će označavati par teksta *t* i njegove odgovarajuće hipoteze *h*. Za nadzirano strojno učenje za svaki *t–h* par potrebna nam je i oznaka svakog uređenoga para. Oznaka nam govori je li logička posljedica prepoznata ili nije ("da" ili "ne"), označena od više osoba s različitim općem znanje i razinom znanja jezika. Moramo razmatrati da prepoznavanje logičke posljedice u tekstovima ovisi isključivo o paru *t–h* , odnosno hipoteza *h* ovisi o značenju teksta *t*. U slučaju kada je hipoteza *h* činjenica, no tekst *t* ju opovrgava logička posljedica nije prepoznata. Na primjer, ako imamo hipotezu *h* koja glasi "Trava je zelena." i tekst u kojem se opisuje kako trava nije zelena, logička posljedica neće bit prepoznata iako je hipoteza istinita uzimajući u obzir opće ljudsko znanje. Postoji i dvosmjerno prepoznavanje logičke

posljedice u tekstovima, koja zahtjeva da je *t* logička posljedica od *h* i *h* je logička posljedica *t*. Prepoznavanje logičke posljedice u tekstovima, može obuhvaćati i koncept kontradikcije određen sljedećom definicijom.

Definicija 2. *Hipoteza h iz uređenoga para t–h je kontradikcija teksta t ako bi osoba zaključila da su veze/događaji opisani u hipotezi h vrlo vjerojatno lažne s obzirom na veze/događaje opisane u tekstu t.*

Kod takve prepoznavanje logičke posljedice u tekstovima postoje tri oznake *t–h* para: da, ne i neodređeno. U vidu ovoga rada usmjeriti ćemo se isključivo na jednosmjerno prepoznavanje logičke posljedice.

Težina zaključivanja odluke o prepoznavanju logičke posljedice pojedinog *t–h* para ovisi o skupu podataka, koji može biti jednostavan i izrazito kompliciran. Kompliciranim *t–h* parovi zahtijevaju da se razumije kontekst rečenice za koje je često potrebno opće znanje. Za zahtjevnije *t–h* parove potrebno je uzeti u obzir više označivanja, jer često ljudi imaju drukčiju percepciju o logičkoj posljedici kada se govori o prirodnom jeziku.

2.2. Složenost zadatka

Kao što je složeno razumijevanje ljudskog jezika, tako je složeno i rješavanje ovog zadatka. Prijetimo da postoji puno riječi i fraza kojima možemo opisati isti pojam. Isto tako objašnjavanje veze među dvaju pojmova možemo opisati na razne načine, koje su nekad i specifične za pojedini jezik. Zbog toga često se susrećemo s rečenica koje mogu biti višeznačne. Ljudi većinom razrješavaju ovakve slučajeve pomoću konteksta, općeg znanja, logičkog zaključivanja ,poznavanju osobe koja je autor tog teksta i drugih obilježja. Mnogi sustavi, kao što su sustavi zao odgovaranje na pitanja, sumarizaciju teksta sa više dokumenata i strojno prevođenje, zahtijevaju model koji bi razriješili ovakve primjere.

Na kompleksnost zadatka ulazi i velik broj podataka, koji su većinom nestrukturirani i ne možemo ih tretirati kao jednu domenu. Složenost samog jezika uvelike otežava ovaj zadatak, te dolazi do velike potrebe predprocesiranja tekstova. Jezičnu kompleksnost možemo podijeliti na četiri razine: leksičku, sintaktičku, semantičku i razina diskursa. Za potpuno razumijevanje teksta odnosno više njih potrebno je analizirati, razaznati i naučiti sve razine jezika, no u obradi prirodnog jezika najčešće sustavi koriste leksičku, sintaktičku i eventualno semantičku analizu. Za svaki prirodni jezik potrebno je napraviti zasebne alate za predprocesiranje određenog jezika.

Kod ljudskog govora, dolazi do čestih izostavljanja riječi i pojma i usprkos tome ljudski um pomoću raznih značajki, na primjer kontekst razgovora, može zaključiti pravo značenje teksta. Ako tekst smatramo kao niz značajki, onda možemo i modelirati modele koji će pomoću tih značajki moći razaznati veze među dva teksta.

Naravno u kompleksnost razumijevanja teksta ulazi i ljudski čimbenici, kao što su laganje, sarkazam i mišljenje, koje nećemo u ovom radu obrađivati, iako se uvelike proučavaju u području obrade prirodnog jezika. Složen je i proces anotacije podataka, jer se primijetilo da ljudi često daju drukčiji sud ovisno o razini istinitosti koja prilazi iz teksta *t*. Ako imamo t_1 = "Kristina je kupila crveni auto" i h_2 = "Kristina posjeduje crveni auto", jedan označivač bi rekao da je logička posljedica prepoznata, dok bi drugi to negirao jer ne postoji vremenski interval, odnosno možda je Kristina u međuvremenu preprodala auto ili to nije isti crveni auto koja je Kristina kupila. Kako bismo se riješili ove dileme autori svakoga natjecanja u prepoznavanju logičke posljedice u tekstovima postave pravila po kojima se označuju podaci, izračunavaju razinu dogovora označivača (engl. *inter annotator agreement*) i publiciraju samo takozvani zlatni skupa podataka (krajnji, najbolji). Tako zadatak prepoznavanja logičke posljedice u tekstovima možemo promatrati kroz više područja obrade prirodnog jezika (ekstrakcija informacija, dohvaćanje informacija, sumarizacija teksta i td.).

2.3. Pristupi rješavanju zadatka

Postoje dva generalna tipa sustava korištenih za rješavanju zadatka, a to su :

1. Sustavi zasnivani na pravilima, odnosno ekspertni sustavi
2. Sustavi zasnivani na strojnom učenju

Ekspertni sustavi su dobivali mnogo slabije rezultate i zahtijevali su puno više resursa i vremena, na primjer (Szpektor et al., 2007), pa se oni rijetko koriste i nisu predmetom ovog rada. Kako smo ranije rekli tekst smatramo kao niz značajki. Takvih značajki ima puno, te svaki sustav koristi nekolicinu njih. Iako je koncept predprocesiranja prirodnog jezika isti za sve sustave, uvelike se razlikuju. Od prve definicije pa do danas, broj različitih pristupa rapidno raste i još uvijek ne možemo zaključiti koji je pristup bolji. Svakih nekoliko godina naprave se novi skupovi podataka, te se testiraju sustavi nad njima (novi i neki stari). Primijećeno je kako neki stariji sustavi koji nisu bili uspješni na prijašnjim skupovima podataka, od te godine postanu uspješni. Tu se vidi problematika obujma leksika jezika. Pošto skup podataka ima otprilike 800 *t*–

h parova, što je neznatna brojka ako ćemo uspoređivati obujam prirodnog jezika, ne možemo generalizirati rezultate. Razne pristupe podijeliti ćemo u tri skupine:

1. Semantički pristup zasniva se na traženju povezanosti teksta i hipoteza semantičkom preklapanju (engl. *semantic overlap*). Najčešće se koriste vreće riječi (engl. *Bag of Words*), koje ignoriraju sintaktička i gramatička pravila jezika. U ovaj pristup pripada i duboka i plitka semantička analiza, koje su prikazali Bos Johan et.al. u radovima (Bos i Markert, 2005a) i (Bos i Markert, 2005b).

2. Leksički pristup je najčešće baziran na leksičkim pravilima koja pokazuju da se jedna riječ može povezati s drugom. Na primjer, kupiti i posjedovati, opasno i riskantno i tako dalje. Glavna prednost leksičkih modela je jednostavna implementacija i brzina. Česti leksički pristup su mjere sličnosti između dva teksta (Malakasiotis i Androutsopoulos, 2007). Leksički pristupi ne uzimaju u obzir semantička značenja.

3. Probabilistički pristup zasniva se na oblikovanju probabilističkih modela na temelju vjerojatnosti da je hipoteza logička posljedica teksta $P(T| = H)$. Često se koriste već poznati modeli npr. Bayesovi modeli. Probabilistički pristupi se koriste samostalno (Dagan i Glickman, 2004) ili češće u kombinaciji s leksičkim značajkama (Glickman et al., 2005a) i (Glickman et al., 2006).

Ovi pristupi vezani su za samu obradu podataka i traženje sličnosti između teksta i hipoteze, dok se učenje modela bazira na metodama strojnog učenja. Najčešće metode su metoda potpornih vektora i metode grupiranja npr. EM-algoritam. Iz ovakvih pristupa moguće je prikupiti mnogo značajki, te većinom sustavi koriste značajke iz sva tri pristupa s jednim dominantnim pristupom.

3. Prikupljanje podataka

Za kvalitetnu evaluaciju i uspoređivanja raznih sustava potrebno je prikupiti podatke na korektan način kako bismo izbjegli preveliku složenost zadatka 2.2.

3.1. Prikupljanje podataka

Većina prikupljenih podataka dobivena je iz izlaznih podataka drugih sustava koji rade obradu prirodnog jezika. Prije samog prikupljanja dogovorena su četiri bitna pravila:

1. Hipoteza mora biti logička posljedica teksta, no tekst ne mora nužno biti logička posljedica hipoteze.

2. Hipoteza mora bit potpuna logička posljedica teksta. Ako bilo koji dio hipoteze nije logička posljedica teksta, logička posljedica nije prepoznata.

3. Primjeri gdje je logička posljedica vrlo vjerojatna bit će označen s DA. Na primjer, ako imamo tekst : "Ana je kupila igračku." i hipotezu "Ana posjeduje igračku." sustav će označiti s DA, iako je Ana mogla izgubiti igračku i više ju ne posjeduje.

4. Pretpostavlja se opće znanje. Na primjer predsjednik države ujedno je i državljanin te države.

Na temelju postavljenih glavnih pravila napravljen je skup podataka od 1600 *t–h* parova, gdje je 800 *t–h* para u skupu za učenje i 800 u skupu za testiranje. Svi podatci podijeljeni su u četiri skupine: ekstrakcija informacija IE (engl. *Information Extraction*), dohvaćanje informacija IR (engl. *Information Retrieval*), pitanje i odgovor QA (engl. *Question Answering*) i sažimanje teksta SUM (engl. *Summarization*).

3.1.1. Ekstrakcija informacija IE

Ova skupina inspirirana je aplikacijama koje su zasnivane na ekstrakciji informacija, gdje su tekst i predložak zamijenjeni sa *t–h* parom. *t–h* parovi napravljeni su na četiri načina:

1. Hipoteze su uzete iz ACE-2004 RDR zadatka, a tekst je izlaz iz sustava za ekstrakciju informacija iz novinskih članaka.

2. Hipoteze su uzete iz MUC-4 TST3 zadatka, a tekst je izlaz iz sustava za ekstrakciju informacija iz podataka vezanih za terorističke događaja.

3. Iz baze podataka MUC-4 i novinskih članaka automatski su generirani *t–h* parovi bazirani na ACE povezanosti.

4. Hipoteze koje nisu nađene u ACE i MUC skupu podataka korištene su da automatski naprave *t–h* parovi iz novinskih članaka.

3.1.2. Pretraživanje informacija (IR)

Hipoteze su upiti u sustave za pretraživanje informacija. Upiti su kratke izjavne rečenice, a dohvaćene informacije sa različitih sustava (npr. Google, Yahoo i MSN) predstavljaju tekst. Nakon toga označivači su provjerili je li logička posljedica prepoznata ili nije, te su označili takav *t–h* par.

Tablica 2. Primjer za IR skupinu podataka iz RTE-3 skupa podataka

ID	Tekst	Hipoteza	Oznaka
1	Between March and June, scientific observers say, up to 300,000 seals are killed. In Canada, seal-hunting means jobs, but opponents say it is vicious and endangers the species, also threatened by global warming.	Hunting endangers seal species.	DA
2	The Italian parliament may approve a draft law allowing descendants of the exiled royal family to return home. The family was banished after the Second World War because of the King's collusion with the fascist regime, but moves were introduced this year to allow.	Italian royal family returns home.	NE

3.1.3. Odgovaranje na pitanja (QA)

Za ovu skupinu autori su uzeli pitanja iz skupova podataka iz različitih natjecanja u QA kao što su TREC QA i QA@CLEF, i odgovarajuće odgovore dobivene iz QA sustava. *t–h* par je formiran na sljedeći način:

1. Odabran je odgovor sa određenim tipom, bio on točan ili netočan.
2. Pitanje je transformirano u izjavnu rečenicu s dodatnom informacijom iz odgovora.
3. *t–h* parovi su generirani tako da su izjavne rečenice hipoteze, a originalni odgovori tekstovi.

Na primjer, ako imamo pitanje *"How high is Mount Everest?"* i tekst *t "The above mentioned expedition team comprising of 10 members was permitted to climb 8848m. high Mt. Everest from Normal Route for the period of 75 days from 15 April, 2007 under the leadership of Mr. Wolf Herbert of Austria"*, onda ćemo pitanje transformirati

u izjavnu rečenicu korištenjem znanja o visini Everesta dobivenoga iz teksta *t*, pa će nam hipoteza biti *"Mount Everest is 8848m high"*.

3.1.4. Sažimanje teksta SUM

Podatci SUM generirani su dvama postupcima. U prvom postupku,i, tekstovi i hipoteze su rečenice uzete iz određenih grupa iz grupiranih novinskih dokumenata. Označivačima je dan izlaz iz više-dokumentnih sustava za sažimanje dokumenata uključujući oznaku grupe dokumenta i generirani sažetak za svaku grupu (iz sažetka su najčešće uzeti tekstovi t). Odabrane su rečenice s velikim leksičkim preklapanjem. Za pozitivne primjere hipoteza je bila pojednostavljena tako da su brisali dijelove rečenice dok hipoteza nije postala potpuna logička posljedica teksta *t*. Negativni primjeri su odabrani na sličan način. Drugi se postupak temelji na eksperimentalnoj metodi evaluacije u DUC 2005. (Passonneau et al., 2005)

3.1.5. Završni skup podataka

Svaki par podataka označila su tri označivača. Na skupu za testiranje označivači koji su imali minimalno 100 zajedničkih parova prosječno su se složili u 87.8% slučaja, s prosječnom vrijednošću kapa-statistike od 0.75. 19.2% podataka su maknuti iz završnog skupa jer se označivači nisu mogli složiti oko njihovih oznaka. Uz to, 9.4% podataka su maknuti zbog prevelike ili premale složenosti u odnosu na ostale podatke. Podatci su često morali biti lektorirani (jer su glavni izvori podataka internetski sustavi) i provjereni još jednom od strane stručnjaka. Na kraju se dobio skup podataka od 1600 *t–h* parova koji se naziva i zlatni skup podataka. U sklopu ovog rada koristili smo podskup od 400 *t–h* parova zlatnog skupa, preveden s engleskog na hrvatski jezik. Podskup sadrži 100 *t–h* para od svake prijašnje objašnjenje skupine. Podskup je podijeljen na skup za treniranje (70 %) i skup za testiranje (30 %).

4. Model

Prepoznavanje logičke posljedice između dva teksta je tipičan binarni klasifikacijski problem koji na ulaz prima skup značajki, a na izlaz je binarna odluka koja označava je li prepoznata logička posljedica. U ovom poglavlju objasnit ćemo što su značajke i kako smo ih prikupi, te klasifikator koji smo koristili. U poglavlju 3 opisani su prikupljeni podatci, te njihovo označivanje. Za potrebe našeg sustava podatci su amaterski prevedeni s engleskoga jezika na hrvatski jezik. Podatci su podvojeni na skup za testiranje i skup za učenje, koji se ponekad koristio i kao skup za treniranje. Skup za testiranje, kao što bi se moglo slutiti iz naziva, služi samo za završno testiranje sustava, te se pogreška nad tim skupom naziva pogreška generalizacije. Anotacije skupa za testiranje se isključivo koriste kako bi usporedili krajnje rezultate, odnosno kako bi izračunali točnost sustava. Pogreška generalizacije prikazuje kako se naš sustav ponaša na neviđenim podacima, te pokazuje sveukupnu točnost rada sustava. Nad skupom za učenje vrši se učenje modela i odabir hiperparametara modela, ako takvi postoje, odnosno radi se testiranje i optimizacija modela. Anotacije skupa za učenje se direktno koriste tijekom učenja modela, te zbog toga kažemo da se koristi nadzirano strojno učenje.

4.1. Prikupljanje značajki

Pošto računalo ne razumije prirodni jezik niti njegove strukture, potrebno je prijenosni ljudsko znanje jezika računalu u njemu poznatom obliku (brojevi). U ovom radu bitan nam je samo odnos dva teksta, te ćemo definirati prikaz i računanje tih odnosna odnosno značajka. Svaki *t–h* potrebno je prikazati kao niz značajki, u našem slučaju vektor realnih brojeva dimenzije n. Značajke su numerički prikaz sličnosti, odnosno različitosti teksta *t* i hipoteze *h*. Skup značajki svih *t–h* parova i anotacijske oznake istih čine ulaz klasifikatora opisan u 4.2. Za ekstrakciju značajki koristi ćemo 7 reprezentacija *t–h* para, 9 mjera sličnosti i 4 dodatne značajke. Od svih 9 značajki samo je jedna semantička značajka 4.1.4, te bismo rekli da mi imamo najizraženiji leksički

pristup rješavanju zadatka.

4.1.1. Reprezentacija *t–h* para

Svaka reprezentacije sadrži dva niza znakova. Niz znakova koji reprezentira tekst i niz znakova koji reprezentira hipotezu.

1. Dva niza znakova koji sadrže originalan *t–h* par bez interpunkcijskih znakova.
2. Dva niza znakova koji sadrže lematizirani *t–h* para.
3. Dva niza znakova koji sadrže POS oznake *t–h* para.
4. Dva niza znakova koji sadrže imenice *t–h* para.
5. Dva niza znakova koji sadrže lematizirane imenice *t–h* para.
6. Dva niza znakova koji sadrže glagole *t–h* para.
7. Dva niza znakova koji sadrže lematizirane glagole *t–h* para.

Lematizacija i označavanje vrsta riječi

Lematizacija je proces svođenje riječi na osnovni oblik (lemu). Pomoću lematizacije zanemarujemo morfološke oznake (rod, broj, padež i druge), kako bismo koristili samo značenje riječi u traženju sličnosti dva teksta. Lematizacija se vrši pomoću već implementiranog alata za obradu hrvatskoga jezika PreprocessingHR, koji je napravljen u laboratoriju TakeLab na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu.

Rečenica se sastoji od rečeničnih dijelova, a to su predikat (temeljni rečenični dio koji otvara mjesto ostalim rečeničnim dijelovima), subjekt (vršitelj predikatne radnje), objekt (predmet glagolske radnje), priložna oznaka (okolnost glagolske radnje), apozicija (dopuna imenici) i atribut (dopuna imenici). POS oznake označavaju vrstu riječi. Ekstrakcijom oznaka dobivamo strukturu rečenice apsolutno zanemarujući njezino značenje. Ovakav zapis nam je bitan za uspoređivanje dva teksta ako uzmemo u obzir: ako su strukture dva teksta slične, postoji veća mogućnost da je značenje tih tekstova slično. Ekstrakcija POS oznaka vrši pomoću alata za obradu hrvatskoga jezika (Agić et al., 2013).

4.1.2. Mjere sličnosti

Kao mjere sličnosti između dva teksta koristili smo sljedeće mjere.

Levenshteinova udaljenost

Levensteinova udaljenost (engl. *edit distance*) definira broj akcija potrebnih da se iz jednog znakovnoga niza dobije drugi, pri čemu je dozvoljeno dodavanje, brisanje ili izmjena znakova na bilo kojoj lokaciji unutar niza. Na primjer udaljenost između riječi "pas" i "as" iznosi 1, a udaljenost između "pas" i "nasip" iznosi 3.

Jaro-Winker udaljenost

Jaro-Winker udaljenost je varijacija Jaro udaljenosti. Jaro udaljenost d_j između teksta s_1 i teksta s_2 je definirana kao:

$$d_j(s_1, s_2) = \frac{m}{3 \cdot l_1} + \frac{m}{3 \cdot l_2} \frac{m-7}{3 \cdot m}$$

gdje su l_1 i l_2 duljine (u znakovima) tekstova s_1 i s_2. Vrijednost m je broj istih znakova, a *t* je polovica transpozicijske vrijednosti. Dva znaka su ista ako su identična i ako je međusobna udaljenost među njima manja od $\frac{max(l_1,l_2)}{2} - 1$. Transpozicijska vrijednost se dobiva tako da iz tekstova s_1 i s_2 izbacimo sve znakove koji nisu isti i izbrojimo pozicije koje ne sadrže iste znakove. Tada je Jaro-Winker udaljenost:

$$d_w(s_1, s_2) = d_j(s_1, s_2) + l \cdot p \cdot [1 - d_j(s_1, s_2)]$$

gdje je l duljina najdužeg zajedničkog prefiksa i p je konstanta. Parametar p kontrolira utjecaj vrijednosti l na udaljenost $d_w(s_1, s_2)$. Za ovu implementaciju koristili smo $p = 0.1$.

Manhattanova udaljenost

Za dva vektora $\overrightarrow{x} = (x_1, x_2, \cdots, x_n)$ i $\overrightarrow{y} = (y_1, y_2, \cdots, y_n)$, gdje je n broj jedinstvenih riječi koji se pojavljuju u tekstovima s_1 i s_2. Manhattanovi udaljenost definiramo kao:

$$l_1(\overrightarrow{x}, \overrightarrow{y}) = \sum_{i=1}^{n} |x_i - y_i|$$

Euklidska udaljenost

Za dva vektora $\overrightarrow{x} = (x_1, x_2, \cdots, x_n)$ i $\overrightarrow{y} = (y_1, y_2, \cdots, y_n)$, gdje je n broj jedinstvenih riječi koji se pojavljuju u tekstovima s_1 i s_2. Euklidsku udaljenost definiramo

kao:

$$l_2(\overrightarrow{x}, \overrightarrow{y}) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Kosinusna sličnost

Imamo dva binarna vektora $\overrightarrow{x} = (x_1, x_2, \cdots, x_n)$ i $\overrightarrow{y} = (y_1, y_2, \cdots, y_n)$, gdje je n broj jedinstvenih riječi koji se pojavljuju u cijelom skupu podataka (na primjer RTE3). Svaki indeks vektora označava jednu jedinstvenu riječ iz skupa podataka. Vrijednost x_i će biti 1 ako se riječ pojavila u tekstu s_1, ako nije biti će 0. Analogno vrijedi za y_i i tekst s_2. Sličnost definiramo kao:

$$cos(\overrightarrow{x}, \overrightarrow{y}) = \frac{\overrightarrow{x} * \overrightarrow{y}}{\|\overrightarrow{x}\| \cdot \|\overrightarrow{y}\|}$$

N-gram udaljenost

Udaljenost se računa slično kao i $l_1(\overrightarrow{x}, \overrightarrow{y})$, ali umjesto pojedinih riječi uzimamo jedinstvene skupove znakova veličine 3 ($n = 3$).

Koeficijent podudarnosti

Računa se kao $|X \bigcap Y|$, gdje su X i Y nizovi jedinstvenih riječi iz tekstova s_1 i s_2. Vraća koliko ima istih riječi u tekstovima s_1 i s_2.

Diceov koeficijent

Računa se kao:

$$d = \frac{2 \cdot X \bigcap Y}{|X \bigcup Y|}$$

gdje su X i Y nizovi jedinstvenih riječi iz tekstova s_1 i s_2.

Jaccardov koeficijent

Računa se kao:

$$d = \frac{|X| \bigcap Y}{|X| + |Y|}$$

gdje su X i Y nizovi jedinstvenih riječi iz tekstova s_1 i s_2.

4.1.3. Parcijalne mjere sličnosti

Kako je najčešće tekst *t* puno duži od hipoteze *h*, uvest ćemo parcijalno računanje mjere sličnosti. Ako je duljina teksta s_1 veća od duljine teksta s_2, sličnosti f_i između dva teksta s_1 i s_2 je $f_i(s'_1, s_2)$, gdje su $f_i(0 < i < 10)$ mjere sličnosti i s'_1 je dio teksta s_1 duljine kao i s_2. Izaberemo onu s'_1 kojoj je prosječni f_i maksimalan.

$$s'^{*}_1 = argmax \sum_{i=1}^{10} f_i(s'_1, s_2)$$

Kao novih 11 značajki zadržati ćemo deset f_i i s'^{*}_1. Analogno za slučaj ako je duljina teksta s_2 veća od duljine teksta s_1. Parcijalne mjere sličnosti mogu se primijeniti na sve reprezentacije *t–h* para. Stvarna primjena prikazana je u poglavlju 5.

4.1.4. Značajke temeljene na WordNet-u

WordNet je računalni resurs na semantičkoj razini koji nam daje podatke o leksiku pojedinog jezika. WordNet obuhvaća samo četiri najveće skupine vrsta riječi – imenice, glagole, pridjeve i priloge. To je i razumljivo s obzirom na usmjerenost prema semantičkoj razini jer su ovo ujedno i semantički pune riječi. Uz pojedinačne lekseme, u WordNetu se mogu naći i kolokacije (poput *morski pas*) te idiomi (primjerice *otegnuti papke*). Osnovna je gradbena jedinica svakoga WordNeta sinonimski grozd. Jedan sinonimski grozd je zapravo koncept kojemu su pridružene njegove moguće leksikalizacije – sinonimski leksemi. No, osim što sadrži listu sinonimskih leksema riječi, svaki sinonimski grozd istovremeno sadržava i skup logičko-semantičkih odnosa kojima je povezan s drugim grozdovima. Osim već spomenute sinonimije najvažnijeg odnosa, kodirane su i hiperonimija, odnosno hiponimija, meronimija i antonimija, koje su posebno važne za imenice ali i troponimija, implikacija, kauzalnost (za glagole).

Na slici 4.1 prikazan je jedan sinonimski grozd. Svaki gvozd ima svoju oznaku (na ovom primjeru ENG30-00002098-a), njegovo korištenje poveznice s drugim sinonimskim grozdovima ($itrs$) i sinonime. Tako jedan sinonimski grozd ne mora nužno označavati jednu riječ, nego označava sve riječi koje su navedene u polju sinonimima ($synonyms$). Kao značajke uzimati ćemo poveznice između sinonimskih grozdova, odnosno gledati ćemo samo polje $itrs$. Za sve vrste poveznice, ako postoji poveznica između bilo koja dva sinonimska grozdova koji označuju riječi teksta *t* i hipoteze *h*, značajka za tu poveznicu je 1, inače 0. Primjer vrsta poveznica su *be_in_state* , *near_antonym*, *verb_group*, *hypernym* i druge.

Slika 4.1: Sinonimski grozd

```
{
   "ENG30-00002098-a": {
        "bcs": 3,
        "def": "onaj koji nema potrebne sposobnosti ili
           sredstva da \u0161to u\u010dini",
        "domain": "quality",
        "ilrs": {
            "be_in_state": [
                "ENG30-05200169-n"
            ],
            "near_antonym": [
                "ENG30-00001740-a"
            ]
        },
        "notes": {
            "usage": [
                "Bez auta nisam u mogu\u0107nosti do\u010
                   7i do grada.",
                "On je nesposoban za taj posao."
            ]
        },
        "pos": "a",
        "stamp": "matea@MATEA-PC.dhcp.ffzg.hr 2012-04-12
           11:13:19",
        "sumo": {
            "equal": [
                "capability"
            ]
        },
        "synonyms": [
            {
                "sense": 1,
                "word": "nesposoban"
            },
            {
```

```
                    "sense": 3,
                    "word": "nemo\u0107an"
                }
            ]
        }
    }
```

4.1.5. Dodatne značajke

Koristili smo i četiri dodatne značajke:

1. Binarna oznaka ako postoji niječna riječ u tekstu *t*.
2. Binarna oznaka ako postoji niječna riječ u hipotezi *t*.
3. $\frac{min(l_t,l_h)}{max(l_t,l_h)}$, gdje su l_t i l_h duljine tekstova s_1 i s_2 .
4. Binarna oznaka koja označava oznaku duljine u skupu RTE3. Ako je oznaka duljine para *t–h* "long", onda je oznaka postavljena na 1 inače 0.

4.2. Klasifikator

Klasifikacija ili kategorizacija (engl. *classification*) je postupak dodjeljivanja oznaka prethodno definiranih klasa nestrukturiranim podacima, u našem slučaju *t–h* uređenim parovima. Proces klasifikacije je nadgledani proces učenja, budući da je proces učenja vođen, odnosno nadgledan znanjem o klasama koje se stječe na osnovi primjera za učenje. Metode klasifikacije podataka potječu iz šezdesetih godina prošlog stoljeća. U ovom radu klasifikator se koristi kao metoda učenja modela, odnosno značajki koje opisuju odnos teksta *t* i hipoteze *h*.

Prepoznavanje logičke posljedice između dva teksta je binarni problem s izlazom 1 ako je logička posljedica prepoznata, inače je izlaz 0. Po uzoru na (Malakasiotis i Androutsopoulos, 2007) korišten je klasifikator metode potpornih vektora SVM (engl. *support vector machines*), s dvije klase.

4.2.1. Metoda potpornih vektora

Objasnit ćemo metodu potpornih vektora po uzoru na (Dobša, 2006). Metoda potpornih vektora (engl. *support vector machines, SVM*) je algoritam klasifikacije koji su

predstavili Vapnik i suradnici 1992. godine (Bosner et al., 1992) i od tada je vrlo moćan aparat koji je nadmašio do tada poznate algoritme u širokom rasponu primjena. Razlog za ovakvo dobru izvedbu SVM metode prvenstveno je čvrsta teorijska utemeljenost koja se odnosi na mogućnosti kontroliranja tzv. pogreške generalizacije. Generalizacija je sposobnost hipoteze da ispravno klasificira podatke koji nisu u skupu podataka za učenje.

Klasifikator SVM nalazi hiper-ravnine ili skup hiper-ravnina koje imaju najveću marginu razdvajanja klasa, gdje je margina najveća udaljenost do najbližih točaka bilo koje klase, korištenjem hipoteza linearnih funkcija. U nekim slučajevima podaci u originalnom prostoru ne mogu biti razdijeljeni korištenjem linearnih funkcija. Zbog toga se podaci preslikavaju, najčešće u visokodimenzijski prostor u kojem mogu biti razdijeljeni korištenjem linearnih hipoteza. Takav se prostor naziva prostor značajki. Ovo je preslikavanje moguće izvesti implicitno, korištenjem jezgrenih funkcija (engl. *kernel*). Direktno izračunavanje skalarnih produkata u prostoru značajki iz vrijednosti u ulaznom prostoru provodi se korištenjem jezgrenih funkcija. Uvođenje jezgrenih funkcija znatno je povećalo ekspresivnost linearnih metoda učenja i povećalo mogućnost prenaučenosti modela. Ako klasifikator uzima u obzir samo najveću marginu razdvajanja klasa tada ne dopušta pogrešnu klasifikaciju primjera za učenje, pa ga se može primijeniti samo na skupu podataka koji su linearno razdjeljivi u prostoru značajki. To bitno ograničava mogućnosti primjene ovog algoritma i tim je nedostatkom motiviran razvoj klasifikatora meke margine (engl. *soft margin classifier*).

4.2.2. Klasifikator meke margine

Klasifikator meke margine predstavljen je 1995. godine (Cortes i Vapnik, 1995) i njegova je prednost nad klasifikatorom maksimalne margine u tome što radi i za skup primjera za učenjem koji nije linearno razdjeljiv u prostoru značajki. U ovom će se slučaju dozvoliti pogrešna klasifikacija primjera za učenje, ali će se takvi slučajevi minimizirati. Dakle, općenito uzevši, dopustit će se odstupanje primjera za učenje od funkcionalne margine vrijednosti 1. Ovo se odstupanje mjeri varijablom olabavljene margine ζ. Klasifikator minimizira kombinaciju vektora težine i gornje ograničenje na broj pogrešno klasificiranih primjera za učenje. Gornje ograničenje broja pogrešno klasificiranih primjera je izraženo u terminima varijable olabavljene margine ζ. U ovom slučaju varijabla olabavljene margine ζ mjeri koliko određeni primjer odstupa od toga da margina skupa primjera za učenje bude 1.

$$y(w * x_i - b) < 1 - \zeta_i, i1 \leq i \leq n$$

gdje je n broj podataka za učenje. Optimizacijski problem SVM linearnog modela s mekim marginama je:

$$f(x) = \min \frac{1}{2}||w||^2 + C * \sum_{i=1}^{n} \zeta_i)$$

,

gdje je w vektor normale hiper-ravnine, a $\sum_{i=1}^{n} \zeta_i$ je gornje ograničenje na broj pogrešno klasificiranih primjera.

Parametar C je od velike važnosti. On regulira odnos između pogreške treniranja i testiranja. Pogreška treniranja je pogreška izračunata nad skupom podataka za treniranje, dok je pogreška testiranja ili pogreška generalizacije izračunata nad rezultatima testnog skupa. Sada možemo primijetiti da će veliki parametar C pokušati čim više smanjiti $|w|$, što će rezultirati hiper-ravninom koja će se probati što bolje klasificirati svaki primjer u skupu podataka. Ovakvo ponašanje klasifikatora povećati će pogrešku generalizacije, te će za jako veliki parametar C doći do prenaučenosti klasifikatora. U drugu ruku, ako postavimo C na premalen broj, klasifikator će povećavati $|w|$ te će više primjera krivo klasificirati. Mali parametar C može rezultirati podnaučenosti modela. Za kvalitetan izbor hiperparametara SVM-a potrebno je napraviti adekvatno testiranje modela.

4.2.3. Klasifikacija podataka

Za klasifikaciju podataka korišten je već implementiran klasifikator SVM u Pythonovoj biblioteci skilearn koji koristi meke margine. Osnovni oblik klase klasifikator SVM izgleda kako slijedi:

```
class sklearn.svm.SVC(C=1.0, kernel='rbf', degree=3,
 gamma=0.0, coef0=0.0, shrinking=True, probability=False,
  tol=0.001, cache_size=200, class_weight=None,
  verbose=False, max_iter=-1, random_state=None)
```

Kao što se može vidjeti, SVM klasifikator ima više podesivih parametara. Drugi parametar, i najbitniji, je odabir jezgrenih funkcija $kernel$. Jezgrene funkcije definiraju skalarni produkt dva vektora odnosno dva vektora značajki, te nam omogućava bolju separaciju primjera i mapiranje u višedimenzionalne prostore. Jezgrena funkcija može biti:

1. linearna, $K_{x,y} = (x^T * y)$
2. polinomijalna, $K_{x,y} = (\gamma(x^T * y) + r)^d$, s dodatnim parametrom stupnjem d ($degree$) i koeficijentom r ($coef0$)
3. rbf, $K_{x,y} = (-\gamma * |x - y|)^2$, gdje γ ($gamma$) mora biti veća od 0
4. sigmoidalna, $K_{x,y} = (tanh((x^T * y) + r)$, s koeficijentom r ($coef0$).

Testiranje modela i odabir hiperparametara prikazat ćemo u poglavlju 5. Na kraju testiranja SVM će imati sljedeći izgled:

```
clasifficator=svm.SVC(kernel= 'linear', C=0.7)
clasifficato.fit(features, tags)
```

Funkcija $fit(features, tags)$ vrši učenje klasifikatora, s ulazima značajki ($features$) i oznakama klasa ($tags$). Za predikciju novih primjera koristi se jednostavna metoda $predictions = svm.predict(newTestFeatures)$. Kada smo izabrali linearnu jezgrenu funkciju, mogli smo koristiti i linearni klasifikator SVM $classsklearn.svm.LinearSVC$, koji ima dodatne parametre. Pošto se rezultati nisu značajno mijenjali korištenjem drugih parametara, korišten je gore opisan klasifikator.

5. Testiranje modela

Nakon ekstrakcije značajki potrebno je izabrati hiperparametre modela i koje ćemo značajke koristiti u ulazu kalsifikatora. Kao evaluacijske metode izračunali smo točnost (engl. *accuracy*) , preciznost (engl. *precision*) i odziv (engl. *recall*).

Točnost je udio točno klasificiranih primjera u skupu svih primjera.

$$\text{točnost} = \frac{\text{broj točno klasificiranih podataka}}{\text{ukupni broj podataka}}$$

Preciznost je udio točno klasificiranih primjera u skupu pozitivno klasificiranih primjera.

$$\text{preciznost} = \frac{\text{broj točno klasificiranih podataka}}{\text{broj pozitivno klasificiranih primjera}}$$

Odziv je udio točno klasificiranih primjera u skupu svih pozitivnih primjera.

$$\text{odziv} = \frac{\text{broj točno klasificiranih podataka}}{\text{broj pozitivih primjera}}$$

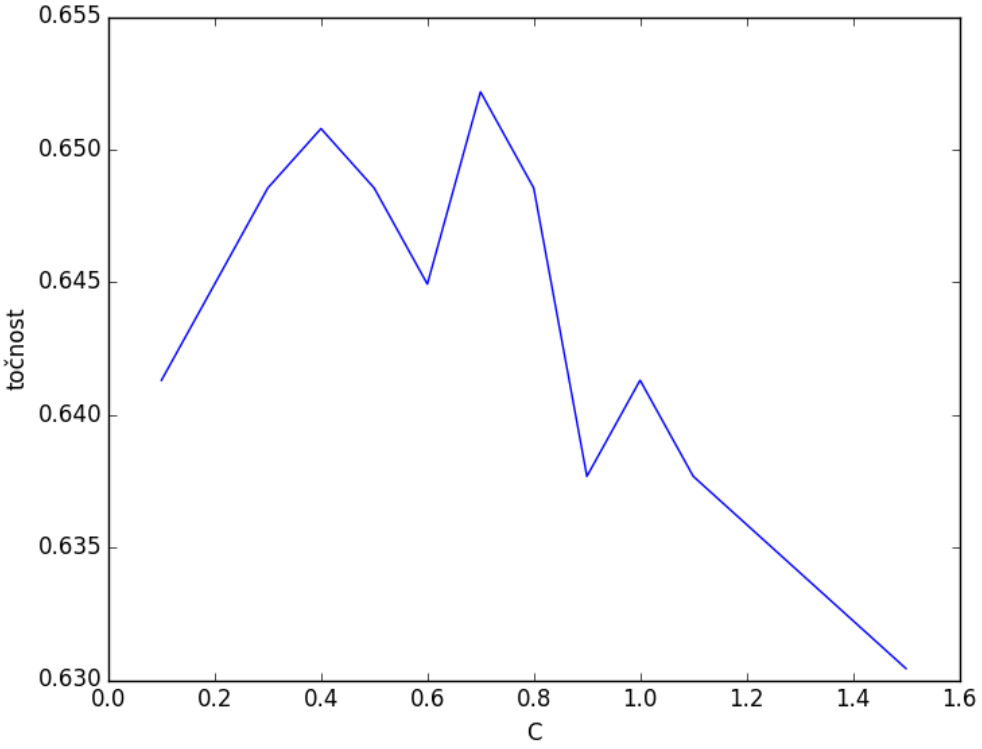
Kada je prepoznata logička posljedica u tekstovima, kažemo da je primjer pozitivan, inače je negativan. Uzimajući u obzir da je podjednako pozitivnih i negativnih primjera, odnosno onih među kojima je prepoznata i nije prepoznata logička posljedica, i da su nam pozitivni i negativni primjeri jednako bitni, točnost je sasvim dovoljna mjera za uspješnost sustava, no prikazati ćemo i preciznost i odziv zbog kasnije analize. Kako podskup skupa RTE3 sadrži četiri različite skupine podataka, koje su prikupljane raznim metodama i s time imaju različita obilježja, bilo bi poželjno koristiti pojedinačni SVM klasifikator za pojedinu skupinu. Povođeni time modelirana su dva modela.

1. Model 1 - korištenje jednog klasifikatora nad cijelim skupom podataka sa svim prikupljenim značajkama.

2. Model 2 - korištenje pojedinačnog klasifikatora za svaki podzadatak, odnosno podskup. Koriste se četiri klasifikatora nad podskupovima: IE, IR, QA i SUM, s različito odabranim značajkama po uzoru na Malakasiotis i Androutsopoulos (2007). Odabir značajki prikazan je u tablici 4.

5.1. Odabir hiperparametara

Kako bismo adekvatno odabrali hiperparametre klasifikatora, provedena je k-struka unakrsna provjera. Skup podataka podijeli se na k jednakih preklopa $skupPodataka = p_1, p_2 \cdots, p_k$. Model učimo na $k-1$ preklopa, te testiramo na k-tom preklopu, te testiramo na k-tom preklopu pomičući poziciju prekopa k . U prvom prolazu testiramo model na preklopim $skupZaTreniranje = p_1, p_2 \cdots, p_{k-1}$, a testiramo na preklopu $skupZaTestiranje = p_k$, te svakom iteracijom kružno mijenjamo preklope. Za svaku iteraciju izračunamo pogrešku sustava, odnosno u našem slučaju točnost sustava acc_i, gdje je $i = (1, 2, \cdots, k)$. Pogreška odnosno točnost sustava je prosječna pogreška svih iteracija $acc = \frac{\sum_{i=1}^{k} acc_i}{k}$. Za potrebe našeg sustava parametar k postaviti ćemo na 5. Kako bismo izbjegli prenaučenost modela, unakrsnu provjeru provesti ćemo nad skupu za učenje koristeći model 1. U 5.1 prikazali smo linearnu jezgrenu funkciju u ovisnosti o parametru C. Druge jezgrene funkcije smo ispitali sa svim parametrima, no nismo dobili nikakve značajne promjene niti poboljšanja.



Slika 5.1: Testiranje modela unakrsnom provjerom.

Parametar C koji mora biti veći od 1.0, je parametar kažnjavanja (što je veći to

model više kažnjava rezultat, odnosno uzima manji udio izračunatoga rezultata u obzir), koji je koristan ako skup podataka ima mnogo šuma, te tako smanjuje prenaučenost modela. Rezultatima unakrsne provjere izabrani su sljedeći hiperparametri : $kernel = linear$ i $C = 0.7$. Najbližu točnost 0.5833 od je imalo model s polinomijanlom jezgrenom funkcijom sa stupnjem $d = 4$ ($degree$).

5.2. Odabir značajki

Prije opisani model 2 ima već odabrane značajke po uzoru na (Malakasiotis i Androutsopoulos, 2007), te u ovom poglavlju nećemo razmatrati taj model. Model 1 sadrži sve značajke od kojih neke mogu biti više i manje bitnije, te neke mogu čak biti i nebitne. Tako ćemo napraviti model 3 s odabranim značajkama. Kako bismo odlučili koje su najbitnije značajke, a koje su redundantne značajke su podijeljene u šest skupina:

1. Wordnet značajke - koje su opisane u 4.1.4.
2. Imenske značajke - sastoje se od izačunatih svih mjera sličnosti nad 4. i 5. reprezentaciji *t–h* para opisane u .
3. Glagolske značajke - sastoje se od izačunatih svih mjera sličnosti nad 6. i 7. reprezentaciji *t–h* para opisane u .
4. Parcijalne sličnosti - sastoje se od izračunatih parcijalnih sličnosti svih reprezentacija *t–h* para
5. POS značajke - sastoje se od izračunatih svih mjera sličnosti nad 3. reprezentaciji *t–h* para opisane u
6. Dodatne značajke - koje su opisane u 4.1.5.

Značajke vezane za 1. i 2. reprezentaciju *t–h* para smatraju se osnovnim, te se u svakom testiranju koriste. Ovim testiranjem htjeli smo pokazati utjecaj pojedine grupe značajki na točnost nad skupom za učenje, dobivši tako dodatnih 6 modela. Skup za učenje je nasumično podijeljen na dva podskupa: podskup za treniranje (75% skupa za učenje) i skup za testiranja (25% skupa za učenje). Kako bismo prikazali pojedinu skupinu, testirali smo model sa svim značajkama bez određene skupine značajki. Testirano je ukupno 5 puta. Svaki puta je izračunata točnost modela te je krajnji rezultat njihova prosječna točnost. Rezultati su prikazani u Tablici 3.

Tablica 3. Testiranje skupova značajki

Značajke	Prosječna točnost modela
bez značajki WordNet-a	0.6232
bez dodatnih značajki	0.6899
bez imeničnih značajki	0.6579
bez glagolskih značajki	0.6696
bez parcijalne sličnosti	0.6232
bez POS značajke	0.4899

Kroz analizu podataka, svakako prvo primjećujemo kako su POS značajke bitne. To znači da nam je bitna sama forma rečenice, odnosno ako je sličniji oblik rečenice postoji veća vjerojatnost kako će logička posljedica biti prepoznata. Pošto jezik ima striktna pravila formiranja rečenica, postoji velika šansa da vrijedi: Rečenice su međusobno sličnije ako imaju sličniji oblik. U našem slučaju takav zaključak je opravdan, no ne bismo generalizirali i rekli da to uvijek vrijedi. Sljedeće po važnosti su značajke wordnet i parcijalne značajke, dok su najmanje važne dodatne značajke. Dodatne značajke ne predstavljaju nikakve semantičke niti leksičke značajke, pa nas ne čudi ovakav rezultat. Više su orijentirane na duljine hipoteze h i teksta t koji imaju jako mali utjecaj na prepoznavanje logičke posljedice. Sada smo dobili i treći model Model 3, koji ima sve značajke osim dodatnih značajki.

Tablica 4.Odabir značajki Modela 2

Opis skupa značajki	Broj	IE	IR	QA	SUM
mjere sličnosti s reprezentacijom 1	9	X	X	X	X
mjere sličnosti sa reprezentacijom 2	9	X	X	X	X
mjere sličnosti sa reprezentacijom 3	9	X	X		
prosječna parcijalna mjera sličnosti sa reprezentacijom 1	1				X
prosječna parcijalna mjera sa reprezentacijom 2	1			X	X
prosječna parcijalna mjera sličnosti sa reprezentacijom 3	1			X	X
parcijalne mjere sličnosti sa reprezentacijom 1	9				
parcijalne mjere sličnosti sa reprezentacijom 2	9				X
parcijalne mjere sličnosti sa reprezentacijom 3	9	X			
negacija *t*	1	X			
negacija *h*	1	X			
omjer duljina	1	X			
mjere sličnosti sa reprezentacijom 4	9	X			
mjere sličnosti sa reprezentacijom 5	9				
mjere sličnosti sa reprezentacijom 6	9				X
mjere sličnosti sa reprezentacijom 7	9				
oznaka duljine para	1	X		X	
ukupno:	97	49	29	21	39

6. Rezultati

U ovom poglavlju razmatrati ćemo točnost skupa za testiranje, što znači da je model učen na skupu za učenje, a testiran na skupu za testiranje. U tablici 5 prikazan je referentni rezultat za svaki podzadatak koji je izračunat kao najveći udio klase u skupa za učenje. Ako sustav ima veću točnost od referentnog rezultata, rezultat se može uzeti u obzir.

Tablica 5. Osnovica zadatka i podzadataka

Zadatak / podzadatak	Osnovica
svi podaci	50.36 %
IE	50%
IR	55%
SUM	52%
QA	59%

Tablica 6 prikazuje rezultate nad skupovima za testiranje modela Model 1 i Model 2. Točnost Modela 2 prikazan je točnostima pojedinih zadataka nad određenim podskupovima za testiranje. Dodatno smo testirali Model 1 i Model 2, sa značajkom WordNet i bez nje kako bi dobili uvid u njezin utjecaj na točnost modela.

Tablica 6. Testiranje Modela 1, Modela 2 i Modela 3

Opis sustava	Točnost	Preciznost	Odziv
Model 1 bez značajki WordNet-a	0.6532	0.8181	0.7642
IE	0.3871	0.5454	0.5714
IR	0.8387	0.8965	0.9286
QA	0.67774	0.9130	0.7241
SUM	0.6129	0.8388	0.7093
Model 1 sa wordNet značajkama	0.6935	0.8686	0.7747
IE sa wordNet značajkama	0.4516	0.6087	0.6363
IR sa wordNet značajkama	0.8387	0.8966	0.9286
QA sa wordNet značajkama	0.6774	0.9130	0.7241
SUM sa wordNet značajkama	0.6452	1	0.6452
Model 3	0.6693	0.8384	0.7685
Model 3 IE	0.6812	0.7833	0.8392
Model 3 IR	0.7826	0.8852	0.87096
Model 3 QA	0.8441	0.8923	0.9355
Model 3 SUM	0.8551	0.9672	0.8806

Prije samih dodavanja WordNet značajki vidimo da je velika točnost skupine IR koja ima samo 26 značajki, dok je manja točnost skupine IE koja ima najviše značajki (49) i jedina koja nema točnost veću od osnovice. Ako uspoređujemo sustava sa jednim klasifikatorom sa sustavom sa četiri klasifikatora, primijetit ćemo da je zbog niske točnosti skupine IE prvi sustav malo bolji no i dalje su sustavi približno jednaki. Ako uzimamo prosječnu točnost kao točnost sveukupnog sustava na cijelom skupu podataka, tada sustav sa jednim klasifikatorom ima veću točnost za otprilike 3%. Dodavajući značajke hrvatskog WordNeta točnost se povećava u većini slučajevima, dok je u dugima (IR i QA) ostala ista. Kod obje metode klasifikatora to posvećenje je malo, kod podzadatka IE povećala za 7% što uopće nije zanemarivo povećanje. Primijetimo da je kod pod zadatka SUM nakon dodavanja wordNet značajki preciznost 1, odnosno svi pozitivni *t–h* parovi su točno klasificirani, no smanjio se odziv što je bilo očekivano. Zbog testiranja na malom skupom podataka, koji smo poslije još podijelili na 4 dijela nije neočekivano da su niži rezultati kada smo koristili 4 klasifikatora. Metodu 3 smo testirali s jednim klasifikatorom i sa četiri klasifikatora. Kada je testirana s jednim klasifikatorom, točnost je približno jednaka drugim sustavima, no kada koristimo četiri klasifikatora točnost je izrazito velika. Pošto imamo jako mali skup podataka, ti podaci nisu najvjerodostojniji, no i dalje su prikazani kako bi se vidjela razlika između modela.

6.1. Analiza pogreške

U ovom poglavlju analizirati ćemo par primjera koje su sve metode pogrešno klasificirale, te ćemo prokomentirati razlog tome. Primjere ćemo prikazati u originalnom obliku kao xml linije, gdje je tekst *t* nalazi između oznaka $< t >$ i $< /t >$, a hipoteza *h* unutar oznaka $< h >$ i $< /h >$.

Primjer 1:

```
<t> Godine 1979., prvi slučajevi AIDS-a pojavili su se
 u New Yorku. Slučajevi bolesti koje se nikad nisu
vidjeli prije 1979. godine, ali se sada smatraju
smrtonosniji nego Kuga, nastavile su se pojavljati
u Kaliforniji i New Yorku, gotovo isključivo
u homoseksualnim muškarcima. </t>
<h> AIDS se širi u Africi. </h>
```

U ovom primjeru vidimo da je tekst puno duži od hipoteze i pošto je hipoteza tako kratka i od 5 riječi 4 postoje u tekstu, modeli su prepoznali logičku posljedicu iako ona ne postoji. Ako bi u hipotezi promijenili jedu riječ, Afrika u Amerika ili New York, logička posljedica bi bila prepoznata.

Primjer 2:

```
<t> Na izborima svibanj 2005 Michael Howard nije
uspio izbaciti vladu laburista, iako su konzervativci
dobili 33 mjesta, igrajući najznačajniju ulogu u
smanjenju laburističku većinu sa 167 na 66. </t>
<h> U svibnju 2005. na izborima konzervativci su
dobili 33 zastupničkih mjesta. </h>
```

Ovaj primjer su označivači označili kao negativan, no svi naši sustavi označili su kao pozitivan. Ovaj primjer je izrazito težak za osobu da razumije. Radite se o tome da su konzervativci povećali broj zastupničkih mjesta za 33, a ne da su dobili ukupno 33 mjesta. Kada bi eksplicitno rečenica glasila da su dobili 33 mjesta više nego na prošlim izborima, moguće je da bi sustav korektno označio ovaj par. Moramo primijetiti i da je velik dio hipoteze *h* sadržan u tekstu *t*. Označivači su koristili svoje opće znanje, te shvatili da ne može biti tako malo zastupničkih mjesta nego je to povećanje od 33.

```
<t> U povijesti umjetnosti, prapovijesna umjetnost
je sva umjetnost proizvedena u preliteralnim
```

```
kulturama (prapovijest), počevši negdje u vrlo
kasnoj geološkoj povijesti. </t>
<h> Prapovijesne umjetnosti otkrivene su u
Južnoj Africi. </h>
```

Ovdje logička posljedica nije prepoznata, no modeli su je prepoznali, vjerojatno zbog čestog ponavljanja oblika riječi *prapovijest* i *umjetnost*.

```
<t> Gabriel Garcia Marquez je autor "Sto godina samoće",
"Jesen patrijarha" i drugih romana. </t>
<h> Gabriel Garcia Marquez je romanopisac i pisac. </h>
```

Kao zadnji primjer uzeli smo tipičan nedostatak znanja o jeziku, odnosno semantičkoga znanja. Očito je da je logička posljedica postoji, no naši modeli ju nisu mogli pronaći zbog nedostatka znanja da je autor sinonim za pisac, te da ako je osoba autor romana da je tada romanopisac.

6.2. Usporedba s postojećim metodama

Zbog korištenja podskupa RTE3, a ne cijelog skupa s amaterskim prijevodom ne možemo napravit egzaktnu usporedbu sa postojećim sustavima, no prokomentirati ćemo njihove rezultate. Najrealnije usporedba je ona s rezultatima sustava dobivena u (Passonneau et al., 2005) na kojem se bazira i naš sustav. Sustav je testiran sa skupom podataka RTE3, te je bio prijavljen na treće natjecanje u prepoznavanju logičke posljedice u tekstovima. Za usporedbu ćemo koristiti točnost sustava oba sustava, zanemarujući preciznost i odziv. Najgori rezultati su kao i u ovom sustavu na skupini IE 0.4950, a dva najbolja su QA 0.76 i IR 0.6450. Sveukupna točnost (prosječna točnost skupina) je 0.6238, što je relativno slično ovog sustava (0.64). Kada gledamo sveukupno natjecanje 2007. godine u kojoj su sudjelovali 26 timova, možemo reći da je naš sustav prosječan. Točnost natjecateljskih sustava bila je od 49% do 80%, a većina je imala točnost od 59% do 66%. Ako bismo uzimali u obzir rezultate modela 3 sa četiri klasifikatora, sustav bi bio među boljim rezultatima (0.7907). Za razliku od puno drugih sustava, jedina semantička značajka modela je značajka WordNet. Za bolje rezultate potrebno je koristiti više semantičkih značajki, naravno uz trenutno korištene, te nadograditi hrvatski WordNet sa još primjera.

7. Zaključak

Prepoznavanje logičke posljedice u tekstovima jednostavan je zadatak za ljude, no napraviti računalni sustav za taj problem je iznimno kompliciranо. Kada bi računalni sustav mogao automatski prepoznati logičku posljedicu, tada bi takav sustav imao sposobnost prepoznavanja logičke posljedice. Takva sposobnost bi bila korak naprijed ka sustavima koji imaju sposobnost razumjeti tj. pročitati tekst. Ako imamo tekst *t* i hipotezu *h*, problem je pronaći je li hipoteza *h* logička posljedica teksta *t*. Ulaz u sustav su t-h (tekst - hipoteza) parovi, a izlaz je odluka je li *h* logička posljedica *t* ili nije. Kada se rade takvi sustavi najprije je potrebno prikupiti i označiti reprezentativan skup podataka, te jednoznačno odrediti načine evaluacije tih sustava. Prikupljanje podataka najčešće se radi kroz postojeće sustave obrade prirodnog jezika, a anotaciju rade stručnjaci s poštivanjem određenih standarda i pravila. Razni stručnjaci imali su različita viđenja ovog problema, pa tako i različit pristup pri njegovom rješavanju. U ovom radu prikazan je jedan sustav za prepoznavanje logičke posljedice u tekstovima na hrvatskome jeziku koji se bazira na prikazivanju *t–h* para kao vektor realnih brojeva odnosno niz značajki. Značajke su uzete uz pomoć različitih prikazivanja *t–h* para korištenjem alata za obradu hrvatskog jezika, računanju sličnosti i hrvatskog WordNeta. Prikupljene i odabrane značajke čine ulaz binarnog linearnog SVM klasifikatora, čiji nam izlaz daje odgovor na pitanje: "Je li hipoteza logička posljedica teksta?". Rezultati sustava su zadovoljavajući i prikazali smo kako se može koristiti hrvatski WordNet u prikupljanju značajki, te da su te značajke povećale točnost ovog sustava. Za unaprjeđenje sustava potrebno je najprije proširiti skup podataka i hrvatski WordNet, a zatim dodati i analizirati razne semantičke i probabilističke značajke. Iako prepoznavanje logičke posljedice nije zadatak određen samo na jednoj domeni, dodavanjem značajki koje su specifične za određenu domenu (na primjer frekvencija ponavljanje fraza ili riječi u korpusu) bi moglo poboljšati (ali i pogoršati) rezultate, te bi bilo poželjno takve značajke testirati.

LITERATURA

Željko Agić, Nikola Ljubešić, i Danijela Merkler. Lemmatization and morphosyntactic tagging of croatian and serbian. U *Proceedings of ACL*, 2013.

Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, i Idan Szpektor. The second pascal recognising textual entailment challenge. 2006.

Božo Bekavac, Krešimir Šojat, i Marko Tadić. Zašto nam treba hrvatski wordnet? *Semantika prirodnog jezika i metajezik semantike*, 2005.

Johan Bos i Katja Markert. Combining shallow and deep nlp methods for recognizing textual entailment. U *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, stranice 65–68, 2005a.

Johan Bos i Katja Markert. Recognising textual entailment with logical inference. U *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, stranice 628–635. Association for Computational Linguistics, 2005b.

BE Bosner, IM Guyon, i VN Vapnik. A training algorithm for optimal margin classifier. U *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, stranice 144–152, 1992.

Corinna Cortes i Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3): 273–297, 1995.

Ido Dagan i Oren Glickman. Probabilistic textual entailment: Generic applied modeling of language variability. 2004.

Ido Dagan, Oren Glickman, i Bernardo Magnini. The pascal recognising textual entailment challenge. U *Machine Learning Challenges. Evaluating Predictive Uncer-*

tainty, Visual Object Classification, and Recognising Tectual Entailment, stranice 177–190. Springer, 2006.

Ido Dagan, Bill Dolan, Bernardo Magnini, Dan Roth, I Dagan, B Dolan, B Magnini, D Roth, IDO DAGAN, BILL DOLAN, et al. Recognizing textual entailment: Rational, evaluation and approaches–erratum. *Natural Language Engineering*, 16(1): 105, 2010.

Jasminka Dobša. *Text mining using concept indexing*. Doktorska disertacija, Fakultet elektrotehnike i računarstva, 2006.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, i Bill Dolan. The third pascal recognizing textual entailment challenge. U *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, stranice 1–9. Association for Computational Linguistics, 2007.

Oren Glickman, Ido Dagan, i Moshe Koppel. A probabilistic classification approach for lexical textual entailment. U *Proceedings of the National Conference On Artificial Intelligence*, svezak 20, stranica 1050. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005a.

Oren Glickman, Ido Dagan, i Moshe Koppel. Web based probabilistic textual entailment. 2005b.

Oren Glickman, Ido Dagan, i Moshe Koppel. A lexical alignment model for probabilistic textual entailment. U *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, stranice 287–298. Springer, 2006.

Andrew Hickl i Jeremy Bensley. A discourse commitment-based framework for recognizing textual entailment. U *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, stranice 171–176. Association for Computational Linguistics, 2007.

Prodromos Malakasiotis i Ion Androutsopoulos. Learning textual entailment using svms and string similarity measures. U *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, stranice 42–47. Association for Computational Linguistics, 2007.

Antoni Oliver. Wn-toolkit: Automatic generation of wordnets following the expand model. *Proceedings of the 7th Global WordNetConference, Tartu, Estonia*, 2014.

Sebastian Padó, TAEGIL NOH, Asher Stern, Rui Wang, i Roberto Zanoli. Design and realization of a modular architecture for textual entailment. *Natural Language Engineering*, 1(1):000–000.

Rebecca J Passonneau, Ani Nenkova, Kathleen McKeown, i Sergey Sigelman. Applying the pyramid method in duc 2005. U *Proceedings of the Document Understanding Conference (DUC 05), Vancouver, BC, Canada*, 2005.

Mark Sammons, VG Vinod Vydiswaran, i Dan Roth. Recognizing textual entailment. *Multilingual Natural Language Applications: From Theory to Practice. Prentice Hall, Jun*, 2011.

Eyal Shnarch, Ido Dagan, i Jacob Goldberger. A probabilistic lexical model for ranking textual inferences. U *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, stranice 237–245. Association for Computational Linguistics, 2012.

Idan Szpektor, Eyal Shnarch, i Ido Dagan. Instance-based evaluation of entailment rule acquisition. U *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, svezak 45, stranica 456, 2007.

Prepoznavanje logičke posljedice u tekstovima na hrvatskome jeziku

Sažetak

Logička posljedica u tekstovima jest relacija koja vrijedi između dva fragmenata teksta ako istinitost prvog fragmenta implicira istinitost drugog. U ovom radu prikazali smo sustav prepoznavanje logičke posljedice u tekstovima na hrvatskome jeziku, baziran na strojnom učenju koristeći binarni klasifikator. Glavna tematika je prikupljanje i odabir značajki. Prikazan je evaluacija i usporedba ovog sustava sa postojećim sustavima.

Ključne riječi: logička posljedica, obrada prirodnog jezika, strojno učenje, SVM , hrvatski WordNet

Recognizing Textual Entailment in Croatian Texts

Abstract

Textual entailment (TE) in natural language processing is a directional relation between text fragments. The relation holds whenever the truth of one text fragment follows from another text. In this thesis, we presented system for recognizing textual entailment in Croatian texts based on machine learning using binary classificator. The main subject is feature extraction and selection. Evaluation of the system and comparison to other systems is shown.

Keywords: textual entailment, natural language processing, machine learning, SVM, Croatian WordNet