SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 122

RAZGRANIČAVANJE VIŠEZNAČNOSTI RIJEČI UPORABOM ALGORITMA MAKSIMIZACIJE OČEKIVANJA

Matea Biočić

Zagreb, srpanj 2010.

Zahvaljujem se svojoj obitelji, posebno Hrvoju na bekonačnom strpljenju i Gabrijeli na veselim danima i mirnim noćima koje su mi omogućile da uspješno završim studij.

Mentorici prof. dr. sc. Bojani Dalbelo Bašić zahvaljujem na razumijevanju i strpljenu, a dr. sc. Janu Šnajderu na pruženoj pomoći i korisnim savjetima tokom izrade ovog rada.

Sadržaj

1. Uvod

Razrješavanje i razgraničavanje višeznačnosti jedan su od glavnih problema obrade prirodnog jezika (engl. *natural language processing, NLP*). Obrada prirodnog jezika je područje računarske znanosti koja se bavi povezanosti računala i prirodnog jezika, a često se klasificira kao dio umjetne inteligencije. Problemi koje područje obrade prirodnog jezika pokušava riješiti su: strojno prevođenje, dohvat informacija, OCR (engl. *optical character recognition*), automatsko generiranje sažetaka, prepoznavanje govornog jezika i mnogi drugi.

U svakom jeziku postoje riječ i koje nose više od jednog značenja – višeznačnice. Ljudi nauče razlikovati pojedina značenja riječi prema kontekstu u kojem se on pojavljuje. U sljedećem primjeru lako ćemo razlikovati dva značenja riječi grad:

1. Idem u *grad* Zagreb na kavu.

2. *Grad* je uništio sve usjeve.

Očito je da se u prvoj rečenici govori o naseljenom mjestu, a u drugoj o vremenskoj pojavi (tuči). Računala nemaju mogućnost razrješavanja višeznačnosti bez posebnih algoritama kojima analiziraju kontekst u kojem se riječ nalazi i na temelju toga donose odluku. Taj problem je izražen kod strojnog prevođenja kada računalo mora odlučiti kako prevesti određenu riječ na drugi jezik.

Automatsko razrješavanje višeznačnosti može se podijeliti na dva manja problema. Prvi je automatsko razgraničavanje višeznačnosti, odnosno mogućnost da računalo prepozna da se radi o različitom značenju riječi a da ne ustanovi o kojim značenjima se radi. Drugi problem je pridjeljivanje smisla pojedinoj riječi.

Cilj ovog rada jest razviti sustav koji će uspješno razgraničavati višeznačnost riječi hrvatskog jezika uz pomoć rječnika i neoznačenog korpusa algoritmom maksimizacije očekivanja. Rezultati kasnije mogu biti iskorišteni u drugim sustavima za obradu jezika i pretraživanje informacija, kao što su na primjer: razvrstavanje

rezultata pretrage na internetskim tražilicama u kategorije po značenju riječi, kod strojnog prevođenja i sl.

U idućem poglavlju dan je kratak pregled srodnih radova i istraživanja na području razrješavanja i razgraničavanja višeznačnosti. U poglavlju 3 objašnjena je osnovna ideja iskorištavanja Bayesovog poučka kod problema razgraničavanja višeznačnosti. Poglavlje 4 objašnjava algoritam maksimizacije očekivanja, a u poglavlju 5 je ukratko opisana programska izvedba i priprema korpusa. U poglavlju 6 nalaze se opis i rezultati testiranja. Upute za korištenje i detaljni rezultati testiranja nalaze se u dodatku.

2. Pregled područja

Pristupi za razrješavanje višeznačnosti mogu se razvrstati u tri glavne kategorije: nadzirani pristupi (engl. *supervised techniques*), pristupi temeljeni na rječniku (engl. *dictionary-based techniques*) i nenadzirani pristupi (engl. *unsupervised techniques*).

Nadzirani pristupi zahtijevaju semantički označeni korpus u kojemu je svaka višeznačna riječ ispravno označena. Moguća značenja riječi su definirana skupom semantičkih oznaka prisutnih u korpusu. Kontekst je prozor oko riječi i informatora (engl. *informant*), riječi koje pripadaju u taj kontekstni prozor.

Pristupi temeljeni na rječniku su slični kao nadzirani pristup, ali koriste neoznačeni korpus i rječnik kao dodatni izvor informacija za definiranje značenja. U jednom od algoritama značenje višeznačne riječi određuje se prema broju preklapanja riječi iz konteksta i definicije u rječniku.

Jedan od prvih radova koji ne zahtijeva vanjske izvore znanja i svrstava se u nenadzirano učenje objavio je Schütze [8], a zasnovan je na (engl. *context-group discrimination*). Njegova metoda svrstava višeznačne riječi u skupine (engl. *clusters*) temeljeno na supojavljivanju drugog reda (engl. *second-order co-occurrence*): konteksti višeznačnih riječi su slični ako su i riječi koje se pojavljuju unutar konteksta slične. Riječi, konteksti i skupine predstavljani su u visokodimenzionalnom vektorskom prostoru.

Yarowsky u svome radu iskorištava činjenice da se značenje višeznačne riječi može dokučiti s obzirom na kontekst koji se nalazi i da je unutra jednog dokumenta obično samo jedno značenje pojedine riječi [11]. Pokazao je značajnu prednost ovakvog nenadziranog pristupa nad nadziranim. Za razliku od Shutzea, on koristi ručno označeni korpus.

Jedini rad koji je vezan uz hrvatski jezik vezan je uz razrješavanje višeznačnosti uz pomoć označenog korpusa [3]. Pokazalo se da je za uspješnost bitan neposredni kontekst višeznačne riječi (1–5 riječi), da se sa jako velikom vjerojatnošću (približno 90%) može pretpostaviti da se u jednom dokumentu pojavljuje samo jedno

značenje te da ako se u obzir uzimaju granice rečenica nema znatnog poboljšanja točnosti.

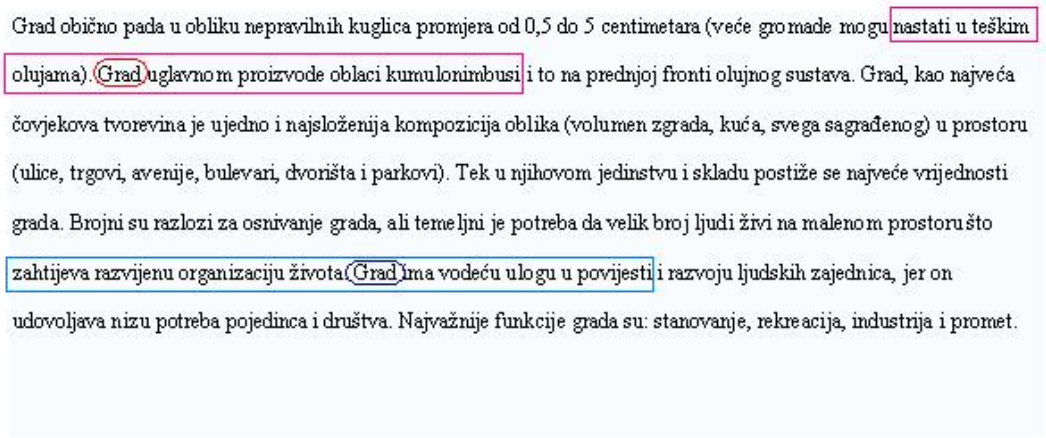
Shinnou i Sasaki [9] pokušavaju usavršiti algoritam maksimizaciije očekivanja odabirom pravog broja iteracija. Primjenom te metode uspjeli su postići veću točnost nego osnovnim algoritmom EM. Pregled postupaka s kojima su eksperimentirali i postignutih rezultata dani su u tablici 2.1.

Tablica 2.1: Pregled važnijih radova

Autori	Referenca	Naziv algoritma	Jezik	Uspješnost
Shütze	[8]	CGD	engl.	83 – 91%
Bakarić, Njavro, Ljubešić	[3]	Bayes	hrv.	60%
Marneffe, Dupont	[5]	Bayes, EM	engl.	63–68%
Marneffe, Dupont	[5]	vektorski model	engl.	64–75%
Levin, Shariffi Ball	[7]	LSA	engl.	60–80%
Shinnou, Sasaki	[9]	Bayes	jap.	76.78%
Shinnou, Sasaki	[9]	EM	jap.	73.56%
Shinnou, Sasaki	[9]	CV-EM	jap.	77.88%
Shinnou, Sasaki	[9]	CV-EM2	jap.	78.56%
Shinnou, Sasaki	[9]	EM najbolji	jap.	79.64%
Yarowsky	[11]	bootstraping	engl.	96%

3. Bayesova teorija odlučivanja u razgraničavanju višeznačnosti

Razgraničavanje višeznačnosti riječi zasniva se na analizi konteksta višeznačne riječi w. Kontekst c je okvir oko višeznačne riječi proizvoljne duljine. Dva konteksta riječi *grad* veličine 4 riječi označena su na slici 3.1. U ovom primjeru kontekst prelazi granice rečenice. Alternativni način, koji je i korišten u ovom radu jer da se poštuje rečenica te se kontekst ne proširuje na sljedeću rečenicu. Riječi koje se nalaze unutar konteksta nazivaju se informatori.



Slika 3.1: Primjer određivanja konteksta

U nastavku će se koristiti sljedeća notacija [5]:

- w označava višeznačnu riječ
- $s_1, \cdots, s_k$ označavaju k mogućih značenja riječi w
- $c_1, \cdots, c_i$ označavaju kontekste i instanci riječi w u korpusu učenja
- $v_1, \cdots, v_j$ označavaju j mogućih informatora

Prema Bayesovoj teoriji vjerojatnosti, razgraničavanje višeznačnosti može prikazati

kao izračun vrijednosti k koja maksimizira posteriornu vjerojatnost $P(s_k|c)$ značenja s_k promatranog u kontekstu c :

$$k = \arg\max_k P(s_k|c) = \arg\max_k P(c|s_k)P(s_k) \tag{3.1}$$

gdje je $P(c|s_k)$ vjerojatnost da se kontekstu c pridruži značenje s_k , a $P(s_k)$ označava vjerojatnost da se pojavi značenje s_k.

Ako gledamo kontekst c_i riječi w , koji je kontekstni prozor oko riječi w u korpusu učenja, onda su informatori kontekstne riječi v_j . Kontekstne riječi pripadaju kontekstu c_i . Za razliku od kontekstnih riječi imamo i zaustavne riječi (engl. *stop words*). To su najčešće prilozi, prijedlozi, veznici, članovi i ostale riječi koje se često pojavljuju u tekstu, ali same ne nose značenje.

Vjerojatnost konteksta se izražava formulom:

$$P(c_i|s_k) = P(\{v_j \in c_i\}\,|s_k) \tag{3.2}$$

Prema Bayesovoj pretpostavci, kontekstne riječi su nezavisne pa se formula može zapisati na drugi način:

$$P(\{v_j \in c_i\}\,|s_k) = \prod_{v_j \in c_i} P(v_j|s_k) \tag{3.3}$$

Skup parametara θ za svaku riječ w se sastoji od JK vjerojatnosti $P(v_j|s_k)$ i K vjerojatnosti $P(s_k)$. Ovi parametri mogu se procijeniti tako da maksimiziraju vjerojatnost korpusa učenja. Kako korpus nije označen, koristimo algoritam EM (Expectation-Maximization).

Algoritam EM je iterativni postupak koji počevši od inicijalnih parametara θ_0 u svakom koraku ponovno procjenjuje vrijednosti parametara kako bi se povećala vjerodostojnost podataka (engl. *log-likelihood, LL*). LL konteksta I unutar korpusa učenja se definira kao:

$$\begin{aligned} LL(\{c_1, ..., c_i\}\,|\theta) = \log \prod_{i=1}^{I} P(c_i) = \prod_{i=1}^{I} \log P(c_i) = \\ = \sum_{i=1}^{I} \log \sum_{k=1}^{K} P(c_i|s_k)P(s_k) \\ = \sum_{i=1}^{I} \log \sum_{k=1}^{K} P(s_k) \prod_{v_j \in c_i} (v_j|s_k) \end{aligned} \tag{3.4}$$

U praksi vjerojatnosti $P(v_j|s_k)$ se inicijaliziraju slučajno tako da zadovoljavaju ograničenja $\sum_{j=1}^{J} P(v_j|s_k) = 1$ i $1 \leq k \leq K$.

Za $P(s_k)$ se uzima uniformna raspodjela, dakle: $P(s_k) = \frac{1}{K}$.

Slijedi iterativno ponavljanje algoritma maksimizacije očekivanja dok maksimalna vjerodostojnost raste. Algoritam je detaljno objašnjen u sljedećem poglavlju.

4. Algoritam maksimizacije očekivanja

Algoritam maksimizacije očekivanja (engl. *Expectation Maximization algorithm, EM*) je učinkovit iterativan postupak za izračun procjene maksimalne vjerodostojnosti (engl. *maximum likelihood*) u slučaju nepotpunih ili skrivenih podataka. [4, 5, 6]

4.1. Maksimalna vjerodostojnost

Imamo funkciju gustoće $p(x|\theta)$ koja je određena skupom parametara θ i skup podataka veličine N. Pretpostavljamo da su ti vektori (npr. $X = x_1, \ldots, x_N$) nezavisni i jednoliko raspodijeljeni s razdiobom p. Gustoća za takve uzorke je:

$$p(X|\theta) = \prod_{i=1}^{N} p(x_i|\theta) = LL(\theta|X) \tag{4.1}$$

Funkcija $LL(\theta|Xi)$ se naziva funkcija vjerodostojnosti. Promatra se kao funkcija parametara θ gdje su podaci X fiksni. Kod problema maksimalne vjerodostojnosti cilj nam je pronaći skup parametara θ koji maksimizira LL.

$$\theta^* = \arg\max_{\theta} LL(\theta|X). \tag{4.2}$$

Ponekad se maksimizira $\log LL(\theta|X)$ jer je analitički jednostavnije.

4.2. Općeniti algoritam maksimizacije očekivanja

Algoritam maksimizacije očekivanja je opća metoda za pronalazak procjene najveće vjerodostojnosti parametara neke distribucije kada su podaci nepotpuni ili nedostaju.

Dvije su glavne primjene ovog algoritma. Prva je kada podaci uistinu imaju vrijednosti koje nedostaju, a druga kada je analitičkim putem nemoguće naći funkciju vjerodostojnosti pa se pretpostavlja postojanje dodatnih parametara. Kao i prije pretpostavljamo da prikupljeni podaci X pripadaju nekoj distribuciji. Nazivamo ih nepotpunim podacima.

Pretpostavljamo i da postoji potpun skup podataka $Z = (X|Y)$ i pretpostavljamo zajedničku funkciju gustoće:

$$p(z|\theta) = p(x, y|\theta) = p(y|x, \theta)p(x|\theta) \tag{4.3}$$

Zajednička funkcija gustoće proizlazi iz marginalne funkcije gustoće $p(x|\theta)$ i pretpostavke skrivenih varijabli i vrijednosti parametara. U preostalim slučajevima moramo pretpostaviti vezu između podataka koji nedostaju i promatranih podataka.

Sa ovom novom funkcijom gustoće možemo definirati i novu funkciju vjerodostojnosti, $LL(\theta|Z) = LL(\theta|X, Y) = p(X, Y|\theta)$, koja se naziva vjerodostojnost potpunih podataka (engl. *complete-data likelihood*). To je u biti slučajna varijabla budući da su podaci koji nedostaju Y nepoznati i nasumični. Možemo smisliti $LL(\theta|X, Y) = h_{x,\theta}(Y)$ za neku funkciju $h_{x,\theta}(\cdot)$ gdje su X i θ konstantni a Y je slučajna varijabla. Originalna sličnost $LL(\theta|X)$ je funkcija vjerodostojnosti nepotpunih podataka (engl. *incomplete-data likelihood function*).

Algoritam maksimizacije očekivanja prvo pronalazi očekivanu vrijednost logaritma vjerodostojnosti potpunih podataka (engl. *complete-data log-likelihood*) $\log p(X; Y|\theta)$ uzimajući u obzir i nepoznate podatke Y, promatrane podatke X i trenutne procijene parametara. Definiramo:

$$Q(\theta, \theta^{(i-1)}) = E[\log p(X, Y|\theta)|X, \theta^{(i-1)}] \tag{4.4}$$

gdje su $\theta^{(i-1)}$ trenutne procijene vrijednosti parametara koje smo koristili za ocijeniti očekivanje i θ su novi parametri koje optimizirali kako bi povećali Q.

Evaluacija ovog očekivanja naziva se korak E algoritma. Treba obratiti pažnju na značenje argumenata u funkciji $Q(\theta, \theta')$. Prvi argument θ odgovara parametrima koji će biti optimizirani u pokušaju maksimizacije vjerodostojnosti. Drugi argument θ' odgovara parametrima koje koristimo za vrednovanje očekivanja.

Drugi korak, korak M, maksimizira očekivanje koje smo izračunali u prvom koraku:

$$\theta^{(i)} = \arg\max_{\theta} Q(\theta, \theta^{(i-1)}). \tag{4.5}$$

Dva koraka se ponavljaju prema potrebi. Svaka iteracija garantirano povećava logaritam vjerodostojnosti i algoritam garantirano konvergira u lokalni maksimum funkcije vjerodostojnosti.

4.3. Algoritam primijenjen na problemu razrješavanja i razgraničavanja višeznačnosti

Kod prilagođavanja algoritma određenom problemu potrebno je definirati skup parametara θ. Skup parametara θ za svaku višeznačnu riječ w se sastoji od JK vjerojatnosti $P(v_j|s_k)$ i K vjerojatnosti $P(s_k)$.

Vjerojatnosti $P(v_j|s_k)$ se inicijaliziraju slučajno tako da zadovoljavaju ograničenja $\sum_{j=1}^{J} P(v_j|s_k) = 1$ i $1 \leq k \leq K$. Za $P(s_k)$ se uzima uniformna raspodjela, dakle: $P(s_k) = \frac{1}{K}$.

U prvom koraku (korak E) izračunava se h_{ik}, procjena posteriorne vjerojatnosti da je značenje s_k u kontekstu c_i.

$$h_{ik} = \frac{P(s_k)P(c_i|s_k)}{\sum_{l=1}^{K} P(s_l)P(c_i|s_l)} = \frac{P(s_k)\prod_{v_j \in c_i} P(v_j|s_k)}{\sum_{l=1}^{K}\left(P(s_l)\prod_{v_j \in c_i} P(v_j|s_l)\right)} \tag{4.6}$$

U drugom koraku (korak M) ponovno se računaju $P(v_j|s_k)$ i $P(s_k)$ kako mi se maksimizirala vjerodostojnost.

$$P(v_j|s_k) = \frac{\sum_{\{c_i:v_j \in c_i\}} h_{ik}}{\sum_{j=1}^{K}\sum_{\{c_i:v_j \in c_i\}} h_{ik}}, \tag{4.7}$$

gdje je $\sum_{\{c_i:v_j \in c_i\}} h_{ik}$ zbraja po svim kontekstima c_i u kojima se pojavljuje v_j.

$$P(s_k) = \frac{\sum_{i=1}^{I} h_{ik}}{\sum_{k=1}^{K}\sum_{i=1}^{I} h_{ik}} = \frac{\sum_{i=1}^{I} h_{ik}}{I} \tag{4.8}$$

Jednom kada se izračunaju parametri na korpusu za učenje, značenje nove instance višeznačne riječi w može se izračunati s obzirom na kontekst c. Konačno pravilo je:

$$\hat{k} = \arg\max_k P(s_k|c) = \arg\max_k P(s_k) \prod_{v_j \in c} P(v_j|s_k) = \\ = \arg\max_k \log P(s_k) + \sum_{v_j \in c} \log P(v_j|s_k) \tag{4.9}$$

Uvjet zaustavljanja

Algoritam se zaustavlja kada maksimalna vjerodostojnost 3.4 prestane rasti [5]. Alternativno, može se zaustaviti nakon određenog broja koraka ili kada promjena vjerojatnosti bude dovoljno mala kao što je napravljeno u radu [9].

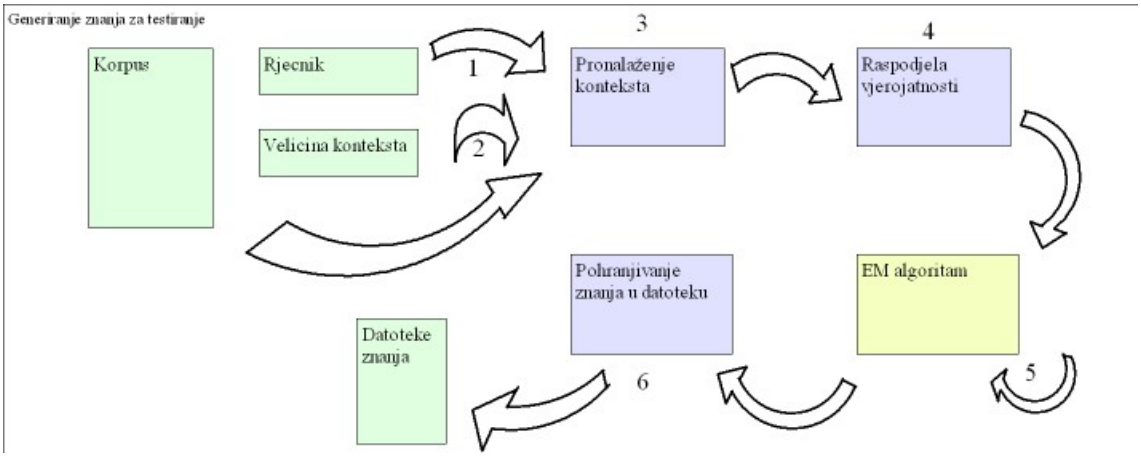
5. Programska izvedba

Algoritam je u potpunosti implementiran u programskom jeziku C#. Izabran je zbog potrebe za objektnim jezikom, a između ostalih objektnih jezika je izabran zbog osobne sklonosti.

Algoritam ima dva glavna načina rada: učenje i izvođenje, koji su detaljnije opisani u nastavku.

5.1. Učenje

Kako bi mogao učiti algoritam na ulazu prima datoteke u kojima se nalaze rječnik i korpus. Za svaku riječ iz rječnika potrebno je pokrenuti učenje na korpusu. Algoritam uči pomoću dva koraka ranije opisana u nastavku. Učenje se zaustavlja nakon što maksimalna vjerodostojnost prestane rasti. Grafički prikaz postupka učenja nalazi se na slici 5.1.



Slika 5.1: Shematski prikaz učenja

1. Iz rječnika preuzeti riječ i broj značenja višeznačne riječi
2. Učitati veličinu konteksta

3. Pronalazak svih konteksta u korpusu

4. Nasumično inicijaliziranje vjerojatnosti $P(v_j|s_k)$ i $P(s_k)$

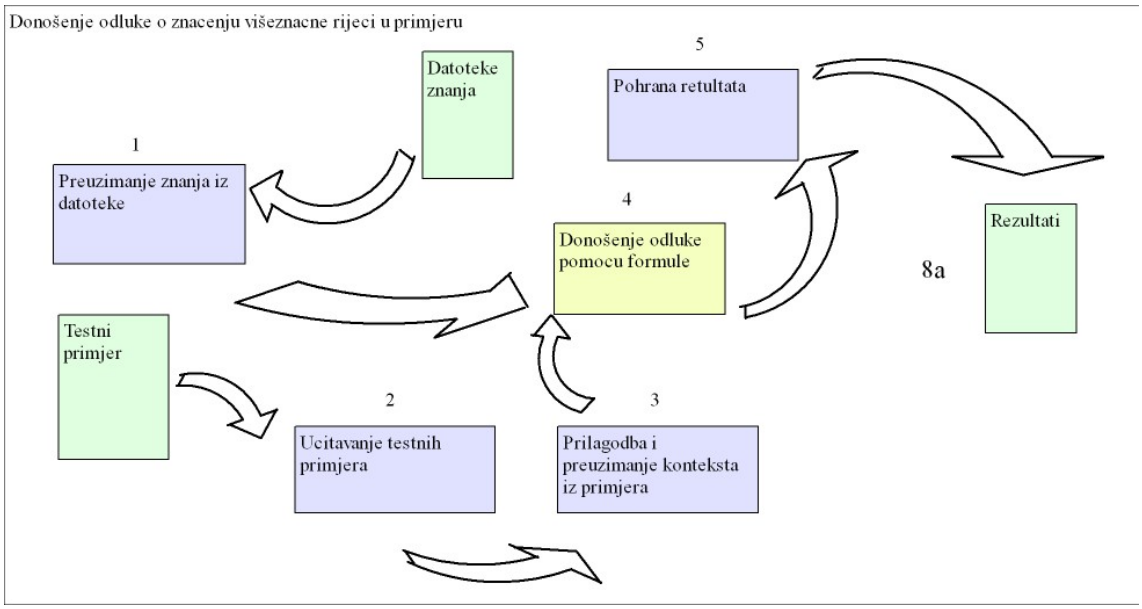
5. Iterativno ponavljanje E i M koraka dok maksimalna vjerodostojnost raste

6. Pohranjivanje rezultata (dobivenih vjerojatnosti) u datoteku

Datoteke u koje se pohranjuju rezultati, odnosno vjerojatnosto dobivene algoritmom u daljnjem tekstu će se radi jednostavnosti nazivati datoteke znanja.

5.2. Izvođenje

Za izvođenje algoritam treba na ulazu primiti testne primjere i datoteku znanja za određenu veličinu konteksta. Grafički prikaz izvođenja nalazi se na slici 5.2.



Slika 5.2: Shematski prikaz izvođenja

1. Za određenu višeznačnu riječ i veličinu konteksta čitamo rezultate učenje iz datoteke

2. Učitavanje testnih primjera

3. Pronalazak višeznačne riječi i određivanje konteksta u primjerima

4. Prema pravilu odlučivanja odrediti značenje višeznačne riječi

5. Prikaz i pohrana rezultata

5.3. Razredi i funkcije

5.3.1. Razredi

U razredu Neighbors pohranjuju se podaci za svaku riječ v_j koja se nalazi u kontekstu višeznačne riječi w unutar korpusa. Podaci sadrže riječ v_j, broj spominjanja te riječi unutar korpusa, vjerojatnosti pojavljivanja riječi v_j u određenom značenju ($P(v_j|s_k)$) te indekse konteksta u kojima se riječ pojavljuje. Instance razreda pohranjuju se u listu. Kod razreda prikazan je u nastavku.

Razred Neighbors

```
public class Neighbors {
static int number = 1;
public string neighborName;
public int timesMentiond;
public double[] probabilities = new double[number];
public List<int> inWhichContexts = new List<int>();
public Neighbors(int numberOfSenses)
{
number = numberOfSenses;
}
}
```

Razred Senses pohranjuje vjerojatnosti pojedinog značenja te definicije iz rječnika. Kod razreda prikazan je u nastavku.

Razred Senses

```
public class Sense {
public string sense;
public double Sk;
}
```

Razred Contexts predstavlja jedan kontekst u korpusu ili primjeru. Sadrži listu riječi koje se nalaze u kontekstu te funkciju koja dodaje riječi u tu listu. Instance razreda se razvrstavaju u dvije liste ovisno da li je riječ o kontekstu unutar korpusa ili primjera. Kod je dan u nastavku:

Razred Contexts

```
public class Contexts{
public List<string> oneContext=new List<string>();
public void AddTooneContext(string word){
oneContext.Add(word);
}
}
```

Razredi EM i Word sadrže velik broj funkcija za pripremu teksta za obradu, pohrane i obrade konteksta, samog EM algoritma, algoritma za donošenje konačne odluke i niza pomoćnih funkcija. Najvažnije navedene su i detaljnije objašnjene u nastavku.

5.3.2. Funkcije

Popis važnijih funkcija unutar EM razreda dan je u nastavku:

Najvažnije funkcije EM razreda

```
public void GetContexts (string word, int contextSize){...}
public void GetCorpusContextsSentence
(string word, int contextSize){...}
public void EStep(){...}
public void MStep(){...}
public void Bayes(){...}
static void ExpelPunctuation(string file){...}
static void MakeAllLowerCase(string file){...}
static void ExpelStopWords(string file){...}
```

Funkcija GetContexts preuzima kontekste oko višeznačne riječi iz testnih primjera, a funkcija GetCorpusContextsSentence preuzima kontekste iz korpusa. Obje funkcije za granicu konteksta imaju i rečenicu i kraj dokumenta.

EStep() i MStep() preslikavaju E i M korak algoritma kako je opisano u poglavlju 4.3..

Bayes() donosi konačnu odluku o razgraničavanju prema formuli 4.9.

Preostale funkcije pripremaju korpus i testne primjere za obradu. Korpus je obrađen samo jednom jer se služimo pretpostavkom da se ne će mijenjati, dok se testni primjeri pri svakom pokretanju ponovno obrade jer su podložni promjenama. Funkcija ExpelPunctuation izbacuje interpunkcijske znakove i brojke, MakeAllLowerCase, a ExpelStopWords izbacuje stop riječi.

Popis važnijih funkcija unutar Word razreda dan je u nastavku:

Najvažnije funkcije Word razreda

```
public void StoreNeighbours
(string word, List<Contexts> corpusContexts){...}
public void RenewProbabilities(){...}
public void StoreKnowlegeToFile(string word){...}
public void GetKnowlegeFromFile(string word){...}
```

Funkcija StoreNeighbours se poziva nakon funkcija GetCorpusContextsSentence i GetContexts te sve riječi unutar konteksta sprema u listu razreda Neighbors.

RenewProbabilities nakon svake iteracije algoritma EM obnavlja vjerojatnosti zapisane u listi razreda Neighbors.

Posljednje dvije funkcije StoreKnowlegeToFile i GetKnowlegeFromFile primaju, odnosni dohvaćaju konačni rezultat algoritma iz datoteke.

6. Vrednovanje

6.1. Korpus

Korpus je skup tekstova pomoću kojih algoritam uči. Sadrži on-line arhiv dnevnika "Vjesnik" [2] iz razdoblja od 31. svibnja 1999. do 1. studenog 2009. Sastoji se od 276 231 članka, 4 466 178 rečenica, 118 955 051 riječi od čega je 1 101 802 različitih riječi. Veličina korpusa zapisanog u tekstnoj datoteci je 693 MB. Prije početka rada potrebno je prilagoditi korpus. Prilagodbe koje su napravljene su:

- ujednačavanje veličine slova
- izbacivanje interpunkcijskih znakova i brojeva
- izbacivanje slika
- izbacivanje stop riječi
- lematizacija

Za potrebu ovog rada nije bitno raditi razliku između riječi koje imaju malo i veliko početno slovo. Takvih višeznačnica, čije značenje ovisi o početnom slovu, nema puno i nad njima se neće vršiti testiranje. Zbog jednostavnosti sva slova će biti mala.

Zatim se izbacuju svi interpunkcijski znakovi. Jedini znak koji nam je od važnosti je znak za kraj rečenice, a kako je korpus priređen tako da svaka rečenica započinje u novom retku i njih možemo izbaciti. Brojevi isto tako ne nose nikakvu težinu kod razgraničavanja i razrješavanja višeznačnosti pa ih izbacujemo.

Prilikom preuzimanja članaka, preuzete su i poveznice na slike koje je bilo potrebno izbaciti.

U svakom jeziku postoji niz riječi koje ne nose posebno značenje. Takve riječi se u području pretraživanja informacija nazivaju zaustavnim riječima (engl. *stop words*). To su prijedlozi, prilozi, veznici i slično. Njih je potrebno izbaciti iz korpusa jer ne pridonose razgraničavanju višeznačnosti i mogu smanjiti točnost. Na popisu stop riječi nalazi se 2 036 riječi.

Zadnja priprema korpusa je lematizacija, odnosno svođenje riječi na normalizirani oblik (nominativ jednine za imenice i pridjeve, infinitiv za glagole). Lematizacija se provodi pomoću pribavljenog automatskog flektivnog leksikona [10].

Nakon obrade korpus sada ima 59 229 781 riječi, a broj različitih riječi je 673 560. Zapis u tekstnoj datoteci je sada 496 MB.

6.2. Rječnik

Za potrebe rada korišteno je elektroničko izdanje Anićevog Velikog rječnika hrvatskoga jezika [1]. Izabrano je 9 riječi za koje će algoritam naučiti vršiti razgraničavanje višeznačnosti. Uz samu riječ pohranjeno je i njeno značenje. Radi jednostavnosti i vremenske zahtjevnosti testiranja odabrano je samo 8 testnih riječi sa jednakom zastupljenošću riječi sa dva i tri značenja. Primjer zapisa za riječ *virus* dan je u nastavku:

> *virus* muški rod biol. parazitski mikroorganizam koji izaziva bolest, ob. nosi naziv prema bolesti koju uzrokuje *virus* muški rod inform. program koji inficira kompjuterske datoteke, ob. putem zaraženih disketa ili preko interneta, te izaziva štetne pojave

6.3. Primjeri za testiranje

Primjeri za testiranje su jednostavne rečenice u kojima se pojavljuje višeznačna riječ. Testiramo na dvije višeznačnice *virus* koja ima 2 značenja i *rak* koja ima tri značenja. Za riječi sa dva značenja imamo po 40 primjera po višeznačnici, a za riječi sa tri značenja 36 primjera po višeznačnici. Rečenice su razvrstane u četiri datoteke radi jednostavnijeg očitavanja rezultata. U svakoj datoteci podjednako je zastupljeno svako značenje višeznačnice. Primjer datoteke za riječ *virus* nalazi se u nastavku:

```
Kompjutorski virusi velika su pošast današnjice.
Ako se želimo zaštititi od virusa ne smijemo koristit USB na javnom
računalu.
Dobar način zaštite od kompjuterskih virusa je ne otvarati
elektroničku pošto od nepoznatih pošiljatelja.
Virus može nanijeti veliku štetu računalu.
```

```
Svako računalo može dobiti virus kada se spoji na Internet.
Tko je bolestan i ide u bolnicu može se zaraziti novim virusom.
Povišena temperatura je simptom virusa ili bakterijske upale.
Virus ptičje gripe je izazvao veliku paniku.
Govore da je svinjska gripa jako opasan virus koji se prenosi zrakom.
Virusi su opasniji od bakterija jer na njih ne djeluje antibiotik.
```

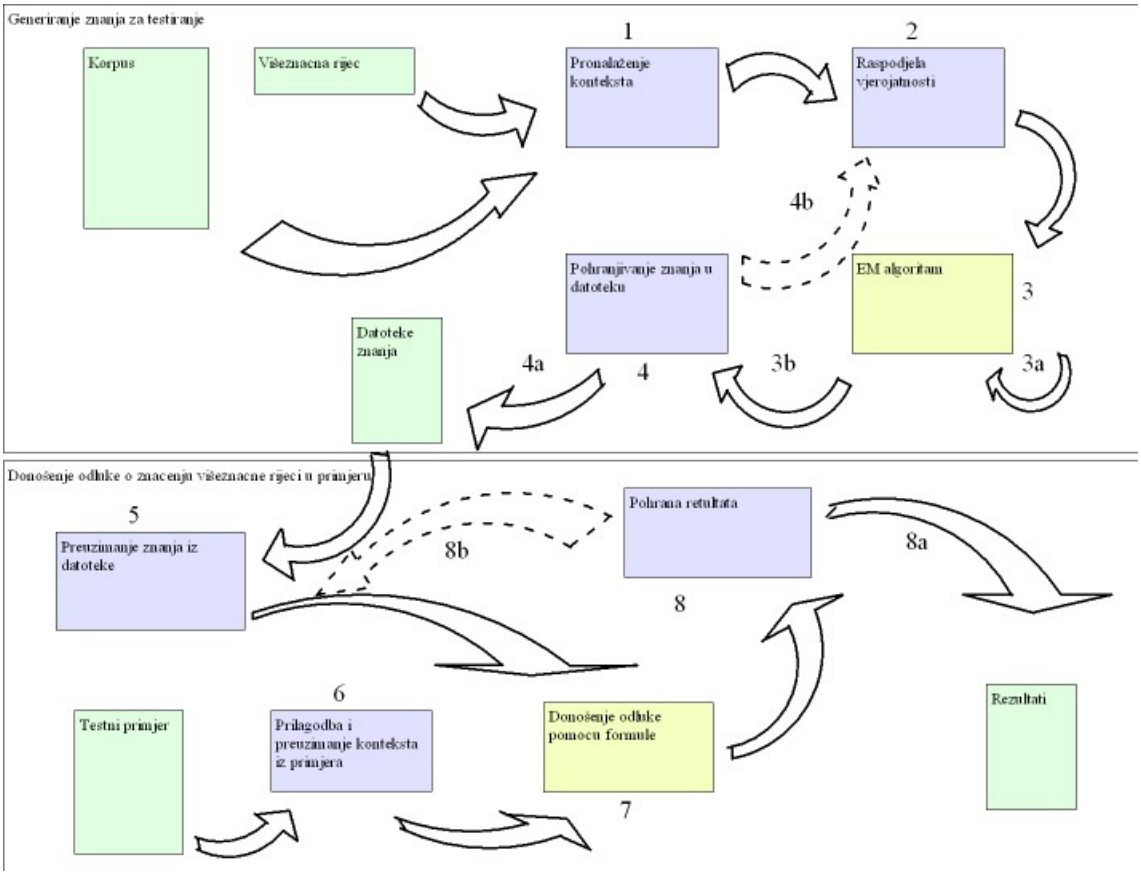
Unutar datoteke, riječi sa istim značenjem su grupirane kako bi se olakšala analiza rezultata testiranja. Nakon ove dvije osnovne riječi, testirano je još 9 riječi, a detalji se nalaze u poglavlju 6.6.. Svaka datoteka se pri učitavanju mora pripremiti na identičan način na koji smo pripremali korpus:

- ujednačavanje veličine slova
- izbacivanje interpunkcijskih znakova i brojeva
- izbacivanje stop riječi
- lematizacija

Jedina razlika je što korpus pripremamo jednom, dok se datoteke prilagođavaju pri svakom učitavanju jer možemo očekivati promjene u sadržaju.

6.4. Postupak testiranja

Prvo testiranje je napravljeno nad jednim primjerom višeznačnice s dva i jednim primjerom višeznačnice s tri značenja. Postupak testiranja grafički je prikazan na slici6.1.



Slika 6.1: Shematski prikaz testiranja

1. **Pronalaženje konteksta**: u korpusu se pronalaze konteksti višeznačne riječi 2. **Raspodjela vjerojatnosti**: svakoj riječi (za svaki kontekst) unutar konteksta dodjeljuje se nasumična vrijednost vjerojatnosti $P(v_j|s_k)$ i $P(s_k)$ 3. **Algoritam EM**: 3a. Računanje $P(v_j|s_k)$ i $P(s_k)$; ponavljanje algoritma sve dok maksimalna vjerodostojnost raste 3b. Kada se ustanovu maksimalna vjerodostojnost algoritam se zaustavlja 4. **Pohranjivanje znanja u datoteku**: 4a Vjerojatnosti dobivene algoritmom pohranjuju se u datoteku 4b. Ponovno nasumično generiranje vjerojatnosti i ponavljanje koraka 3 i 4 5. **Preuzimanje znanja iz datoteke**: preuzimanje vjerojatnosti dobivenih algoritmom EM iz datoteke 6. **Prilagodba i preuzimanje konteksta iz primjera**: Prilagodba primjera za testiranje i pronalaženje konteksta oko višeznačne riječi 7. **Donošenje odluke**

pomoću formule: uvrštavanjem vjerojatnosti u formulu 4.9 dolazi se do odluke o značenju riječi u rečenici 8. **Pohrana rezultata**: 8a pohrana rezultata u datoteku ili ispis na zaslon 8b. ponavljanje koraka 6-.8 za drugačije datoteke znanja

Generirano je po pet datoteka znanja za svaku višeznačnicu. Svaka od njih testirana je nad svim primjerima. Bitno je da se u generira više datoteka znanja i tek nakon testiranja možemo odlučiti koju ćemo u budućnosti koristiti jer rezultat algoritma ovisi o nasumično generiranim vjerojatnostima koje ulaze u algoritam i moguća su velika odstupanja u rezultatima jer algoritam može zaglaviti u lokalnom minimumu.

Cijeli postupak je ponavljan za pet različitih veličina konteksta (1–5 riječi). Na temelju ova dva primjera odabrana je fiksna veličina konteksta sa kojoj je preveden testiranje za ostale riječi.

Rezultati testiranja se nalaze u nastavku, a detaljni rezultati za prva dva testiranja nalaze se u dodacima A i B.

6.5. Provjera točnosti

Pretpostavimo da smo testirali točnost za višeznačnicu *virus* nad primjerima navedenim ranije i da smo dobili slijedeći rezultat:

```
Kompjutorski virusi velika su pošast današnjice.  novim virusom. (*)
Ako se želimo zaštititi od virusa ne smijemo      Povišena temperatura je simptom virusa
koristit USB na javnom računalu.                  ili bakterijske upale.
Govore da je svinjska gripa jako opasan virus     Virus ptičje gripe je izazvao veliku paniku.
koji se prenosi zrakom. (*)                       Dobar način zaštite od kompjutorskih virusa
Virus može nanijeti veliku štetu računalu.        je ne otvarati elektroničku pošto od nepoznatih
Svako računalo može dobiti virus kada se          pošiljatelja. (*)
spoji na Internet.                                Virusi su opasniji od bakterija jer na njih ne djeluje
Tko je bolestan i ide u bolnicu može se zaraziti  antibiotik.
```

Znamo da bi se u svakom stupcu trebale nalaziti rečenice sa pojedinim značenjem. Znakom (*) su označene rečenice koje su svrstane krivo, tj. značenje višeznačne riječi u toj rečenici ne odgovara značenju u većini rečenica u tom stupcu. Kako ne određujemo točno značenje riječi ne može predvidjeti u kojem stupcu će se naći koje značenje. Budući da su tri od deset rečenica krivo svrstane, možemo reći da je točnost za ovaj primjer 70%. Na sličan način određuje se točnost i za više od dva značenja riječi.

6.6. Rezultati testiranja

Testiranje je započeto sa riječi *virus*. Generirano je po četiri datoteke znanja za veličine konteksta 1–5 riječi. Svaka datoteka znanja okrenuta je nad četiri datoteke sa po 10 primjera. Rezultati se nalaze u tablici 6.1.

Tablica 6.1: Rezultati testiranja - *virus*(% točnosti)

Testni primjer	min	max	avg
1	56	72	64
2	56	72	62
3	56	66	63,6
4	62	72	66,4
	57,5	70,5	64

Za riječ *rak* napravljeno je isto, samo što je u svakoj datoteci bilo po 9 primjera. Rezultati se nalaze u tablici 6.2

Tablica 6.2: Rezultati testiranja - *rak*(% točnosti)

Testni primjer	min	max	avg
1	44,4	77,7	56,39
2	33,3	66,6	54,61
3	44,4	100	56,39
4	44,4	77,7	56,83
	41,63	80,5	56

Generiranje datoteka znanja je vremenski veoma zahtjevno i već za kontekst veličine 5 rijeli za generiranje jedne datoteke znanja potrebno je otprilike 15 minuta. Također vrijeme potrebno za generiranje datoteke znanja ovisi i o broju spominjanja riječi u korpusu.

Iz tog razloga testirano je samo sa kontekstima veličina 1–5 riječi s ciljem da se ustanovi koja je optimalna veličina konteksta da bi se dobila najveća točnost. Rezultati točnosti s obzirom na veličinu konteksta dati su u tablici 6.3.

Najbolji rezultati dobiveni su za veličine konteksta dvije i četiri riječi i u nastavku će se samo te veličine koristiti za testiranje. Umjesto četiri datoteke znanja za svaku veličinu konteksta biti će generirane po dvije za svaku veličinu, dakle ukupno četiri.

Tablica 6.3: Rezultati s obzirom na veličinu konteksta(% točnosti)

Veličina konteksta	1	2	3	4	5
virus	62,5	69,5	63,5	61	63,5
rak	53,28	52,73	55,5	60,5	58,28
	57,89	61,11	59,5	60,75,	60,88

Za daljnje testiranje uzeto je šest riječi, tri od njih imaju tri značenja dok preostale imaju po dva značenja. Konačni rezultati nalaze se u tablicama 6.4 i 6.5.

Tablica 6.4: Rezultati sveukupnog testiranja za kontekste veličine 2 i 4 za riječi sa dva značenja(% točnosti)

Riječ	Veličina konteksta 2	Veličina konteksta 4
grana	70	70
paket	55	65
park	60	85
avg	61,66	73,33

Iz rezultata u tablici 6.4 vidimo da je prosječna točnost kada je veličina konteksta dvije riječi sa svake strane višeznačne riječi 61,66%, a kada je veličina konteksta četiri riječi točnost je 73,33%.

Tablica 6.5: Rezultati sveukupnog testiranja za kontekste veličine 2 i 4 za riječi sa tri značenja(% točnosti)

Riječ	Veličina konteksta 2	Veličina konteksta 4
ekipa	49,95	55,5
pauk	61,05	66,6
plod	66,6	44,4
avg	55,87	51,88

Vidimo da točnost razrješavanja za riječi sa tri značenja koja je prikazana u tablici 6.5 manja nego za riječi sa dva značenja. Za veličinu konteksta dvije riječi iznosi 55,87%, a za četiri riječi 51,88%.

U jedinom srodnom radu koji je vezan uz hrvatski jezik [3] autori su testiranjem sa višeznačnicom *stanica* došli do zaključka da su najbolji rezultati za male veličine konteksta. Ističu važnost riječi koje nalazi neposredno ispred jer

ona pobliže opisuje višeznačnu riječ te joj jednoznačno dodjeljuje značenje (npr. tramvajska stanica, matična stanica, biljna stanica).

U radu [3] se koristi označeni korpus, te su autori pronašli čak šest značenja riječi stanica. U ovom radu moguća značenja preuzeta su iz rječnika [1] koji prepoznaje samo dva značenja. Dodatnim testiranjem želimo utvrditi da li isto pravilo vrijedi i u našem slučaju. Rezultati testiranja dani su u tablici 6.6.

Tablica 6.6: Točnost s obzirom na veličinu konteksta - *stanica*(% točnosti)

Veličina konteksta	1	2	3	4
Točnost (%)	75	70	57,5	55

Vidimo da rezultati testiranja potvrđuju da točnost pada da veličinom konteksta. Razlog tome je to riječ koja se nalazi neposredno ispred riječi *stanica* određuje njezino značenje. Kod večer konteksta u obzir se uzima više riječi, a kako ne dodjeljujemo težinski faktor s obzirom na udaljenost od višeznačnice u pitanju, težina najbitnije riječi iz konteksta se smanjuje.

Kod ostalih riječi ne možemo ovako precizno utvrditi utjecaj veličine konteksta na točnost. Pokazalo se da za ovdje testirane primjere najbolju točnost daje veličina konteksta od četiri riječi sa svake strane višeznačnice.

7. Zaključak

U radu je prikazan razvoj i rezultati sustava za razgraničavanje višeznačnosti koji uči iz neoznačenog korpusa algoritmom maksimizacije očekivanja. Ovaj veoma složen problem iz područja obrade prirodnog jezika morali smo pojednostaviti tako da u obzir uzimamo samo imenice i to one sa dva ili tri značenja. Ograničenje broja značenja bilo je potrebno napraviti i iz razloga što je samo učenje veoma vremenski zahtjevno, a složenost raste s porastom broja značenja.

Dobiveni rezultati točnosti se kreću u rasponu od 55–73% ovisno o veličini konteksta i broju značenja višeznačnice. Jedini rad vezan uz hrvatski jezik [3] bavi se problemom razrješavanja višeznačnosti i zato rezultate nije moguće direktno uspoređivati, no potvrđena je tvrdnja da kod riječi *stanica* točnost raste kako je kontekst manji zbog direktne ovisnosti značenja riječi sa riječi koja joj prethodi.

Usporedba rezultata sa onima iz rada vezanog za engleski jezik [5] moguća je samo za riječi s dva značenja, gdje točnost iznosi 63–68%. U ovom radu srednja točnost za riječi s dva značenja je približno 67% pri čemu vidimo da su dobiveni rezultati gotovo identični.

Trenutnu izvedbu algoritma moglo bi se nadograditi na mnogo načina. Moglo bi se ostvariti da kontekst prelazi granice rečenice, ali ne i članka,te utvrditi kakav bi bio utjecaj na točnost. Druga preinaka bi moglo biti uzimanje isključivo lijevog ili desnog konteksta ili pak upotreba težinskih faktora s obzirom na udaljenost u kontekstu. Zatim moguće je testirati sa novim primjerima i riječima i vidjeti ovisnost bliskosti značenja i točnosti.

Pretpostavka je da su pojedine riječi u ovom testiranju dale lošije rezultate zbog korpusa iz kojeg je algoritam učio. U njemu nije moguće naći sva značenja pojedine riječi u dovoljnom broju da bismo dobili realnu reprezentaciju. Korpus bi trebalo proširiti tekstovima iz drugih izvora kao što su časopisi, knjige, udžbenici, internetske stranice i blogovi. Time bi se dobio širi pregled mogućih uporaba riječi, s naglaskom na žargonske izraze koje je u novinama teško pronaći, a veoma često jedno značenje riječi je žargonsko.

Kasnije je moguće proširenje na druge vrste riječi i uklapanje u veće sustave za strojno prevođenje ili klasifikaciju rezultata pretraživanja, te pokušaj optimizacije koraka izvođenja i brzine algoritma.

Literatura

[1] Velimir Anić. *Veliki rječnik hrvatskoga jezika*. Novi Liber, 2003.

[2] Skupina autora. Vjesnik on-line arhiva, 1998-2010. URL `http://www.vjesnik.hr/html/`.

[3] Nikola Bakarić, Jasmina Njavro, i Nikola Ljubešić. What makes sense? searching for strong wsd predictors in croatian, 2007.

[4] Jeff Bilmes. A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical report, 1998.

[5] Marie-Catherine de Marneffe i Pierre Dupont. Comparative study of statistical word sense discrimination techniques, 2004.

[6] A. P. Dempster, N. M. Laird, i D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977. Series B.

[7] Esther Levin, Mehrbod Sharifi, i Jerry Ball. Evaluation of utility of lsa for word sense discrimination. U *In Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, stranice 77–80, 2006.

[8] Hinrich Schutze. Automatic word sense discrimination. *Journal of Computational Linguistics*, 24:97–123, 1998.

[9] H. Shinnou i M. Sasaki. Unsupervised learning of word sense disambiguation rules by estimating an optimum iteration number in the EM algorithm. U *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, stranice 41–48. Association for Computational Linguistics, 2003.

[10] Jan Šnajder, Bojana Dalbelo Bašić, i Marko Tadić. Automatic acquisition of inflectional lexica for morphological normalisation. *Information Processing and Management*, 44(5):1720–1731, 2008.

[11] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. U *ACL*, stranice 189–196, 1995.

Dodatak A
Detalji testiranja: Virus

U nastavku se nalaze detaljne tablice ispitivanja točnosti da riječ *virus*. U svakoj tablici dani su rezultati za četiri tetstna uzorka i pet datoteka znanja koje ovise o veličini konteksta.

Tablica A1: Virus 1(% točnosti)

1 x 1	1	2	3	4	5	
uzorak 1	60	50	60	60	50	56
uzorak 2	50	50	50	50	80	56
uzorak 3	60	70	60	60	80	66
uzorak 4	80	50	80	80	70	72
	62,5	55	62,5	62,5	70	62,5

Tablica A2: Virus 2(% točnosti)

2 x 2	1	2	3	4	5	
uzorak 1	60	80	60	90	70	72
uzorak 2	70	70	60	80	80	72
uzorak 3	50	80	80	70	50	66
uzorak 4	70	60	60	70	80	68
	62,5	72,5	65	77,5	70	69,5

Tablica A3: Virus 3(% točnosti)

3 x 3	1	2	3	4	5	
uzorak 1	70	60	70	50	60	62
uzorak 2	60	50	70	70	50	60
uzorak 3	90	50	50	70	70	66
uzorak 4	70	60	80	50	70	66
	72,5	55	64,5	60	62,5	63,5

Tablica A4: Virus 4(% točnosti)

4 x 4	1	2	3	4	5	
uzorak 1	70	50	60	60	80	64
uzorak 2	60	70	60	70	50	62
uzorak 3	60	60	60	50	50	56
uzorak 4	60	70	60	70	50	62
	62,5	62,5	60	62,5	57,5	61

Tablica A5: Virus 5(% točnosti)

5 x 5	1	2	3	4	5	
uzorak 1	50	80	60	70	70	66
uzorak 2	50	50	60	70	70	60
uzorak 3	60	60	70	60	70	64
uzorak 4	60	60	50	70	80	64
	55	62,5	60	67,5	72,5	63,5

Dodatak B
Detalji testiranja: Rak

U nastavku se nalaze detaljne tablice ispitivanja točnosti da riječ *rak*. U svakoj tablici dani su rezultati za četiri tetstna uzorka i pet datoteka znanja koje ovise o veličini konteksta.

Tablica B1: Rak 1(% točnosti)

1 x 1	1	2	3	4	5	
uzorak 1	55,5	55,5	44,4	55,5	55,5	53,28
uzorak 2	66,6	44,4	66,6	55,5	44,4	55,5
uzorak 3	44,4	55,5	44,4	55,5	55,5	51,06
uzorak 4	55,5	55,5	55,5	44,4	55,5	53,28
	55,5	52, 73	52,73	52,73	52,73	53,28

Tablica B2: Rak 2(% točnosti)

2 x 2	1	2	3	4	5	
uzorak 1	55,5	44,4	44,4	55,5	55,5	51,06
uzorak 2	44,4	55,5	66,6	55,5	33,3	51,06
uzorak 3	44,4	55,5	44,4	44,4	77,7	53,28
uzorak 4	55,5	55,5	55,5	66,6	44,4	55,5
	49,95	52,73	52,73	55,5	52,73	52,73

Tablica B3: Rak 3(% točnosti)

1 x 1	1	2	3	4	5	
uzorak 1	55,5	55,5	55,5	66,6	66,6	59.94
uzorak 2	44,4	66,6	66,6	44,4	44,4	53,28
uzorak 3	55,5	44,4	55,5	55,5	55,5	53,28
uzorak 4	66,6	55,5	44,4	55,5	55,5	55,5
	55,5	55,5	55,5	55,5	55,5	55,5

Tablica B4: Rak 4(% točnosti)

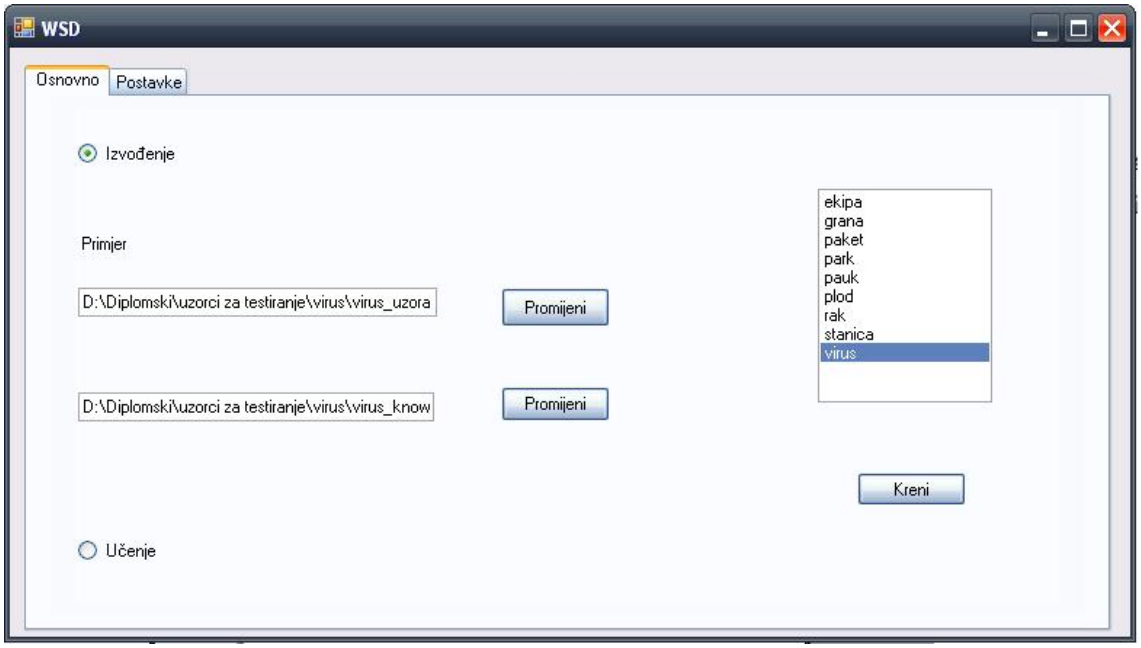
1 x 1	1	2	3	4	5	
uzorak 1	44,4	44,4	66,6	44,4	66,6	53,28
uzorak 2	55,5	55,5	66,6	55,5	66,6	59,94
uzorak 3	100	44,4	44,4	77,7	44,4	62,18
uzorak 4	55,5	77,7	77,7	66,6	55,5	66,6
	63,85	55,5	63,83	61,05	58,28	60,5

Tablica B5: Rak 5(% točnosti)

1 x 1	1	2	3	4	5	
uzorak 1	66,6	44,4	77,7	77,7	55,5	64,38
uzorak 2	44,4	55,5	55,5	55,5	55,5	53,28
uzorak 3	77,7	44,4	66,6	77,7	44,4	62,16
uzorak 4	44,4	66,6	44,4	55,5	55,5	53,28
	58,28	52,73	61,05	66,6	52,73	58,28

Dodatak C
Upute za korištenje

Pri pokretanju programa otvara se glavni prozor prikazan na slici C1. U ovom prozoru vršimo odabir načina rada: učenje ili izvođenje.

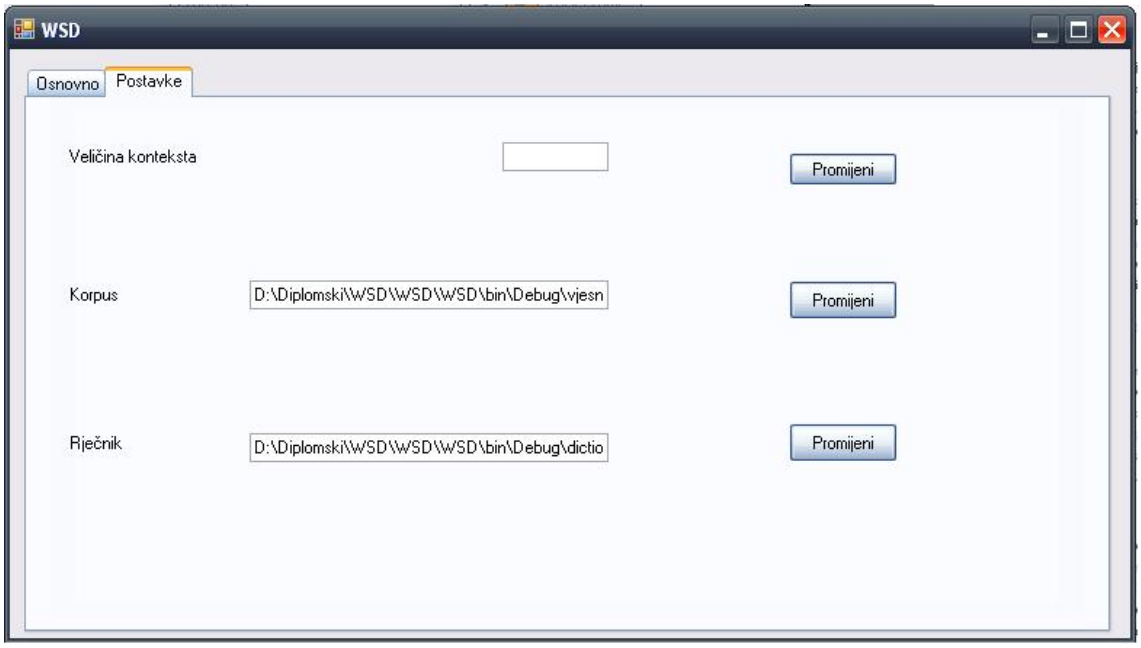


Slika C1: Glavni prozor

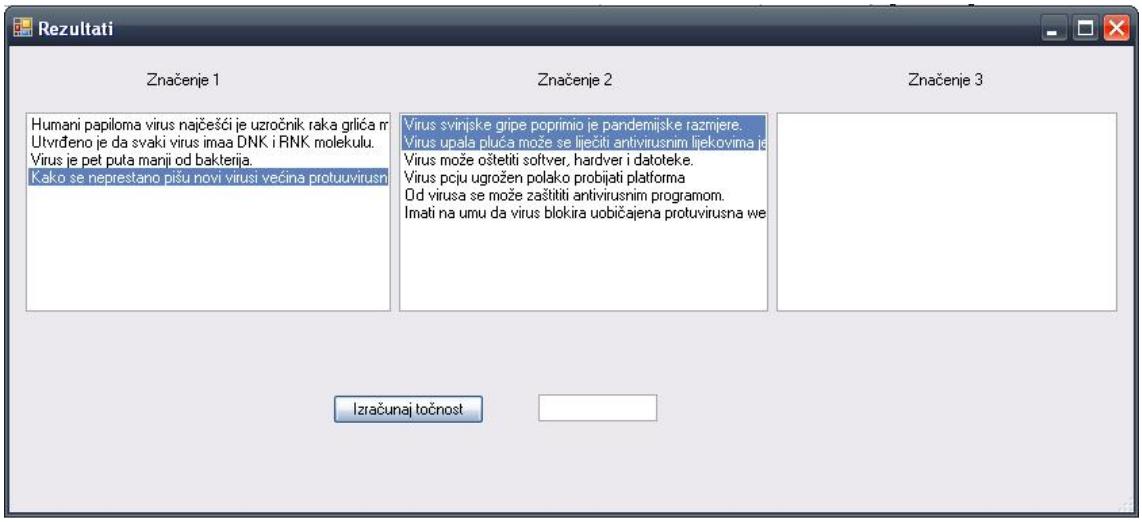
Ako je izabrani način rada učenje tada je potrebno samo odabrati željenu riječ u izborniku i pokrenuti program pritiskom na tipku *Kreni*. Dodatno je još moguće podesiti veličinu konteksta ili datoteke iz kojih se učitavaju korpus i rječnik. Ta podešenja rade se u kartici *Postavke* koja je prikazana na slici C2.

Ako je izabrani način rada izvođenje tada je potrebno zadati datoteku s primjerima i datoteku u kojoj se nalaze vjerojatnosti genereirane izvođenjem algritma, tzv. datoteku znanja. Također postoji mogućnost promjene veličine konteksta kao i kod učenja. Nakon toga se odabire riječ na koju se primjeri odnose i pri-

tiskom na tipku *Kreni* se pokreće program i prikazuje prozor s rezultatima.



Slika C2: Prozor za izmjene postavki



Slika C3: Prozor za prikaz rezultata

Prozor s rezultatima prikazan je na slici C3. Rezultati koje daje algoritam su prikazani tako da su rečenice u kojima se višeznačna riječ spominje u istom smislu grupirane u jedan prozor.

Ako želimo izračunati točnost izaberemo rečenice koje su krivo svrstane i pritiskom na tipku *Izračunaj točnost* prikazuje nam se točnost za navedeni primjer.

RAZGRANIČAVANJE VIŠEZNAČNOSTI RIJEČI UPORABOM ALGORITMA MAKSIMIZACIJE OČEKIVANJA

Sažetak

Kod mnogih se zadataka dubinske analize teksta, strojnog prevođenja i pretraživanja informacija problematičnom pokazuje višeznačnost riječi prirodnog jezika. Postupcima razgraničavanja višeznačnosti moguće je pojavljivanja višeznačne riječi svrstati u razrede, gdje svaki razred odgovara jednom značenju riječi. U okviru rada ostvarena je programska implementacija algoritma maksimizacije očekivanja (engl. *EM algorithm*) te provedeno eksperimentalno vrednovanje algoritma na uzorcima. Algoritam je primijenjen i vrednovan za hrvatski jezik.

Ključne riječi: višeznačnice, razgraničavanje višeznačnosti, algoritam maksimizacije očekivanja, Bayesov poučak, nenadzirano učenje

Word sense discrimination using Expectation Maximization algorithm

Abstract

In many of the tasks of text mining, machine translation and information retreval we encounter problems with the ambiguity of natural language words. Word sense discrimination can classify ambiguous word occurrences into classes where each class corresponds to one sense of the word. As a part of the thesis implementation of Expectation Maximization algorithm (EM) was realized and the experimental evaluation of the samples was performed. The algorithm was applied and evaluated for the Croatian language.

Keywords: ambiguous words, word sense iscrimination, Expectation Maximization algorithm, Bayes' Theorem, unsupervised learning