

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1060

**APLIKACIJA ZA AUTOMATIZIRANO
STVARANJE STATISTIČKOG JEZIČNOG
MODELA OBRADOM JEZIČNOG
KORPUSA**

Tomislav Kovačić

Zagreb, lipanj 2015.

Sadržaj

| | |
|--|----|
| Uvod | 1 |
| 1. Modeliranje jezika i modeli N-grama | 3 |
| 1.1 Jezični korpus i brojanje riječi u korpusu | 3 |
| 1.2 N-grami | 4 |
| 1.3 Priprema korpusa i osjetljivost N-grama | 8 |
| 2. Zaglađivanje | 9 |
| 2.1 Aditivno zaglađivanje | 10 |
| 2.2 Good-Turing procjena | 14 |
| 2.3 Witten-Bell zaglađivanje | 19 |
| 2.3.1 Witten-Bell snižavanje | 19 |
| 2.3.2 Witten-Bell odstupanje | 22 |
| 2.4 Apsolutno snižavanje | 24 |
| 2.5 Kneser-Ney zaglađivanje | 25 |
| 2.6 Modificirano Kneser-Ney zaglađivanje | 30 |
| 3. Skriveni Markovljev model | 32 |
| 4. Viterbijev algoritam | 33 |
| 5. Vrednovanje modela | 37 |
| 5.1 Entropija | 37 |
| 5.2 Stopa pogreške riječi | 40 |
| 6. Aplikacija za stvaranje jezičnog modela | 41 |
| 6.1 Zahtjevi i specifičnosti | 41 |
| 6.2 Jezični korpus | 41 |
| 6.3 Arhitektura | 43 |
| 6.4 Rezultati | 45 |

| | |
|------------------|----|
| Zaključak | 47 |
| Literatura | 48 |
| Sažetak..... | 50 |
| Summary | 51 |

Uvod

Jezični modeli su poveznica mnogih znanstvenih disciplina i domena kao što su to digitalna obrada govora, optičko raspoznavanje znakova, raspoznavanje rukopisa, strojno učenje, ispravljanje pravopisa te mnogih drugih. Kako bi se opisala sva kompleksnost jezika te kako bi se njegovo ponašanje moglo modelirati potrebno je imati širok spektar znanja o jeziku. Neka od područja proučavanja jezika uz pomoć kojih možemo modelirati njegovo ponašanje su:

- Fonetika i fonologija (glasoslovlje) – znanstvene discipline koje se bave proučavanjem artikulacijskih i akustičkih obilježja glasova i govora
- Morfologija ili oblikoslovlje – znanstvena disciplina koja proučava sustav jezičnih oblika, odnosno načina na koji se riječi u nekom jeziku oblikuju i mijenjaju
- Sintaksa – dio gramatike u kojem se proučavaju pravila koja upravljaju ustrojem rečenica
- Semantika – znanstvena disciplina posvećena proučavanju značenja, zasnovano na sintaksnim razinama riječi, fraza, rečenica ili većih jezičnih cjelina
- Pragmatika – dio lingvistike i teorije komunikacije koji se bavi odnosima između znakova i njihovih tumača u odnosu na situaciju u kojoj se oni nalaze, njihove ciljeve i potrebe, odnosno na koji je način jezik korišten u ostvarivanju ciljeva
- Diskurs – disciplina koja proučava složene lingvističke jedinice

Uz svu nabrojanu složenost jezika jedna od ključnih pretpostavki prilikom obrade jezika je da sva spomenuta jezična područja mogu biti obuhvaćena koristeći relativno mali broj formalnih modela i teorija koje proizlaze iz dobro definiranih područja računarske znanosti, matematike i lingvistike. Neki od važnijih između njih su teorija automata, logika, formalne metode, posebice teorija vjerojatnosti i ostali oblici strojnog učenja. Automati stanja u svojem osnovnom obliku se sastoje od skupa stanja u kojima se automat može nalaziti, skupa prijelaza između stanja te od nekog ulaznog skupa koji će inicirati prijelaze između stanja. Između često korištenih determinističkih i

nedeterminističkih te težinskih automata u vidu ovog rada važni će biti Markovljevi modeli te skriveni Markovljevi modeli koji imaju vjerojatnosnu komponentu. Teorija vjerojatnosti ima važnu ulogu u procesu obrade prirodnog jezika te proširuje spomenute modele i načine obrade jezika. U konačnici svaki svaki vid obrade jezika se susreće sa osnovnim problemom a to je višeznačnost ulaznog skupa. Imajući takav ulazni skup kojeg treba klasificirati potrebno je donijeti odluku koja je najvjerojatnija moguća od ponuđenih mogućnosti.

U okviru ovog rada statistički jezični modeli će pretpostaviti postojanje Markovljevih svojstava među uzastopnim riječima prirodnog jezika. U tu svrhu bit će potrebno odrediti vjerojatnost pojavljivanja pojedinih nizova riječi zadane duljine koji se nazivaju N-grami u velikom jezičnom korpusu na temelju njihove frekvencije pojavljivanja. Jezični korpus ima svojstvo podatkovne rijetkosti stoga je vrlo vjerojatno da se sve moguće kombinacije riječi u njemu neće pojaviti. Iako kombinacija nije uočena u korpusu iz kojeg gradimo jezični model to ne znači da ona u jeziku ne može postojati. Potrebno je primjeniti odgovarajuće tehnike koje će i takvoj kombinaciji pridjeliti odgovarajuću vjerojatnost imajući u vidu razne informacije sadržane u jezičnom korpusu na temelju kojih će se ta vjerojatnost pokušati što bolje procijeniti. Te tehnike se nazivaju tehnikama zaglađivanja (engl. *smoothing techniques*) i imaju za cilj bolje procijeniti vjerojatnosti kada nemamo dovoljno podataka da ih točno odredimo. Za rješavanje ove problematike razvijeno je dosta algoritama zaglađivanja od kojih će oni koje se dominantno danas koriste biti opisani i testirani u ovom radu.

1. Modeliranje jezika i modeli N-grama

Mogućnost predviđanja sljedeće riječi je osnovni podzadatak u raspoznavanju govora, rukopisa i otkrivanju pravopisnih pogrešaka. U takvim zadaćama identifikacija riječi se pokazuje kao ne tako jednostavan zadatak pošto su ulazni podaci vrlo često višeznačni i nepovezani. U takvom okruženju prethodna riječ može biti važan podatak u određivanju sljedeće riječi sužujući vjerojatnosni prostor na osnovu kojeg donosimo odluku koja riječ je sljedeća u nizu. Predviđanje sljedeće riječi se pokazuje usko vezano uz problem određivanja vjerojatnosti niza uzastopnih riječi. Algoritmi koji pridjeljuju vjerojatnost određenom nizu riječi mogu biti korišteni prilikom određivanja vjerojatnosti sljedeće najvjerojatnije riječi u nepotpunim rečenicama što se uvelike koristi prilikom označavanja jezika (engl. *part-of-speech-tagging*) i određivanju značenja riječi. U raspoznavanju govora često se za statistički model niza riječi koristi termin jezični model. Jezični model se obično definira i kao distribucija vjerojatnosti koja govori koliko često se određeni niz riječi pojavljuje kao rečenica.

1.1 Jezični korpus i brojanje riječi u korpusu

Vjerojatnosti pomoću kojih gradimo jezični model dolaze kao posljedica brojanja riječi ili nizova riječi unutar jezičnog korpusa. Korpus je kolekcija raznih tekstova i izgovorenih riječi odnosno rečenica. Izgrađeni jezični model će ovisiti o vjerojatnostima koje proizlaze iz onoga što brojimo stoga je vrlo važno kao početni korak precizno odrediti što i kako brojimo. Brojanje riječi korpusa naziva još i treniranje modela, a takav korpus trening korpus. Unutar korpusa koji sadrže pisane tekstove susrećemo pored normalnih riječi i interpunkcijske znakove. Da li ćemo interpunkcijske znakove također tretirati kao i ostale riječi ovisi o specifičnoj primjeni kao što su to provjera gramatike ili ispravljanje pogrešaka gdje su ti znakovi od određene važnosti. S druge strane kod izgovorenih riječi ne postoje interpunkcijski znakovi ali se pojavljuju druge pojave koje ne želimo tretirati kao riječi kao što su na primjer razni šumovi, zastajkivanja ili zamuckivanje govornika. Također da li ćemo riječi razlikovati ukoliko jedna počinje velikim slovom ili ukoliko jednina riječi implicira oblik u množini koji se možda uopće ni ne razlikuju ovisi o primjeni modela.

Današnje tehnike i načini određivanja i brojanja riječi se baziraju na oblicima riječi (engl. *wordform*) koje odgovaraju upravo oblicima riječi onako kako se nalaze u korpusu. Ovaj način tretiranja riječi ne raspoznaje jedninu, množinu ili možda oblik riječi u nekom od padeža i slično. Ovo pojednostavljenje neće uvijek biti dobro jer bi se u nekim primjenama htjelo sve te oblike tretirati kao oblike koji su izvedeni iz osnovnog oblika riječi odnosno leme. Prilikom brojanja riječi razlikujemo dva podatka. Prvi je broj vrsta riječi odnosno broj različitih riječi u korpusu ili veličina vokabulara. Drugi broj je broj oznaka odnosno tokena koji predstavljaju ukupan broj riječi koje se nalaze u korpusu koji se obrađuje.

1.2 N-grami

Najjednostavniji model niza riječi bi prepostavio da bilo koja riječ može uslijediti nakon bilo koje riječi. To bi pak značilo da je svaka riječ jezika jednako vjerojatna kao sljedeća u nizu. U kompleksnijim modelima svaka riječ također može uslijediti nakon bilo koje riječi ali te riječi neće biti jednako vjerojatne već će to ovisiti o frekvenciji njihova pojavljivanja.

Najčešće korišten način modeliranja jezika je modelima N-grama. N-grami su niz susjednih elemenata teksta ili izgovorenih riječi. Ti elementi mogu biti fonemi, slogovi, riječi ili par riječi ovisno o namjeni. N-grami se obično prikupljaju iz raznih tekstova pa će stoga ti elementi tu biti riječi u tekstu. Model N-grama je vjerojatnosni jezični model za predviđanje sljedećeg elementa u nizu duljine $(n-1)$ elemenata. Dvije su osnovne prednosti korištenja ovog modela i algoritama koji ga koriste, a to su relativna jednostavnost modela i njegova mogućnost skaliranja povećavanjem broja n što će omogućiti jednostavnijim izvedbama pohranu većeg broja informacija uz uštedu resursa. N-grami duljine $N=1$ se nazivaju unigrami, duljine $N=2$ bigrami, duljine $N=3$ trigrami i tako dalje slično za N većeg reda. Rečenice s su sastavljene od riječi $w_1 \dots w_n$. Ako pretpostavimo pojavu svake riječi na određenom mjestu kao nezavisan događaj tada možemo izraziti vjerojatnost te rečenice kao $P(w_1, w_2, \dots, w_{n-1}, w_n)$ odnosno kraće zapisano $P(w_1^n)$. Ako iskoristimo svojstvo dekompozicije vjerojatnosti kao:

$$\begin{aligned} P(w_1^n) &= P(w_1)P(w_2|w_1)P(w_3|w_1^2) \dots P(w_n|w_1^{n-1}) \\ &= \prod_{k=1}^n P(w_k | w_1^{k-1}) \end{aligned} \tag{1.1}$$

Ovdje se postavlja pitanje kako izračunati vjerojatnosti $P(w_n|w_1^{n-1})$ pošto za to ne postoji niti jedan jednostavan način. Iz tog razloga uvodimo pojednostavljenje na način da ćemo vjerojatnost riječi aproksimirati na osnovu određenog niza riječi koje joj prethode. Najjednostavniji takav primjer je model bigrama koji aproksimira vjerojatnost riječi s obzirom na sve riječi koje joj prethode tj. $P(w_n|w_1^{n-1})$ kao uvjetnu vjerojatnost s obzirom na prethodnu riječ $P(w_n|w_{n-1})$. Pretpostavka da vjerojatnost neke riječi ovisi samo o prethodnoj riječi se naziva Markovljeva pretpostavka (engl. *Markov assumption*). Markovljevi modeli su vrsta vjerojatnosnih modela koji pretpostavljaju da možemo predvidjeti vjerojatnost nekog budućeg događaja gledajući ne tako daleko u prošlost događaja. Jednako kao i bigram koji gleda jednu prethodnu riječ možemo definirati i trigram koji gleda dvije prethodne riječi i u konačnici N-gram koji će gledati n prethodnih riječi. Bigram se još naziva Markovljevim modelom prvog reda, trigram modelom drugog reda, a općenito N-gram Markovljevim modelom N-1 reda.

Općenita formula za takvu aproksimaciju N-grama uvjetne vjerojatnosti sljedeće riječi je:

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-N+1}^{n-1}) \quad (1.2)$$

Jednadžba prikazuje da se vjerojatnost riječi w_n može aproksimirati imajući u vidu samo N prethodnih riječi, a ne sve koje joj prethode. Na primjeru bigrama vjerojatnost niza riječi se tako može predočiti koristeći jednadžbe 1.2 i 1.1 kao:

$$P(w_1^n) \approx \prod_{k=1}^n P(w_k | w_{k-1}) \quad (1.3)$$

Sljedeća tablica je ilustrativan primjer na koji način se može izračunati vjerojatnost rečenice ukoliko raspolažemo vjerojatnostima bigrama:

Tablica 1. Vjerojatnosti bigrama u korpusu

| | | | | | | | |
|------------|------|----------|------|------------|------|-------------|------|
| <BOS>Ja | 0.20 | Ja idem | 0.10 | idem kući | 0.11 | na ručak | 0.22 |
| <BOS>Ti | 0.15 | Ja želim | 0.08 | idem na | 0.25 | na posao | 0.35 |
| <BOS>Idemo | 0.05 | Ja sam | 0.30 | idem sutra | 0.05 | na fakultet | 0.09 |

U tablici bigrama se nalazi još dodatna oznaka <BOS> koja označava početak rečenice (engl. *beginning of sentence*).

Vjerojatnost rečenice “Ja idem na fakultet” se može izračunati množeći odgovarajuće vrijednosti bigrama:

$$\begin{aligned} P(\text{Ja idem na fakultet}) &= P(\text{Ja} | < \text{BOS} >) P(\text{idem} | \text{Ja}) P(\text{na} | \text{idem}) P(\text{fakultet} | \text{na}) \\ &= 0.20 * 0.10 * 0.25 * 0.09 = 0.00045 \end{aligned}$$

Kao što se može vidjeti vjerojatnosti N-grama će uvijek biti manje od 1, a njihovim množenjem dobivamo sve manje vrijednosti vjerojatnosti. U složenim aplikacijama koje rade s velikim brojem N-grama te prilikom izračunavanja vjerojatnosti dugačkog niza riječi na taj način riskiramo numeričku pogrešku. Stoga umjesto da množimo vjerojatnosti koristimo njihov logaritam $\log(P)$ te ih zbrajamo u logaritamskom prostoru što je ekvivalentna operacija. Na kraju dobiveni rezultat pretvorimo natrag iz logaritma u normalnu vjerojatnost.

Operacije s trigramima odnosno općenito s N-gramima slijede isti princip kao i bigrami. Kod modela trigrama prethodne dvije riječi uvjetuju vjerojatnost, kod prvog trigrama one mogu biti predstavljene kao dvije početne oznake npr. <BOS1><BOS2>Ja, no način kako ćemo ih definirati ovisi ponovo o konkretnoj primjeni.

Vjerojatnosti koje dobivamo operacijama s N-gramima moraju biti unutar intervala [0,1] stoga model N-grama može biti treniran brojanjem N-grama u korpusu te potom normalizacijom odnosno dijeljenjem s njihovim ukupnim brojem, na taj način vjerojatnosti će se uvijek nalaziti u željenom intervalu. Vjerojatnost bigrama u tako definiranim uvjetima u korpusu je:

$$P(w_n | w_{n-1}) = \frac{C(w_{n-1}w_n)}{\sum_w C(w_{n-1}w)} \quad (1.4)$$

Odnosno vjerojatnost bigrama u korpusu odgovara ukupnom broju pojava tog bigrama podijeljenog sa zbrojem svih bigrama koji imaju istu prvu riječ (w_{n-1}) kao i promatrani bigram. Taj broj djelitelj za bigrame ne znači ništa drugo nego ukupan broj unigrama w_{n-1} stoga se jednadžba može napisati i kao:

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})} \quad (1.5)$$

Općenita formula primjenjiva na bilo koji broj N glasi:

$$P(w_n|w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}w_n)}{C(w_{n-N+1}^{n-1})} \quad (1.6)$$

Formula 1.6 predviđa vjerojatnost N-grama kao omjer ukupnog broja pojavljivanja N-grama podijeljenog s ukupnim brojem pojavljivanja njegovog prefiksa. Taj omjer se naziva relativna frekvencija. Tehnika gdje se relativne frekvencije koriste za predviđanje vjerojatnosti poznata je kao procjena najveće vjerojatnosti (engl. *Maximum Likelihood Estimation, MLE*). Primjena tehnike na skupu podataka uz odgovarajući statistički model rezultirat će procjenom vjerojatnosti parametara modela. Drugim riječima ako imamo na raspolaganju određeni statistički model te konačni skup podataka MLE će odabrati onaj skup vrijednosti parametara modela koji će maksimizirati funkciju vjerojatnosti. Za mnoge modele MLE može biti predstavljen kao funkcija promatranih podataka x_1, \dots, x_n , a za druge modele MLE se mora pronaći numerički koristeći neke od metoda optimizacije. S druge strane postoje problemi kod kojih uopće nije moguće maksimizirati vjerojatnost tj. nije moguće dostići supremum. Na konkretnom primjeru brojanja riječi u korpusu određena riječ može imati određeni broj pojavljivanja, a MLE njene vjerojatnosti će biti ukupan broj njene pojave podijeljen s ukupnim brojem svih riječi u korpusu. Iako nam ta vrijednost može dati neki okvirni uvid koliko je riječ vjerojatna metoda procjene MLE uglavnom nije dobra zbog toga što u nekim drugim korpusima riječ može imati potpunu drugačiju distribuciju vjerojatnosti. Za očekivati je da će vjerojatnosti riječi varirati ovisno da li je korpus sastavljen od raznih novinskih članaka ili se radi o djelima nekog književnika gdje vjerojatnosti primjerice mogu biti utjecane književnikovim osebnim stilom pisanja.

1.3 Priprema korpusa i osjetljivost N-grama

Kao što je već spomenuto statistički jezični model kao što je model N-grama se gradi iz jezičnog korpusa koji se koristi za trening modela. Iz toga razloga vrlo je važno kvalitetno i pažljivo dizajnirati jezični korpus jer će buduće performanse i parametri modela o tome ovisiti. U nekim korpusima vjerojatnosti mogu biti usko vezane za određeno područje iz kojeg korpus potječe pa se stoga model neće moći dobro prilagoditi ukoliko se zatekne u nekom drugom okruženju tj. domeni. S druge strane ako je korpus preopćenit on također neće dati dobre rezultate jer se neće moći prilagoditi specifičnoj domeni na očekivani način. Ako smo model istrenirali nad određenim skupom rečenica i ako za test modela koristimo iste rečenice koje smo koristili prilikom treniranja tada će one imati neprirodno velike vjerojatnosti jer potječu iz istog izvora. Dakle trening korpus ne smije biti u sprezi s testnim rečenicama jer se na taj način izgrađeni model neće moći ocijeniti. Prije početka izgradnje jezičnog modela korpus treba podijeliti na skup rečenica namjenjene treningu modela i skup rečenica namjenjene testiranju modela. Koristimo jezični model kako bismo izračunali vjerojatnosti na testnom skupu. Ova paradigma se može koristiti prilikom usporedbe performansi različitih modela N-grama posebice algoritama zaglađivanja. Nastojimo ocijeniti koliko dobro će trenirani jezični model moći modelirati testni skup. Da bi to bili u mogućnosti koristit ćemo mjeru koja će nam moći dati odgovor koliko dobro statistički model odgovara testnom skupu (korpusu). Ta mjera je oblik entropije i naziva se perpleksija. Međutim najbolji način vrednovanja modela bi bio kad bi ga se pustilo u rad na konkretnom zadatku. To bi mogao biti alat koji raspoznaje govor ili ispravlja pravopis i slično koji bi koristio jezični model. Takav način vrednovanja se naziva još i ekstrinzični način vrednovanja jezičnog modela. Problem takvog načina vrednovanja je što je vremenski zahtjevan, vrednovanje može potrajati danima ili tjednima dok prilikom korištenja perpleksije to nije slučaj. Korištenje perpleksije pri vrednovanju modela se naziva još i intrinzičan način vrednovanja. Potrebno je kombinirati oba načina kako bi se sa sigurnošću mogla ocijeniti kvaliteta jezičnog modela.

2. Zaglađivanje

Jedan od glavnih problema osnovnog modela N-grama je taj što su trenirani na osnovu nekog korpusa. Jezik je vrlo kompleksan, on se sastoji od velikog broja riječi, jezičnih oblika, odnosa između riječi, skupova riječi, rečenica itd. Jezični korpus ma kako velik bio je konačan i s njime nikad nismo u mogućnosti u potpunosti obuhvatiti sve mogućnosti nekog jezika. Iz tog razloga neki N-grami koji su sasvim točni i pripadaju jeziku se ne pronalaze u korpusu. Svojstvo korpusa je da je podatkovno rijedak što će rezultirati da će matrica vjerojatnosti N-grama biti u velikom broju slučajeva popunjena vjerojatnostima vrijednosti 0 iako mi to ne bismo na tom mjestu očekivali. Svojstvo N-grama je da nastoje podcijeniti vjerojatnost niza koji se nije dogodio u njihovoj neposrednoj blizini prilikom treniranja jer promatraju nizove konačne duljine N . To možemo uvidjeti na sljedećem primjeru: zamislimo rečenicu “Tomislav voli igrati nogomet.”, imamo

$$P(voli|Tomislav) = \frac{C(Tomislav\ voli)}{\sum_w C(Tomislav\ w)} = \frac{0}{5}$$

Riječ “Tomislav” se u korpusu pojavila 5 puta ali nikad na način da prethodi riječi “voli” što će rezultirati frekvencijom pojavljivanja ovog bigrama 0. Vjerojatnost prethodne rečenice je $P(Tomislav\ voli\ igrati\ nogomet) = 0$. Vjerojatnost ove rečenice je podcijenjena i sasvim je razumno očekivati određenu vjerojatnost njene pojave u jeziku. Jedna od osnovnih primjena jezičnih modela je u alatima za raspoznavanje govora gdje se za primljeni akustični signal A pokušava pronaći

rečenica S takva da vjerojatnost $P(S|A) = \frac{P(A|S)P(S)}{P(A)}$ bude maksimalna. Ukoliko

je $P(S) = 0$ tada će i $P(S|A)$ biti jednaka 0 i S nikad neće biti smatran transkriptom signala A bez obzira koliko razumljiv bio signal A . Ukoliko se takva situacija dogodi prilikom raspoznavanja govora potrebno ju je klasificirati kao pogrešku. Pridjeljujući neku vjerojatnost nizovima riječi ili N-gramima koji se u korpusu ne pojavljuju pomažemo sprečavanju pogrešaka prilikom raspoznavanja govora kao i u svim drugim područjima koji koriste ovakve jezične modele. Prethodno spomenuta MLE metoda također daje loše rezultate i kada frekvencije nisu nula, ali opet relativno male. S ciljem rješavanja ovog problema nastale su tehnike zaglađivanja. Tehnike zaglađivanja nastoje prilagoditi vjerojatnosti dobivene putem MLE kako bi se došlo do što je moguće točnijih vjerojatnosti. Razlog zašto se zovu

tehnikama zaglađivanja je taj što pokušavaju ujednačiti distribuciju vjerojatnosti prilagođavajući vjerojatnosti na način da vjerojatnosti malog iznosa ili one koje su iznosa nula povećaju, a vjerojatnosti većeg iznosa smanje. Danas je razvijeno puno algoritama zaglađivanja i njihovih varijacija koje na različite načine pokušavaju rasporediti vjerojatnosnu masu. Neki od njih su se pokazali bolji od ostalih te se danas intenzivno koriste u različitim domenama.

2.1 Aditivno zaglađivanje

Aditivno zaglađivanje je jedna od najjednostavnijih metoda zaglađivanja. U literaturi se još mogu sresti nazivi Laplaceovo ili Lidstonovo zaglađivanje. Ideja je svim frekvencijama N-grama dodati neki određenu vrijednosti prije nego taj broj normaliziramo u vjerojatnosti. Na taj način smo riješili problem frekvencija vrijednosti 0. Metoda je vrlo jednostavna i zapravo u mnogim slučajevima ne daje dobre rezultate ali bit će dobra osnova za druge bolje algoritme. Ideja algoritma je dodati neku vrijednost N-gramima koji se ne pojavljuju ili je njihov broj mali te im na taj način povećati vjerojatnosti, s druge strane to za N-grame većih frekvencija znači smanjivanje vjerojatnosti pošto ukupan zbroj uvijek mora biti $P = 1$. Vjerojatnost unigrama dobivenog preko MLE se računa kao broj tog unigrama podijeljen s ukupnim brojem unigrama. Unigrami su zapravo riječi u korpusu. Imamo:

$$P(w_x) = \frac{C(w_x)}{\sum_i C(w_i)} = \frac{C(w_x)}{N} \quad (2.1)$$

Algoritam daje prilagođene (izglađene) frekvencije N-grama, na primjeru unigrama taj broj je jednak:

$$C_i^* = (C_i + \delta) \frac{N}{N + \delta|V|} \quad (2.2)$$

U formuli 2.2 svakoj vrsti riječi tj. unigrama dodajemo neku vrijednost δ stoga taj broj moramo podijeliti s faktorom $\frac{N}{N + \delta|V|}$ gdje je N broj oznaka (tokena), a V je broj vrsta riječi odnosno na primjeru unigrama to je upravo veličina vokabulara.

Kako bismo iz takvih zaglađenih frekvencija dobili zaglađene vjerojatnosti potrebno je dobivene frekvencije još podijeliti s brojem oznaka N.

$$P_i^* = \frac{C_i^*}{N} \quad (2.3)$$

Pošto algoritmi zaglađivanja smanjuju frekvencije N-grama koje se pojavljuju u korpusu kako bi povećale frekvencije N-grama koji se ne pojavljuju, drugi način gledanja na njih je u vidu snižavanja (engl. *discounting*) stoga se može definirati faktor snižavanja kao omjer zaglađenih i originalnih frekvencija:

$$D_c = \frac{C^*}{C} \quad (2.4)$$

Nakon uvida kako se aditivno zaglađivanje definira nad unigramima možemo ga proširiti na bigrame:

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n) + \delta}{C(w_{n-1}) + \delta|V|} \quad (2.5)$$

odnosno općenito za N-grame:

$$P(w_n|w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}w_n) + \delta}{C(w_{n-N+1}^{n-1}) + \delta|V|} \quad (2.6)$$

Parametar V se u formuli zaglađivanja za N-grame 2.6 definira kao broj mogućih vrsta N-grama reda N-1 odnosno veličina vokabulara (N-1)-grama. Parametar δ se definira na intervalu $0 < \delta < 1$. Često se za parametar δ uzima vrijednost 1, pa je algoritam poznat i pod nazivom *add-one*. Za parametar δ se može i uzeti vrijednost $\frac{1}{2}$ prema Jeffreys-Perks zakonu gdje je to maksimalno očekivanje MLE.

Na sljedećem primjeru windows haiku korpusa demonstrirat će se rad algoritma. Haiku korpus se sastoji od 16 haikua, 253 oznaka te 165 vrsta riječi. Oznaka <BOS> označava početak rečenice, a <EOS> kraj rečenice.

Tablica 2. Zaglađene vjerojatnosti unigrama windows haiku korpusa

| unigram | frekvencija | nezaglađeni MLE: $\frac{C(w)}{N}$ | add-one: $\frac{C(w)+1}{N+V}$ |
|----------|-------------|--------------------------------------|-------------------------------|
| . | 35 | 0.1383 | 0.0860 |
| , | 8 | 0.0316 | 0.0215 |
| the | 7 | 0.0277 | 0.0191 |
| The | 4 | 0.0158 | 0.0119 |
| that | 3 | 0.0119 | 0.0095 |
| on | 2 | 0.0079 | 0.0072 |
| We | 1 | 0.0040 | 0.0048 |
| operator | 0 | 0.0000 | 0.0024 |

Tablica 3. Zaglađene vjerojatnosti bigrama wondows haiku korpusa

| bigram | frekvencija bigrama | frekvencija W_{n-1} | nezaglađeni MLE: $\frac{C(w_{n-1}w_n)}{C(w_{n-1})}$ | add one: $\frac{C(w_{n-1}w_n)+1}{C(w_{n-1})+V}$ |
|-------------|------------------------|--------------------------|--|--|
| .<EOS> | 35 | 35 | 1.0000 | 0.1800 |
| <BOS> The | 3 | 35 | 0.0857 | 0.0200 |
| <BOS> You | 2 | 35 | 0.0571 | 0.0150 |
| is not | 2 | 7 | 0.2857 | 0.0174 |
| Your ire | 1 | 2 | 0.5000 | 0.0120 |
| You bring | 1 | 3 | 0.3333 | 0.0119 |
| not found | 1 | 4 | 0.2500 | 0.0118 |
| is the | 0 | 7 | 0.0000 | 0.0058 |
| This system | 0 | 2 | 0.0000 | 0.0060 |

Uobičajeno je nakon provedenog zaglađivanja ponovo izračunati vrijednosti frekvencija kako bi uspoređujući s originalnom vrijednosti frekvencije mogli vidjeti u kojoj ih je mjeri je algoritam zaglađivanja promijenio. Tablica 4. prikazuje rekonstruirane vrijednosti frekvencija.

Tablica 4. Rekonstruirane vrijednosti frekvencija

| bigram | originalna frekvencija | zaglađena frekvencija |
|-------------|------------------------|-----------------------|
| .<EOS> | 35 | 6.3000 |
| <BOS> The | 3 | 0.7000 |
| <BOS> You | 2 | 0.5250 |
| is not | 2 | 0.1218 |
| Your ire | 1 | 0.0240 |
| You bring | 1 | 0.0357 |
| not found | 1 | 0.0472 |
| is the | 0 | 0.0406 |
| This system | 0 | 0.0120 |

Iz tablice 4. se može vidjeti da je algoritam zaglađivanja uvelike izmijenio vrijednosti određenih frekvencija. Frekvencija bigrama .<EOS> je zaglađena na vrijednost 6.3 u odnosu na originalnu vrijednost 35, što je faktor smanjenja 5.56. Ta promjena se može vidjeti i u vjerojatnosnom prostoru gdje su neke vjerojatnosti doživjele velike promjene u odnosu na MLE, u tablici 3. označene su crvenom bojom. Uzrok ovakvih rezultata je taj što se prevelik udio ukupne vjerojatnosne mase usmjerava na sve vrste riječi sa frekvencijom 0. Odabir odgovarajućeg parametra δ može pomoći u ovom problemu no na kraju se dolazi do zaključka da ova metoda uglavnom nije dobar odabir za zaglađivanje gdje je u nekim slučajevima i obična MLE metoda može dati bolje rezultate.

2.2 Good-Turing procjena

Good-Turingova procjena je središnji dio mnogih algoritama zaglađivanja. Osnovna ideja Good-Turingova zaglađivanja je procijeniti vjerojatnosti N-grama čije su frekvencije vrlo male ili su jednake nula gledajući broj N-grama sa većim iznosima frekvencija. Kao glavni parametar gledamo vrijednost N_r što je broj N-grama koji su se pojavili s frekvencijom r . Vrijednost N_r nazivamo još i frekvencijom frekvencija. Slijedeći ovako definirane parametre N_0 je broj N-grama koji se nisu pojavili odnosno njihova frekvencija pojave je 0, N_1 je broj N-grama s frekvencijom 1, N_{15} je broj N-grama s frekvencijom 15 itd. Vrijedi:

$$N_r = \sum_{N:r(N)=r} 1 \quad (2.7)$$

Good-Turingova procjena nam daje zaglađene vrijednosti frekvencija r^* na osnovu skupa vrijednosti N_r za sve r .

$$r^* = (r + 1) \frac{N_{r+1}}{N_r} \quad (2.8)$$

Prema formuli 2.8 zaglađena vrijednost frekvencije N-grama koji se nisu pojavili u korpusu odgovara broju N-grama koji su se pojavili točno jedanput N_1 podijeljenog s brojem N-grama koji se uopće nisu pojavili N_0 . Tablica 5. prikazuje primjenu Good-Turingova algoritma nad bigramima koje su izračunali Church i Gale (1991) od 22 milijuna riječi dobivenih iz Associated Press (AP) informativne agencije. U tablici r označava frekvenciju pojave bigrama, N_r je broj bigrama koji su imali frekvenciju r dok r^* predstavlja Good-Turingovu procjenu frekvencije r .

Tablica 5. Frekvencije frekvencija 22 milijuna bigrama dobivenih iz Associated Press-a

| r (MLE) | N_r | $r^*(G-T)$ |
|-----------|----------------|------------|
| 0 | 74,671,100,000 | 0.0000270 |
| 1 | 2,018,046 | 0.446 |
| 2 | 449,721 | 1.26 |
| 3 | 188,933 | 2.24 |
| 4 | 105,668 | 3.24 |
| 5 | 68,379 | 4.22 |
| 6 | 48,190 | 5.19 |
| 7 | 35,709 | 6.11 |
| 8 | 27,710 | 7.21 |
| 9 | 22,280 | 8.25 |

Ova procjena pretpostavlja da znamo broj N-grama koji se nisu pojavili u korpusu tj. N_0 . Taj broj možemo izračunati ukoliko znamo veličinu vokabulara V . Broj svih mogućih vrsta N-grama reda N tada je jednak V^N , a broj N_0 je jednak razlici V^N i broja svih vrsta N-grama koji su se pojavili u korpusu. Nadalje zaglađena vrijednost frekvencije r^* se ne primjenjuje na sve vrijednosti r već samo do određenog broja k jer se prepostavlja da frekvencijama većih iznosa možemo vjerovati. Prema Katzu (1987) k se može postaviti na vrijednost 5. Tada vrijedi:

$$r^* = r \text{ za } r > k$$

(2.9)

Točna formula za r^* nakon što smo uveli parametar k glasi:

$$r^* = \frac{(r+1)\frac{N_{r+1}}{N_r} - r\frac{(k+1)N_{k+1}}{N_1}}{1 - \frac{(k+1)N_{k+1}}{N_1}}, \text{ za } 1 \leq r \leq k$$

(2.10)

Tablicom 6. prikazani su bigrami iz windows haiku korpusa zaglađeni korišteći Good-Turingov algoritam uz parametar $k=3$, vjerojatnosti bigrama jednake su:

$$P(w_n|w_{n-1}) = \frac{r^*(w_{n-1}w_n)}{C(w_{n-1})} \quad (2.11)$$

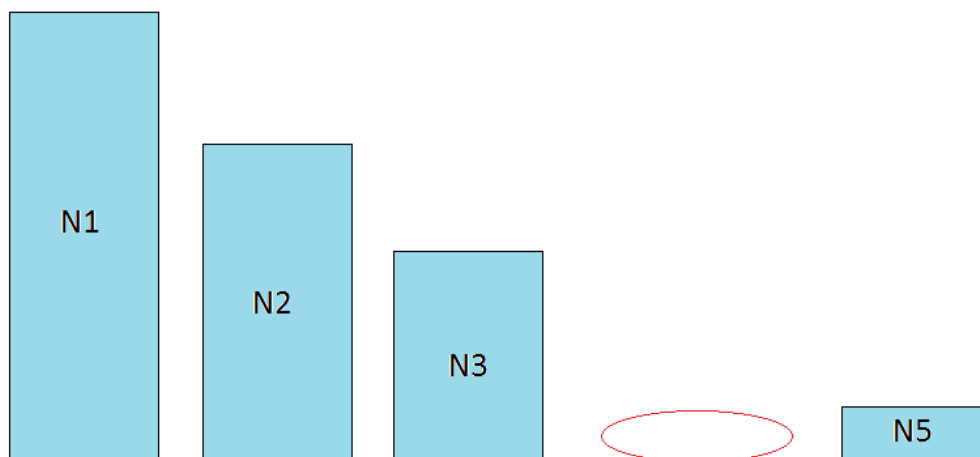
Tablica 6. Bigrami windows haiku korpusa zaglađeni algoritmom Good-Turing

| bigram | frekvencija | MLE | add-one | G-T |
|-------------|-------------|--------|---------|--------|
| .<EOS> | 35 | 1.0000 | 0.1800 | 1.0000 |
| <BOS> The | 3 | 0.0857 | 0.0200 | 0.0857 |
| <BOS> You | 2 | 0.0571 | 0.0150 | 0.0122 |
| is not | 2 | 0.2857 | 0.0174 | 0.1837 |
| Your ire | 1 | 0.5000 | 0.0120 | 0.0297 |
| You bring | 1 | 0.3333 | 0.0119 | 0.0198 |
| not found | 1 | 0.2500 | 0.0118 | 0.0148 |
| is the | 0 | 0.0000 | 0.0058 | 0.0012 |
| This system | 0 | 0.0000 | 0.0060 | 0.0044 |

Razlog zašto bi željeli koristiti parametar k i primjeniti algoritam do određenih iznosa frekvencija je i u tome što recimo za najveću frekvenciju r ne možemo izračunati parametar N_{r+1} koji u tom slučaju iznosi 0, odnosno:

$$r^* = (r + 1) \frac{N_{r+1}}{N_r} = (r + 1) \frac{0}{N_r} = 0$$

Također ova pojava ne vrijedi samo za najveći r , u korpusu sve frekvencije neće biti zastupljene, a to se posebno uočava pri velikim iznosima frekvencija r . Možemo recimo imati vrijednosti za r_{1991} i prvu sljedeću vrijednost r_{2222} , dakle postoje praznine između frekvencija što je pri velikim frekvencijama česta pojava. Ova pojava prikazana je na slici 1.



Slika 1. Pojava praznina između frekvencija

Zapravo bi vrijednost r^* trebali definirati kao:

$$r^* = (r + 1) \frac{E(N_{r+1})}{E(N_r)} \quad (2.12)$$

U formuli 2.12 $E(x)$ predstavlja očekivanje slučajne varijable x . Ukupna vjerojatnost svih N -grama frekvencije je definirana potom kao $\frac{E(N_1)}{N}$. Vrijednost N_1 je obično najveća i najvjerodostojnija od svih N_r , stoga je prihvatljivo zamijeniti ju sa $E(N_1)$. Kada je napravljena zamjena procjenu se obično naziva *Turingova* procjena. Ova supstitucija je opravdana samo za male iznose r stoga je Good (1953) predložio da se vrijednost N_r u formuli 2.12 zamijeni sa izglađenim vrijednostima $S(N_r)$, što se naziva *Good-Turingova* procjena. S obzirom da može biti više različitih načina zaglađivanja postoji i više načina kako definirati Good-Turingovu procjenu. Jedan od mogućih načina ćemo upravo spomenuti. Svaku vrijednost N_r koja nije 0 računamo s obzirom na vrijednosti N koje ju okružuju i koje također nisu 0. Drugim rječima potrebno je pronaći uzastopni niz frekvencija koje nisu 0 dok su indeksi tih frekvencija jednaki q , r i t . Vrijednost N_r tada zamjenjujemo s izrazom:

$$Z_r = \frac{N_r}{0.5(t-q)} \quad (2.13)$$

U formuli 2.13 procjenjujemo očekivanu vrijednost N_r sa gustoćom od N_r za velike r . Korak koji slijedi je zaglađivanje vrijednosti Z_r . Teško je pronaći najbolji način kako to učiniti stoga ćemo kombinirati dva pristupa. Za male iznose r se može koristiti prethodno spomenuta *Turingova* procjena koja će dati bolje rezultate od neke tehnike zaglađivanja dok se za velike iznose koristi *Good-Turingova* procjena. Najjednostavniji način zaglađivanja koji možemo koristiti je linija te se ovaj način procjene naziva Linearna Good Turing procjena (engl. *Linear Good Turing estimate*, LGT).

$$\log(N_r) = a + b \log(r) \quad (2.14)$$

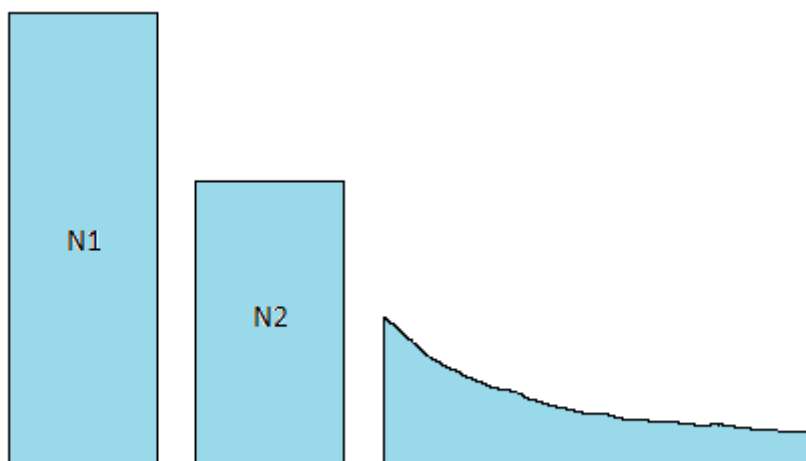
Pravilo za odabir između Turingove i Good-Turingove procjene je da se koristi Turingova procjena sve dok se ona bitno razlikuje od Good-Turingove procjene za N_r odnosno za neki faktor povjerenja. Jednom kada smo odlučili koristiti Good-Turingovu procjenu koristimo ju do kraja. Turingova procjena se smatra značajno različita od Good-Turingove procjene ukoliko se one razlikuju više od standardne devijacije što je korijen varijance Turingove procjene pomnožen s faktorom 1.65. Varijanca Turingove procjene je približno:

$$Var(r_T^*) = (r + 1)^2 \frac{N_{r+1}}{N_r^2} \left(1 + \frac{N_{r+1}}{N_r}\right) \quad (2.15)$$

S obzirom da zbrajamo dvije procjene ne možemo očekivati da će njihov zbroj biti 1, procjena u tom slučaju nije normalizirana. Kako bi zbroj procjena vjerojatnosti bio 1 dijelimo s ukupnim zbrojem procjena koje nisu normalizirane. Postupak se zove renormalizacija:

$$P_r = \left(1 - \frac{N_1}{N}\right) \frac{P_r^{nenorm}}{\sum_{r=1} P_r^{nenorm}}, r \geq 1 \quad (2.16)$$

Rezultat koji bi željeli postići prethodnim postupkom te odabirom odgovarajućeg parametra k prikazan je na slici 2. gdje su praznine sada popunjene.



Slika 2. Rezultat Good-Turingove procjene

U praksi Good-Turingova procjena se nikad ne koristi sama za zaglađivanje N-grama zbog toga što ne kombinira N-grame višeg i N-grame nižeg reda što je nužno za zadovoljavajuće performanse te se koristi kao dobar alat za ostale tehnike zaglađivanja.

2.3 Witten-Bell zaglađivanje

2.3.1 Witten-Bell snižavanje

Klasični Witten-Bellov algoritam zaglađivanja frekvencija vrijednosti 0 spada u algoritme snižavanja (engl. *discounting*) jer snižuje vrijednosti postojećih vjerojatnosti kako bi dio pridjelio onima vrijednosti 0 ili vrlo malog broja. Motivacija ovog algoritma zasnovana je na jednoj važnoj pretpostavci o pojavama frekvencija vrijednosti 0. N-gram frekvencije 0 možemo zamisliti kao N-gram koji se do tog trenutka još nije pojavio. Kada se pojavi to će tada biti prvi put pa se stoga vjerojatnost N-grama koji se još nije pojavio može modelirati kao vjerojatnost njegove pojave po prvi put. Ovaj koncept se često susreće u statističkoj obradi jezika. Vjerojatnost pojave N-grama po prvi put možemo izračunati iz jezičnog korpusa na način da prebrojimo N-grame koje smo tijekom treniranja modela vidjeli po prvi put. Ovaj broj je upravo jednak broju vrsta N-grama. Ukupna vjerojatnost svih N-grama može se izračunati po formuli:

$$\sum_{i:c_i=0} P_i^* \frac{T}{N+T} \quad (2.17)$$

Broj T u formuli 2.17 predstavlja broj N-grama koji su uočeni po prvi put tijekom treninga, što je različito od veličine vokabulara V ili ukupnog broja mogućih vrsta N-grama dok je N broj N-gram tokena. Jezični korpus se može gledati i kao niz događaja pojave vrste riječi i tokena te je formulom dan MLE vjerojatnosti pojave događaja. Vjerojatnosti koju smo dobili formulom 2.17 trebamo nekako razdijeliti na sve N-grame frekvencije 0. Najjednostavniji način je to učiniti ravnomjerno stoga definiramo Z , $Z = \sum_{i:c_i=0} 1$ kao broj N-grama frekvencije 0. Vjerojatnost pojedinačnog N-grama je tako:

$$P_i^* = \frac{T}{Z(N+T)} \quad (2.18)$$

Vjerojatnost koja je pridjeljena N-gramima frekvencije 0 se sada mora oduzeti od vjerojatnosti ostalih N-grama čija je sada vjerojatnost:

$$P_i^* = \frac{c_i}{N+1} \text{ za } c_i > 0 \quad (2.19)$$

Također umjesto vjerojatnosti možemo prikazati zaglađene frekvencije:

$$c_i^* = \begin{cases} \frac{T}{Z} \frac{N}{N+T} & \text{za } c_i = 0 \\ c_i \frac{N}{N+T} & \text{za } c_i > 0 \end{cases} \quad (2.20)$$

Na primjeru unigrama Witten-Bellov algoritam nalikuje aditivnom zaglađivanju dok se bitna razlika uočava ukoliko se algoritam primjeni na N-grame višeg reda. Kako bi izračunali vjerojatnost nekog bigrama $w_{n-1}w_{n-2}$ koji se još nije pojavio koristimo podatak o prethodnoj riječi tj. unigramu w_{n-1} na način da odredimo kolika je vjerojatnost pojave bigrama koji počinje s riječi w_{n-1} . Vjerojatnost pojave općenito

N-grama je na taj način uvjetovana povješću riječi koje su mu prethodile. Ta vjerojatnost će biti to manja što se rijeđe pojavio niz riječi koji mu prethodi odnosno veća ukoliko se taj niz često pojavljuje u korpusu. Za bigrame ta vjerojatnost je jednaka:

$$\sum_{i:c(w_x w_i)=0} p^*(w_i w_x) = \frac{T(w_x)}{N(w_x) + T(w_x)} \quad (2.21)$$

gdje je T broj vrsta bigrama, a N broj oznaka bigrama s obzirom na prethodnu riječ. Formula 2.21 predstavlja ukupnu vjerojatnost svih bigrama frekvencije 0 koje trebamo razdjeliti na svaki bigram pojedinačno. Definiramo Z kao broj svih vrsta bigrama s frekvencijom 0 s obzirom na prethodnu riječ s kojom ćemo podijeliti ukupnu vjerojatnost:

$$Z(w_x) = \sum_{i:c(w_x w_i)=0} 1 \quad (2.22)$$

$$P^*(w_i | w_{i-1}) = \frac{T(w_{i-1})}{Z(w_{i-1})(N + T(w_{i-1}))} \quad \text{za } c_{w_{i-1} w_i} = 0 \quad (2.23)$$

Vjerojatnost preostalih bigrama čija frekvencija nije 0 je tada:

$$\sum_{i:c(w_x w_i)>0} P^*(w_i w_x) = \frac{c(w_x w_i)}{c(w_x) + T(w_x)} \quad (2.24)$$

Tablica 7. prikazuje bigrame windows haiku korpusa zaglađene Witten-Bell algoritmom.

Tablica 7. Bigrami windows haiku korpusa zaglađeni algoritmom Witten-Bell

| bigram | frekvencija | MLE | add-one | G-T | W-B |
|-------------|-------------|--------|---------|--------|--------|
| .<EOS> | 35 | 1.0000 | 0.1800 | 1.0000 | 0.9722 |
| <BOS> The | 3 | 0.0857 | 0.0200 | 0.0857 | 0.0625 |
| <BOS> You | 2 | 0.0571 | 0.0150 | 0.0122 | 0.0417 |
| is not | 2 | 0.2857 | 0.0174 | 0.1837 | 0.1538 |
| Your ire | 1 | 0.5000 | 0.0120 | 0.0297 | 0.2500 |
| You bring | 1 | 0.3333 | 0.0119 | 0.0198 | 0.2000 |
| not found | 1 | 0.2500 | 0.0118 | 0.0148 | 0.1250 |
| is the | 0 | 0.0000 | 0.0058 | 0.0012 | 0.0029 |
| This system | 0 | 0.0000 | 0.0060 | 0.0044 | 0.0031 |

2.3.2 Witten-Bell odstupanje

Prethodno opisan algoritam Witten-Bell, kao i Good-Turing te aditivno zaglađivanje s obzirom na način kako su definirani nazivaju se algoritmima snižavanja (engl. *discounting*). Rješavanju problema frekvencija vrijednosti 0 se može pristupiti i na drugi način i to tako da se poslužimo vrijednostima vjerojatnosti N-grama nižih redova. Postoje dva moguća pristupa kako se možemo osloniti na takvu hijerarhijsku strukturu N-grama, a to su odstupanje (engl. *backoff*) te tzv. izbrisana interpolacija (engl. *deleted interpolation*). Odstupanje je nelinearna metoda modeliranja N-grama gdje stvaramo model N-grama reda N na osnovu modela reda N-1. Metoda radi na način da ukoliko imamo N-gram frekvencije 0 tada njegovu vjerojatnost računamo interpolacijom N-grama nižeg reda pomnoženog s odgovarajućim faktorom. Taj postupak kod metode odstupanja za razliku od izbrisane interpolacije radimo samo u slučaju N-grama frekvencije 0 dok za ostale N-grame vjerojatnost računamo na uobičajen način. Metoda odstupanja se za trigrame tako može definirati na način:

$$\hat{P}(w_i|w_{i-2}w_{i-1}) = \begin{cases} P(w_i|w_{i-2}w_{i-1}) & \text{ako } C(w_{i-2}w_{i-1}w_i) > 0 \\ \alpha_1 P(w_i|w_{i-1}) & \text{ako } C(w_{i-2}w_{i-1}w_i) = 0 \text{ i } C(w_{i-1}w_i) > 0 \\ \alpha_2 P(w_i) & \text{inače} \end{cases} \quad (2.24)$$

dok je općenita formula za metodu snižavanja dana kao:

$$\hat{P}(w_n|w_{n-N+1}^{n-1}) = \tilde{P}(w_n|w_{n-N+1}^{n-1}) + \theta(P(w_n|w_{n-N+1}^{n-1}))\alpha(w_{n-N+1}^{n-1})\hat{P}(w_n|w_{n-N+2}^{n-1}) \quad (2.25)$$

U prethodnoj formuli θ označava kada treba odstupiti na N-gram nižeg reda što se događa kada je N-gram višeg reda frekvencije 0.

$$\theta(x) = \begin{cases} 1 & \text{ako } x = 0 \\ 0 & \text{inače} \end{cases} \quad (2.26)$$

Faktor α služi kako bi se ukupna vjerojatnost dovela do vrijednosti 1. Razlog je taj što odstupanjem na N-grame nižeg reda dodajemo određeni iznos vjerojatnosti te stoga ukupan zbroj vjerojatnosti neće biti 1. \tilde{P} je snižena vjerojatnost dobivena preko MLE, ostatak vjerojatnosti se odnosi na N-grame nižeg reda prilikom odstupanja. Zaglađivanje obično koristi postupak odstupanja zajedno s postupkom snižavanja. Postupak snižavanja će pritom odrediti koliko se vjerojatnosti ukupno treba pridjeliti N-gramima frekvencije 0 dok će postupak odstupanja odrediti na koji način će se ta vjerojatnost rasporediti na pojedinačne N-grame umjesto ravnomjernog raspoređivanja.

Jedan od algoritama odstupanja je Witten-Bellov algoritam odstupanja razvijen za potrebe kompresije teksta te se može smatrati posebnom inačicom algoritma zaglađivanja Jelinek-Mercer. Model zaglađivanja N-tog reda je definiran rekursivno kao linearna interpolacija između MLE N-tog reda i zaglađenog modela N-1 reda:

$$P_{WB}(w_n|w_{n-N+1}^{n-1}) = \alpha_{w_{n-N+1}^{n-1}} P_{MLE}(w_n|w_{n-N+1}^{n-1}) + (1 - \alpha_{w_{n-N+1}^{n-1}}) P_{WB}(w_n|w_{n-N+2}^{n-1}) \quad (2.27)$$

Kako bismo izračunali parametre $\alpha_{w_{n-N+1}^{n-1}}$ potrebno je znati broj jedinstvenih riječi koje slijede nakon niza riječi w_{n-N+1}^{n-1} . Ta se vrijednost definira kao:

$$N_{1+}(w_{n-N+1}^{n-1} \bullet) = |\{w_n : c(w_{n-N+1}^{n-1} w_n) > 0\}| \quad (2.28)$$

Notacija N_{1+} označava riječi čiji je broj 1 ili više dok \bullet označava slobodnu varijablu po kojoj se vrši zbrajanje. Parametari $\alpha_{w_{n-N+1}^{n-1}}$ se sada mogu definirati kao:

$$1 - \alpha_{w_{n-N+1}^{n-1}} = \frac{N_{1+}(w_{n-N+1}^{n-1} \bullet)}{N_{1+}(w_{n-N+1}^{n-1} \bullet) + \sum_{w_n} c(w_{n-N+1}^n)} \quad (2.29)$$

Supstitucijom svih vrijednosti formula 2.27 se sada može pisati:

$$P_{WB}(w_n | w_{n-N+1}^{n-1}) = \frac{c(w_{n-N+1}^n) + N_{1+}(w_{n-N+1}^{n-1} \bullet) P_{WB}(w_n | w_{n-N+2}^{n-1})}{\sum_{w_n} c(w_{n-N+1}^n) + N_{1+}(w_{n-N+1}^{n-1} \bullet)} \quad (2.30)$$

Formula 2.27 se može opisati na način da sa vrijednošću $\alpha_{w_{n-N+1}^{n-1}}$ koristimo model višeg reda dok s vrijednošću $1 - \alpha_{w_{n-N+1}^{n-1}}$ koristimo model nižeg reda. Tada je vrijednost $1 - \alpha_{w_{n-N+1}^{n-1}}$ vjerojatnost pojave riječi koje se do tada nije pojavila nakon slijeda riječi w_{n-N+1}^{n-1} . Frekvencija takvih riječi se može procijeniti brojanjem koliko puta se riječ koja je uslijedila nakon slijeda w_{n-N+1}^{n-1} razlikovala od svih dotadašnjih riječi koje su uslijedile nakon w_{n-N+1}^{n-1} . Taj broj je upravo jednak broju jedinstvenih riječi koje slijede w_{n-N+1}^{n-1} odnosno $N_{1+}(w_{n-N+1}^{n-1} \bullet)$.

2.4 Apsolutno snižavanje

Apsolutno snižavanje kao i Witten-Bellovo zaglađivanje uključuje interpolaciju modela viših i nižih redova. Za razliku od Witten-Bella umjesto množenja modela višeg reda faktorom $\alpha_{w_{n-N+1}^{n-1}}$ distribucija vjerojatnosti višeg reda se dobiva oduzimanjem fiksnog iznosa $D \leq 1$ od svake frekvencije N-grama koja nije 0 pa tako umjesto formule 2.27:

$$P_{WB}(w_n | w_{n-N+1}^{n-1}) = \alpha_{w_{n-N+1}^{n-1}} P_{MLE}(w_n | w_{n-N+1}^{n-1}) + (1 - \alpha_{w_{n-N+1}^{n-1}}) P_{WB}(w_n | w_{n-N+2}^{n-1})$$

imamo sljedeće:

$$P_{ABS}(w_n|w_{n-N+1}^{n-1}) = \frac{\max\{c(w_{n-N+1}^n) - D, 0\}}{\sum_{w_n} c(w_{n-N+1}^n)} + (1 - \alpha_{w_{n-N+1}^{n-1}}) P_{ABS}(w_n|w_{n-N+2}^{n-1}) \quad (2.31)$$

Kako bi ukupan zbroj vjerojatnosti bio 1 faktor $(1 - \alpha_{w_{n-N+1}^{n-1}})$ iznosi:

$$(1 - \alpha_{w_{n-N+1}^{n-1}}) = \frac{D}{\sum_{w_n} c(w_{n-N+1}^n)} N_{1+}(w_{n-N+1}^{n-1}) \quad (2.32)$$

gdje je $N_{1+}(w_{n-N+1}^{n-1})$ iznos definiran formulom 2.28 i uz pretpostavku $0 \leq D \leq 1$. Sugerira se koristiti D dobiven preko izbrisane procjene (engle. *deleted estimation*) trening podataka. Ney, Essen i Kneser (1994) su tako došli do procjene za D koja iznosi:

$$D = \frac{N_1}{N_1 + 2N_2} \quad (2.33)$$

gdje N označava model višeg reda koji se interpolira, a N_1 i N_2 predstavljaju broj N -grama čije su frekvencije točno 1 odnosno 2. Faktor iz 2.32 se može promatrati isto kao i kod Witten-Bell zaglađivanja iz 2.29 kao aproksimacija iste vrijednosti, vjerojatnosti pojave riječi koje se do tada nije pojavila nakon određenog prethodnog slijeda riječi.

2.5 Kneser-Ney zaglađivanje

Kneser-Ney algoritam zaglađivanja predstavlja nadogradnju algoritma apsolutnog snižavanja gdje se kombiniranje modela N -grama višeg i nižeg reda ostvaruje na drugačiji način od prethodno spomenutih algoritama gdje je distribucija vjerojatnosti modela nižeg reda značajan faktor samo u slučaju frekvencija iznosa 0 ili vrlo malog broja. Ideja je optimizirati izračun vjerojatnosti N -grama nižeg reda ukoliko N -gram višeg reda nije zapažen u korpusu. Motivacija za to je što recimo koristeći algoritam apsolutnog snižavanja s odstupanjem se ne uzima u obzir informacija da N -gram višeg reda nije zapažen u korpusu u postupku odstupanja i računanja vjerojatnosti N -grama nižeg reda. Tako na primjer Chen i Goodman

ističu riječ *Francisco* koja se pojavljuje relativno često ali samo u slučaju ukoliko joj prethode nekolicina određenih riječi kao što je to npr. riječ *San* u nazivu *San Francisco*. Riječ *Francisco* je česta pa će joj stoga algoritam apsolutnog snižavanja pridjeliti visoku vjerojatnost iako se ona može pojaviti tek u nekolicini N-grama višeg reda što zapravo sugerira nižu vjerojatnost. Taj podatak je značajan ukoliko N-gram višeg reda (bigram) ne postoji dok mu je N-gram nižeg reda (unigram) *Francisco*. Ukoliko bigram postoji tada se odstupanje ne radi i vjerojatnost se izračunava na osnovu njegove frekvencije. Iz prethodnog razmatranja vjerojatnost unigrama ne bi smjela ovisiti o frekvenciji pojave unigrama već o broju različitih riječi nakon koje može uslijediti. U originalnoj verziji algoritma distribucija nižeg reda bi se trebala definirati na način da granice zaglađene distribucije vjerojatnosti odgovaraju granicama podataka za trening. Drugim riječima na primjeru modela bigrama potrebno je definirati zaglađenu distribuciju vjerojatnosti P_{KN} da zadovoljava uvjet:

$$\sum_{w_{n-1}} P_{KN}(w_{n-1}w_n) = \frac{c(w_n)}{\sum_{w_n} c(w_n)} \quad (2.34)$$

Chen i Goodman predstavljaju varijaciju osnovnog Kneser-Ney zaglađivanja transformirajući ga u interpoliranu verziju koja se pritom pokazala boljom od verzije odstupanja. Prilikom računanja vjerojatnosti unigrama interpolacija nije moguća s obzirom da ne postoji model nižeg reda pa se vjerojatnost definira kao:

$$P_{KN}(w_n) = \frac{N_{1+}(\bullet w_n)}{N_{1+}(\bullet\bullet)} \quad (2.35)$$

Gdje je $N_{1+}(\bullet w_n)$ broj različitih riječi koje prethode w_n barem jedanput u korpusu:

$$N_{1+}(\bullet w_n) = |\{w_{n-1} : c(w_{n-1}^n) > 0\}| \quad (2.36)$$

$$N_{1+}(\bullet\bullet) = |\{w_{n-1}, w_n : c(w_{n-1}^n) > 0\}| = \sum_{w_n} N_{1+}(\bullet w_n) \quad (2.37)$$

Vjerojatnost N-grama višeg reda ali ne i najvišeg je dana izrazom:

$$P_{KN}(w_n | w_{n-N+1}^{n-1}) = \frac{\max\{N_{1+}(\bullet w_{n-N+1}^n) - D, 0\}}{N_{1+}(\bullet w_{n-N+1}^{n-1} \bullet)} + \frac{D}{N_{1+}(\bullet w_{n-N+1}^{n-1} \bullet)} N_{1+}(w_{n-N+1}^{n-1} \bullet) P_{KN}(w_n | w_{n-N+2}^{n-1}) \quad (2.38)$$

gdje je

$$N_{1+}(\bullet w_{n-N+1}^n) = |\{w_{n-N} : c(w_{n-N}^n) > 0\}| \quad (2.39)$$

$$N_{1+}(\bullet w_{n-N+1}^{n-1} \bullet) = |\{w_{n-N}, w_n : c(w_{n-N}^n) > 0\}| = \sum_{w_n} N_{1+}(\bullet w_{n-N+1}^n) \quad (2.40)$$

Vrijednost D je ista kao i kod apsolutnog snižavanja definirana s formulom 2.33.

Faktor $\frac{D}{N_{1+}(\bullet w_{n-N+1}^{n-1} \bullet)} N_{1+}(w_{n-N+1}^{n-1} \bullet)$ slično kao i $(1 - \alpha_{w_{n-N+1}^{n-1}})$ kod Witten-Bella određuje utjecaj N-grama nižeg reda na ukupnu vjerojatnost. Predstavlja dio vjerojatnosti koji je oduzet od vjerojatnosti N-grama višeg reda snižavajući ga za iznos D. Izraz $N_{1+}(\bullet w_{n-N+1}^{n-1} \bullet)$ je broj različitih riječi koje su prethodile w_{n-N+1}^n . Pošto distribucija vjerojatnosti nižeg reda ima veći utjecaj ukoliko su N-grami višeg reda manjih iznosa frekvencija te ukoliko imaju veći broj različitih riječi koje im prethode potrebno je takvu distribuciju pomnožiti s faktorom $N_{1+}(w_{n-N+1}^{n-1} \bullet)$. N-grami najvišeg reda umjesto frekvencija zasnovanih na brojanju slijeda riječi kao u 2.38 koriste uobičajene apsolutne iznose frekvencija što odgovara N-gramima najvišeg reda kod apsolutnog snižavanja:

$$P_{KN}(w_n | w_{n-N+1}^{n-1}) = \frac{\max\{c(w_{n-N+1}^n) - D, 0\}}{c(w_{n-N+1}^{n-1})} + \frac{D}{c(w_{n-N+1}^{n-1})} N_{1+}(w_{n-N+1}^{n-1} \bullet) P_{KN}(w_n | w_{n-N+2}^{n-1}) \quad (2.41)$$

U slučaju kada je unigram N-gram najvišeg reda tada je Kneser-Ney algoritam jednak MLE pošto se koriste apsolutne frekvencije i nema interpolacije između modela. Primjer koji slijedi demonstrira Kneser-Ney algoritam.

Pretpostavimo da imamo sljedeći korpus:

<BOS> B S S <EOS>

Ukoliko radimo s modelom bigrama počinjemo s $P(B | < BOS >)$

$$P_{KN}^2(B | < BOS >) = \frac{\max\{c(< BOS >, B) - D, 0\}}{\sum_{w_n} c(< BOS >, w_n)} + \frac{D}{\sum_{w_n} c(< BOS >, w_n)} N_{1+}(< BOS > \bullet) P_{KN}^1(B) \quad (2.42)$$

gdje je

$$P_{KN}^1(B) = \frac{\max\{N_{1+}(\bullet B) - D, 0\}}{\sum_{w_n} N_{1+}(\bullet, w_n)} + \frac{D}{\sum_{w_n} N_{1+}(\bullet \bullet)} N_{1+}(\bullet) P_{KN}^0(B) \quad (2.43)$$

te

$$P_{KN}^0(B) = \frac{1}{V} = \frac{1}{3} \quad (2.44)$$

Izraz $N_{1+}(\bullet) = V$ jednak je veličini vokabulara dok je izraz $\max\{N_{1+}(\bullet B) - D, 0\}$ za slučaj unigrama jednak $N_{1+}(\bullet B) - D$ pošto će se on uvijek uočiti barem jednom što daje pojednostavljenu formulu:

$$P_{KN}^1(B) = \frac{N_{1+}(\bullet B)}{\sum_{w_n} N_{1+}(\bullet, w_n)} \quad (2.45)$$

Formula za model bigrama sada je:

$$P_{KN}^2(B|<BOS>) = \frac{\max\{c(<BOS>,B) - D, 0\}}{\sum_{w_n} c(<BOS>,w_n)} + \frac{D}{\sum_{w_n} c(<BOS>,w_n)} N_{1+}(<BOS>•) \frac{N_{1+}(•B)}{\sum_{w_n} N_{1+}(•,w_n)} \quad (2.46)$$

Na primjeru za $P(B|<BOS>)$ i uz $D = 0.2$ imamo:

$$c(<BOS>,B) = 1 \quad (2.47)$$

$$\sum_{w_n} c(<BOS>,w_n) = 1 \quad (2.48)$$

$$N_{1+}(<BOS>•) = 1 \quad (2.49)$$

$$N_{1+}(•B) = 1 \quad (2.50)$$

$$\sum_{w_n} N_{1+}(•,w_n) = 4 \quad (2.51)$$

što kao rezultat daje

$$P_{KN}^2(B|<BOS>) = \frac{1 - 0.2}{1} + \frac{0.2}{1} \times 1 \times \frac{1}{4} = 0.85 \quad (2.52)$$

slično za preostale vjerojatnosti

$$P_{KN}^2(S|<BOS>) = \frac{0}{1} + \frac{0.2}{1} \times 1 \times \frac{2}{4} = 0.1 \quad (2.53)$$

$$P_{KN}^2(<EOS><BOS>) = 0.05 \quad (2.54)$$

2.6 Modificirano Kneser-Ney zaglađivanje

Modificirani oblik Kneser-Ney zaglađivanja uvode Chen i Goodman te je algoritam polučio odlične rezultate. Umjesto korištenja samo jednog parametra snižavanja D za sve frekvencije vrijednosti 0 kao u originalnom Kneser-Ney algoritmu imamo tri parametra, D_1 , D_2 , D_{3+} koje se redom odnose na N-grame frekvencija jedan, dva te tri ili više pa tako umjesto formule za osnovni oblik Kneser-Ney algoritma koristimo:

$$P_{MKN}(w_n | w_{n-N+1}^{n-1}) = \frac{c(w_{n-N+1}^n) - D(c(w_{n-N+1}^n))}{\sum_{w_n} c(w_{n-N+1}^n)} + \gamma(w_{n-N+1}^{n-1}) P_{MKN}(w_i | w_{n-N+2}^{n-1}) \quad (2.55)$$

gdje je

$$D(c) = \begin{cases} 0 & \text{ako } c = 0 \\ D_1 & \text{ako } c = 1 \\ D_2 & \text{ako } c = 2 \\ D_{3+} & \text{ako } c > 2 \end{cases} \quad (2.56)$$

Parametri D_1 , D_2 , D_{3+} i Y modela su definirani kao:

$$\begin{aligned} Y &= \frac{N_1}{N_1 + 2N_2} \\ D_1 &= 1 - 2Y \frac{N_2}{N_1} \\ D_2 &= 2 - 3Y \frac{N_3}{N_1} \\ D_{3+} &= 3 - 4Y \frac{N_4}{N_3} \end{aligned} \quad (2.57)$$

Prethodni parametri nastali su po uzoru na procjenu za optimalni D koju su za potrebe apsolutnog snižavanja te Kneser-Ney algoritma razvili Ney, Essen i Kneser (1994). Formula za vjerojatnost unigrama $P_{MKN}(w_n)$ se za modificirani Kneser-Ney algoritam definira jednako kao i za osnovni algoritam po formuli 2.35. N-grami višeg reda se računaju po formuli:

$$P_{MKN}(w_n|w_{n-N+1}^{n-1}) = \frac{\max\{N_{1+}(\bullet w_{n-N+1}^{n-1}) - D(c(w_{n-N+1}^n)), 0\}}{N_{1+}(\bullet w_{n-N+1}^{n-1})} + \gamma_{niži}(w_{n-N+1}^{n-1})P_{MKN}(w_n|w_{n-N+2}^{n-1}) \quad (2.58)$$

gdje se faktor koji će učiniti da zbroj vjerojatnosti bude 1 $\gamma_{niži}$ definira kao:

$$\gamma_{niži} = \frac{D_1 N_1(w_{n-N+1}^{n-1} \bullet) + D_2 N_2(w_{n-N+1}^{n-1} \bullet) + D_{3+} N_{3+}(w_{n-N+1}^{n-1} \bullet)}{N_{1+}(\bullet w_{n-N+1}^{n-1})} \quad (2.59)$$

Faktori $N_1(w_{n-N+1}^{n-1} \bullet)$, $N_2(w_{n-N+1}^{n-1} \bullet)$ i $N_{3+}(w_{n-N+1}^{n-1} \bullet)$ su pritom jednaki:

$$\begin{aligned} N_1(w_{n-N+1}^{n-1} \bullet) &= |\{w_n: c(w_{n-N+1}^n) = 1\}| \\ N_2(w_{n-N+1}^{n-1} \bullet) &= |\{w_n: c(w_{n-N+1}^n) = 2\}| \\ N_{3+}(w_{n-N+1}^{n-1} \bullet) &= |\{w_n: c(w_{n-N+1}^n) > 2\}| \end{aligned} \quad (2.60)$$

Formula za N-gram najvišeg reda glasi:

$$P_{MKN}(w_n|w_{n-N+1}^{n-1}) = \frac{\max\{c(w_{n-N+1}^n) - D(c(w_{n-N+1}^n)), 0\}}{c(w_{n-N+1}^{n-1})} + \gamma_{viši}(w_{n-N+1}^{n-1})P_{MKN}(w_n|w_{n-N+2}^{n-1}) \quad (2.61)$$

gdje je faktor $\gamma_{viši}$ jednak:

$$\gamma_{viši}(w_{n-N+1}^{n-1}) = \frac{D_1 N_1(w_{n-N+1}^{n-1} \bullet) + D_2 N_2(w_{n-N+1}^{n-1} \bullet) + D_{3+} N_{3+}(w_{n-N+1}^{n-1} \bullet)}{c(w_{n-N+1}^{n-1})} \quad (2.62)$$

Modificirani Kneser-Ney algoritam nije u potpunosti definiran. Algoritam ostavlja parametre snižavanja D otvorenim za slobodnu interpretaciju. Postoje barem dva načina za njihov odabir od kojih je jedan na osnovu vrijednosti frekvencija, a drugi na osnovu proširenog konteksta N-grama. Također moguće je koristiti različite načine za njihov izračun od kojih je jedan od načina opisan u prethodnom tekstu s

2.57. Također je moguće implementirati algoritam tako da se N-grami frekvencije 1 tretiraju kao i N-grami frekvencije 0 na svim razinama modela.

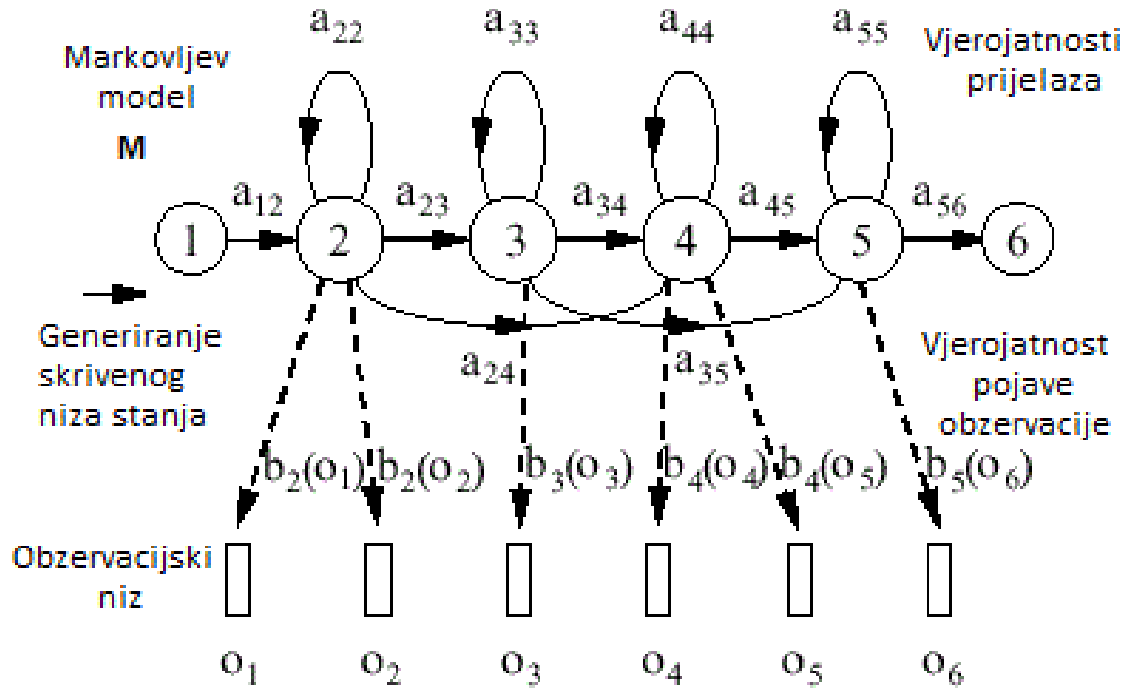
3. Skriveni Markovljev model

Prethodno opisani algoritmi te modeli N-grama se široko koriste u obradi prirodnog jezika te predstavljaju samo jedan dio cjelokupnog procesa obrade. Jedan od tih dijelova je i skriveni Markovljev model (engl. *Hidden Markov Model*, *HMM*). Općenito Markovljev lanac je poseban oblik težinskog automata u kojem ulazni niz određuje kroz koja stanja će automat prolaziti. Kod tog modela znamo u koje stanje treba prijeći na osnovu ulaznog niza. Za razliku kod skrivenog Markovljevog modela to nismo u stanju učiniti zbog toga što ulazni niz jedinstveno ne određuje u koje sljedeće stanje treba prijeći. Skriveni Markovljev model se razlikuje od osnovnog Markovljevog modela tako što dodaje dva dodatna zahtjeva. Prvi je postojanje odvojenog skupa simbola obzervacija O koji ne proizlazi iz iste abecede kao stup stanja Q . Drugi je taj da funkcija vjerojatnosti obzervacija B nije ograničena samo na vrijednosti 0 i 1 već kod skrivenog Markovljevog modela može poprimiti bilo koju vrijednost između 0 i 1. Također stanjima je dozvoljeno generiranje većeg broja iste obzervacije. Kako bi definirali skriveni Markovljev model potrebni su sljedeći parametri:

- stanja: Skup stanja $Q = q_1 q_2 \dots q_N$.
- vjerojatnosti prijelaza: Skup vjerojatnosti $A = a_{01} a_{02} \dots a_{n1} \dots a_{nn}$. Svaka vrijednost a_{ij} predstavlja vjerojatnost prijelaza iz stanja i u stanje j . Skup se prikazuje matricom koja se naziva matrica vjerojatnosti prijelaza.
- vjerojatnosti obzervacija: Skup vjerojatnosti obzervacija $B = b_i(o_t)$ gdje svaka predstavlja vjerojatnost pojave obzervacije o_t iz stanja i .
- početna distribucija: Početna distribucija vjerojatnosti stanja, π takva da je π_i vjerojatnost da će se HMM početno nalaziti u stanju i . Stanja koja ne mogu biti početna stanja HMM-a će imati vrijednost $\pi_i = 0$.
- prihvatljiva stanja: Skup dopuštenih, prihvatljivih stanja.

Skup simbola koji predstavljaju ulazne podatke modela ili oni koji su nastali kao izlaz modela nazivaju se obzervacijski niz ili sekvenca $O = (o_1 o_2 o_3 \dots o_T)$.

Na slici 3. ilustriran je princip rada skrivenog Markovljevog modela.



Slika 3. Skriveni Markovljev model

4. Viterbijev algoritam

Postoji nekoliko problema koje susrećemo koristeći se skrivenim Markovljevim modelom (*HMM*). Jedan od njih je problem dekodiranja odnosno pronalaska najvjerojatnijeg puta kroz model tj. pronalaska niza skrivenih stanja $q = (q_1 q_2 q_3 \dots q_t)$ ukoliko na raspolaganju imamo observacijski niz $o = (o_1 o_2 o_3 \dots o_t)$. Za rješavanje tog problema koristimo Viterbijev algoritam koji nam može dati odgovor koji je to niz skrivenih stanja na osnovu prethodno izgrađenog modela N-grama i konstruiranog HMM-a. Tako recimo za ulazni observacijski niz *Tomislav voli igrati nogomet* niz skrivenih stanja tj. vrsta riječi bi mogao biti *N V V N* - (pritom je N imenica, a V glagol). Pri traženju najvjerojatnijeg puta koristimo se matricom gdje svako polje $viterbi[i, t]$ matrice sadrži vjerojatnost najboljeg puta koji se pronalazi u prvih t observacija i koji završava u stanju i HMM-a. Taj put je najvjerojatniji put od svih mogućih nizova stanja duljine $t - 1$:

$$viterbi[t, i] = \max_{q_1 q_2 \dots q_{t-1}} P(q_1 q_2 \dots q_{t-1} q_t = i, o_1, o_2 \dots o_t | \lambda)$$

(4.1)

λ u prethodnom izrazu označava model (težinski automat ili graf stanja). Jedan od mogućih pristupa određivanja najboljeg odnosno najvjerojatnijeg puta bio bi provjeriti sve puteve u HMM-u i odabrati onog s najvećom vjerojatnosti. Takav pristup nije najbolji iz razloga što je takva zadaća izuzetno složena i računski zahtjevna. Kao moguće rješenje nameće se uporaba dinamičkog programiranja. Postupak pojednostavljuje problem ali se uvodi pogrešna pretpostavka da ukoliko najbolji put za cijeli obzervacijski niz prolazi stanjem q_i tada taj put također sadrži najbolji put i do stanja q_i . Ne mora značiti da je najbolji put u nekom trenutku t najbolji put za cijeli obzervacijski niz. Konačni najbolji put može u početku biti lošiji od ostalih mogućih puteva u tom trenutku t no na kraju se može pokazati da je upravo taj put najbolji mogući. Razlog ovog pojednostavljenja je taj što nam omogućuje razlaganje računanja ukupno najboljeg puta u manje cjeline. Svaki od najboljih puteva u trenutku t je najbolji produžetak od svih puteva koji završavaju u trenutku $t - 1$:

$$viterbi[t, j] = \max_i (viterbi[t - 1, i] a_{ij}) b_j(o_t) \quad (4.2)$$

Za svaki stupac matrice odnosno za svaki vremenski indeks t svako polje matrice $viterbi[t, j]$ će sadržavati vjerojatnost najvjerojatnijeg puta koji završava u tom polju. Ta se vrijednost izračunava rekurzivno maksimizacijom vjerojatnosti dolaska u tu ćeliju iz svih prethodnih stanja. Također prilikom postupka održavamo niz stanja koji u trenutku t čine najbolji mogući put koji se pritom naziva najbolji parcijalni put. Zadnji korak algoritma dat će najbolji mogući put kroz model te njegovu vjerojatnost. Sada možemo u potpunosti definirati algoritam i podijeliti ga u tri dijela.

- Inicijalizacija:

$$viterbi[1, j] = a_{0j} b_j(o_1) \quad (4.3)$$

$$bt_1(j) = 0 \quad (4.4)$$

- Rekurzija:

$$viterbi[t, j] = \max_{i=1 \dots N} (viterbi[t-1, i] a_{ij}) b_j(o_t), \quad 1 \leq j \leq N, 1 < t \leq T \quad (4.5)$$

$$bt_t(j) = \operatorname{argmax}_{i=1 \dots N} (viterbi[t-1, i] a_{ij}) b_j(o_t), \quad 1 \leq j \leq N, 1 < t \leq T \quad (4.6)$$

- Zaustavljanje:

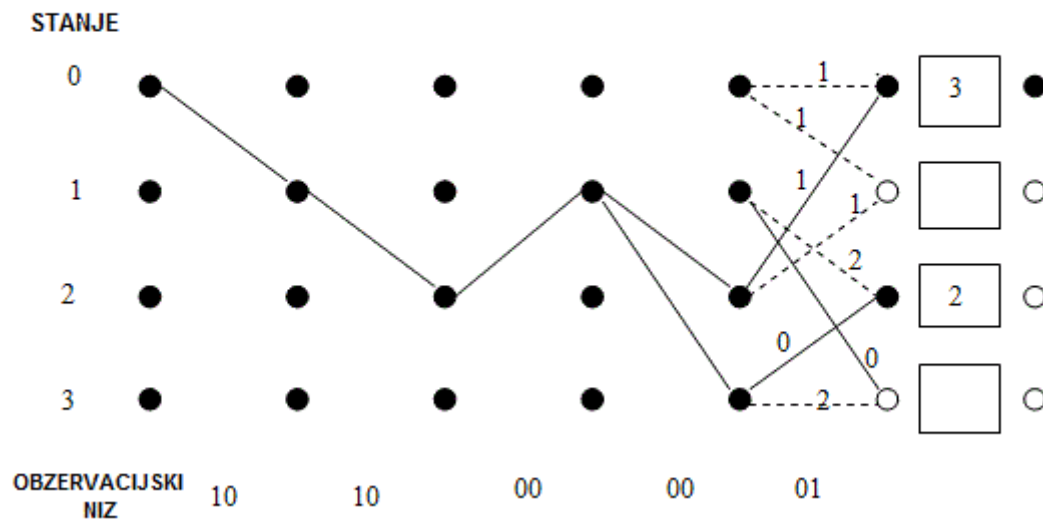
$$P^* = viterbi[t, q_F] = \max_{i=1 \dots N} (viterbi[T, i] a_{iF}) \quad (4.7)$$

$$q_T^* = bt_T(q_F) = \operatorname{argmax}_{i=1 \dots N} (viterbi[T, i] a_{iF}) \quad (4.8)$$

Funkcija koja provodi Viterbijev algoritam opisana je sljedećim pseudokôdom:

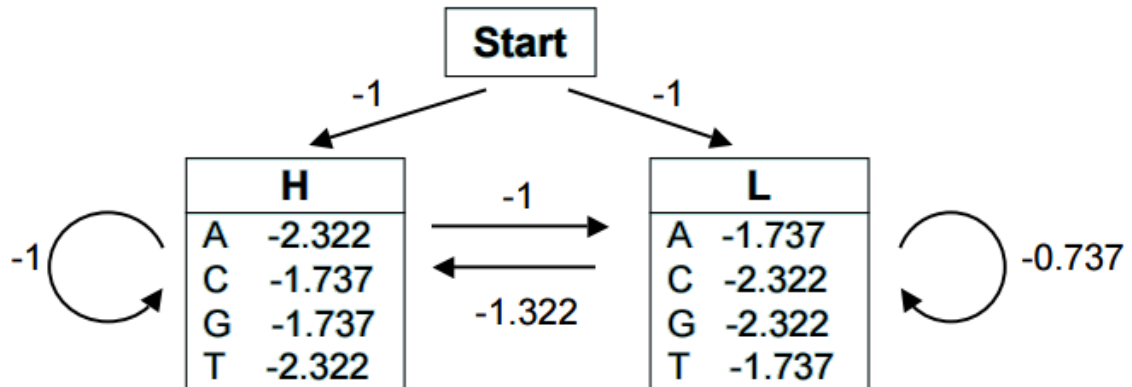
funkcija VITERBI (*obzervacije* duljine T , *automat stanja*) **vraća** najbolji – put
broj – stanja \leftarrow BROJ – STANJA(*automat – stanja*)
 Stvaranje matrice vjerojatnosti puteva $viterbi[broj – stanja + 2, T + 2]$
 $viterbi[0, 0] \leftarrow 1.0$
za svaki vremenski trenutak t **od** 0 **do** T **čini**
 za svako stanje s **od** 0 **do** *broj – stanja* **čini**
 za svaki prijelaz s' iz s definiran *automatom – stanja*
 novi – iznos $\leftarrow viterbi[s, t] * a[s, s'] * b_{s'}(o_t)$
 ako $((viterbi[s', t-1] = 0) \vee (novi – iznos > viterbi[s', t+1]))$
 tada
 $viterbi[s', t+1] \leftarrow novi – iznos$
 povratni – pokazivač $[s', t+1] \leftarrow s$
 Prati unatrag pokazivač od stanja najveće vjerojatnosti u zadnjem polju i
 vrati najbolji put

Ilustracija algoritma nalazi se na slici 4.



Slika 4. Postupak pronalaska najboljeg puta Viterbijevim algoritmom

Sljedeći primjer prikazuje način izračuna najboljeg puta Viterbijevim algoritmom



Slika 5. Automat stanja kojeg koristi Viterbijev algoritam

Definirano je početno stanje *START*, stanja *H* i *L* te vjerojatnosti prijelaza između stanja u logaritamskom prostoru Log_2 . Definirane su observacije *A*, *C*, *G* i *T* kao i vjerojatnosti observacija da se nađu u nekom od stanja.

Pretpostavimo da imamo ulazni niz observacija CAGTCT. Izračun najboljeg puta nalazi se u tablici 8.

Tablica 8. Izračun najboljeg puta pomoću Viterbijevog algoritma

| | C | A | G | T | C | T |
|---|--------|--------|--------|---------|---------|---------|
| H | -2.737 | -6.059 | -8.211 | -11.533 | -14.007 | -17.651 |
| L | -3.322 | -5.474 | -8.533 | -10.948 | -14.007 | -16.481 |

Najveća vjerojatnost je $2^{-16.481} = 1.0933 \times 10^{-5}$ dok je najbolji put niz skrivenih stanja HLHLLL.

5. Vrednovanje modela

Nakon što su izgrađeni statistički jezični modeli koristeći se različitim tehnikama opisanim u poglavlju 2 potrebno ih je na neki način usporediti i dati odgovor koji daje bolju procjenu stvarne distribucije vjerojatnosti. Neki od načina usporedbe jezičnim modela opisani su u nastavku.

5.1 Entropija

Entropija i perpleksija su najčešće korištene metrike za vrednovanje jezičnog modela zasnovanog na N-gramima. Entropija je količina informacije te je nezamjenjiva u područjima obrade prirodnog jezika, raspoznavanju govora i računalnoj lingvistici. Koristi se kako bi dala odgovor koliko je informacije sadržano u nekoj gramatici, koliko dobro neka gramatika odgovara jeziku te koliko dobro je gramatika N-grama u stanju procijeniti koja će biti sljedeća riječ u nizu. U stanju smo procijeniti u kojoj mjeri neki vjerojatnosni model odgovara stvarnom modelu, na taj način možemo i usporediti koji od vjerojatnosnih modela daje bolje rezultate od drugog. Entropija se definira kao:

$$H(X) = - \sum_{x \in \chi} p(x) \log_2 p(x) \quad (5.1)$$

Gdje je χ slučajna varijabla koja se proteže preko onoga što predviđamo, to može biti slovo, riječ, rečenica itd. Ukoliko koristimo \log_2 entropija je tada donja granica prosječnog broja bitova potrebnog za šifriranje poruke. Umjesto računanja entropije jedne varijable željeli bismo izračunati entropiju niza riječi

$W = \{... w_0, w_1, w_2, ... w_n\}$. Slučajna varijabla u tom slučaju mora obuhvatiti cijeli niz riječi jezika L pa tako imamo:

$$H(w_1, w_1, ..., w_n) = - \sum_{W_1^n \in L} p(W_1^n) \log p(W_1^n) \quad (5.2)$$

Također možemo definirati entropiju po riječi kao entropiju niza riječi podijeljenu s brojem riječi u nizu:

$$\frac{1}{n} H(W_1^n) = - \frac{1}{n} \sum_{W_1^n \in L} p(W_1^n) \log p(W_1^n) \quad (5.3)$$

Kako bismo pak odredili entropiju cijelog jezika L moramo pretpostaviti nizove riječi beskonačne duljine:

$$H(L) = \lim_{n \rightarrow \infty} \frac{1}{n} H(w_1, w_2, ..., w_n) = \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{W \in L} p(w_1, ..., w_n) \log p(w_1, ..., w_n) \quad (5.4)$$

Prema Shannon-McMillan-Breimanovu teoremu ukoliko je jezik regularan odnosno stacionaran i ergodičan možemo pisati:

$$H(L) = \lim_{n \rightarrow \infty} - \frac{1}{n} \log p(w_1 w_2 ... w_n) \quad (5.5)$$

Slučajan proces je stacionaran i ergodičan ukoliko je vremensko usrednjavanje jednako statističkom usrednjavanju odnosno na primjeru riječi distribucija vjerojatnosti za riječi u trenutku t je ista kao i distribucija vjerojatnosti u trenutku $t + 1$. Markovljevi modeli, a time i N-grami su stacionarni jer vjerojatnost N-grama u nekom trenutku t ovisi o vjerojatnosti u trenutku $t - 1$. To se razlikuje od prirodnog jezika koji nije stacionaran stoga statistički model daje samo aproksimaciju točne distribucije vjerojatnosti i entropije prirodnog jezika. Za usporebu vjerojatnosnih modela koristi se poseban oblik entropije koji se još naziva unakrsna entropija (engl. *cross entropy*). Ona se koristi u situacijama kada

ne znamo stvarnu distribuciju vjerojatnosti p nekog skupa podataka što može biti i prirodni jezik. Koristimo model m koji je aproksimacija stvarne distribucije vjerojatnosti p . Unakrsna entropija modela m za distribuciju vjerojatnosti p je:

$$H(p, m) = \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{w \in L} p(w_1, \dots, w_n) \log m(w_1, \dots, w_n) \quad (5.6)$$

Pošto je vjerojatnosni model N-grama stacionaran možemo pisati:

$$H(p, m) = \lim_{n \rightarrow \infty} -\frac{1}{n} \log m(w_1 w_2 \dots w_n) \quad (5.7)$$

Razlog zbog čega se unakrsna entropija koristi za usporedbu modela je i taj što je ona gornja granica stvarne entropije $H(p)$. Tako za svaki model m vrijedi:

$$H(p) \leq H(p, m) \quad (5.8)$$

Razlika između $H(p)$ i $H(p, m)$ je mjera koliko dobro model m vrši aproksimaciju. Model je točniji ukoliko je ta razlika što manja pa tako uspoređujući dva modela m_1 i m_2 točniji je onaj s nižom vrijednosti unakrsne entropije. Prilikom usporedbe modela mjera koja se često koristi je 2^H koja se naziva perpleksija. Perpleksija se može shvatiti kao mjera koliko dobro distribucija vjerojatnosti ili vjerojatnosni model predviđaju neki uzorak u odnosu na stvarnu distribuciju vjerojatnosti p . Kako bismo usporedili dva statistička modela N-grama m temeljem njihovih perpleksija potrebno je pripremiti korpus na odgovarajući način. Korpus je potrebno podijeliti u dva dijela, jedan dio će se koristiti za trening modela dok će se drugi koristiti kao testni skup za izgrađene jezične modele. Jezični model je potrebno izgraditi bez ikakvog znanja o testnom skupu jer bilo kakvo znanje o skupnom testu kao posljedicu ima neprirodno nisku vrijednost perpleksije. Jezični model je to bolji što je iznos perpleksije manji, a ona će biti manja od uobičajene ukoliko se testni skup preklapa s trening skupom. Korištenje perpleksije pri vrednovanju modela se naziva još i intrinzičan način vrednovanja, a prednost mu je ta što nije vremenski zahtjevan kao što je to vanjski (ekstrinzični) način vrednovanja modela.

5.2 Stopa pogreške riječi

Druga mogućnost vrednovanja modela koju vrijedi spomenuti je stopa pogreške riječi (engl. *word error rate*, *WER*). WER se češće koristi u sustavima za raspoznavanje govora. Osnovni problem koji se javlja pri mjerenju performansi modela je taj što rečenica koja je rekonstruirana uporabom modela može imati različitu duljinu od referentne rečenice. WER je tako mjera za usporedbu različitih modela i mjera poboljšanja modela. Pritom WER ne otkriva detalje kao što je uzrok pogreške za što se treba primjeniti neku od drugih tehnika. Problem različite duljine rečenica se rješava na način da se referentna i rekonstruirana rečenica poravnaju uspoređujući ih po riječima. Ispitivanje problema se vrši preko zakona potencija (engl. *power law*) koji donosi korelaciju između perpleksije i stope pogreške riječi. WER se može izračunati kao:

$$WER = \frac{S + D + I}{N} = \frac{S + D + I}{S + D + C} \quad (5.9)$$

gdje je S broj zamjena, D broj brisanja, I broj umetanja, C broj točnih riječi, a N broj riječi u referentnoj rečenici. Problem formule 5.9 je taj što ne uzima u obzir prirodu nastale pogreške jer se neke mogu ispraviti lakše od drugih te također ne može razlikovati zamjenu riječi od kombinacije brisanja i potom umetanja riječi. Predložena je stoga formula gdje je broj brisanja i broj umetanja pomnožen s faktorom 0.5 dok broj zamjena ostaje nepromijenjen:

$$WER = \frac{S + 0.5D + 0.5I}{N} \quad (5.10)$$

6. Aplikacija za stvaranje jezičnog modela

Kao rezultat ovog rada razvijena je aplikacija za automatizirano stvaranje statističkog jezičnog modela obradom jezičnog korpusa. Implementirani su algoritmi objašnjeni u prijašnjim poglavljima te je dana ocjena njihovog rada i performansi.

6.1 Zahtjevi i specifičnosti

Aplikacija pruža dva osnovna skupa funkcionalnosti:

- Automatsko stvaranje statističkog modela zasnovanog na N-gramima na temelju ulaznih datoteka s jezičnim korpusom odabirom željene tehnike zaglađivanja
- Testiranje postojećih jezičnih modela nad ulaznim testnim skupom uz ispis odgovarajućih evaluacijskih parametara

Implementirane su sljedeće tehnike zaglađivanja:

- aditivno zaglađivanje
- Good-Turing zaglađivanje
- Witten-Bell zaglađivanje s odstupanjem
- apsolutno snižavanje
- originalni Kneser-Ney algoritam zaglađivanja
- modificirani Kneser-Ney algoritam zaglađivanja

Good Turing zaglađivanje se pritom ne koristi kao samostalan algoritam zaglađivanja već u kombinaciji s drugim tehnikama. Nakon izgrađenog statističkog jezičnog modela nad trening skupom aplikacija koristi Viterbijev dekođer za određivanje najboljeg puta odnosno niza stanja skrivenog Markovljevog modela u testnom skupu niza ulaznih obzervacija. Postupak je opisan u poglavlju 4. Kao mjere usporedbe modela koriste se perpleksija i WER opisani u poglavlju 5.

6.2 Jezični korpus

Jezični korpus koji se koristio tijekom izrade ovog rada je pojednostavljena verzija korpusa SETimes.HR. SETimes.HR je lingvistički označen korpus hrvatskog jezika te je prvi i do ovog trenutka najbolji jezični izvor hrvatskog jezika distribuiran pod CC BY-SA 3.0 licencom. Sastoji se od svakodnevno korištenog hrvatskog jezika iz

novinskih članaka iz različitih web izvora. Format SETimes.HR korpusa kojeg koristi aplikacija je sljedeći:

| | |
|-------------------------|----------------|
| <i>riječ1</i> | <i>oznaka1</i> |
| <i>riječ2</i> | <i>oznaka2</i> |
| <i>riječ3</i> | <i>oznaka3</i> |
| <i>riječ4</i> | <i>oznaka4</i> |
| <i>* prazni redak *</i> | |
| <i>riječ5</i> | <i>oznaka5</i> |
| <i>riječ6</i> | <i>oznaka5</i> |
| ... | |
| <i>* prazni redak *</i> | |
| <i>/kraj</i> | |

svaka riječ rečenice je zapisana u posebnom retku u kojem se zajedno s riječi nalazi i oznaka koja označava vrstu riječi odvojena prazninom. Prazan redak odvaja trenutnu i sljedeću rečenicu dok se na samom kraju korpusa nalazi oznaka /kraj. Testni skup se zadaje bez praznog retka i oznake /kraj na kraju datoteke. Korištena verzija SETimes.HR korpusa sastoji se od:

Tablica 9. Statistički podaci SETimes.HR korpusa

| rečenice | oznake riječi (tokeni) | vrste riječi (tipovi) | tipovi oznaka vrste riječi |
|----------|------------------------|-----------------------|----------------------------|
| 7994 | 178981 | 27587 | 12 |

Za testiranje modela korišten je odvojen i posebno pripremljen testni skup istog formata kao i SETimes.HR korpus, *set.hr.test* koji sadrži rečenice hrvatskih i srpskih novinskih agencija.

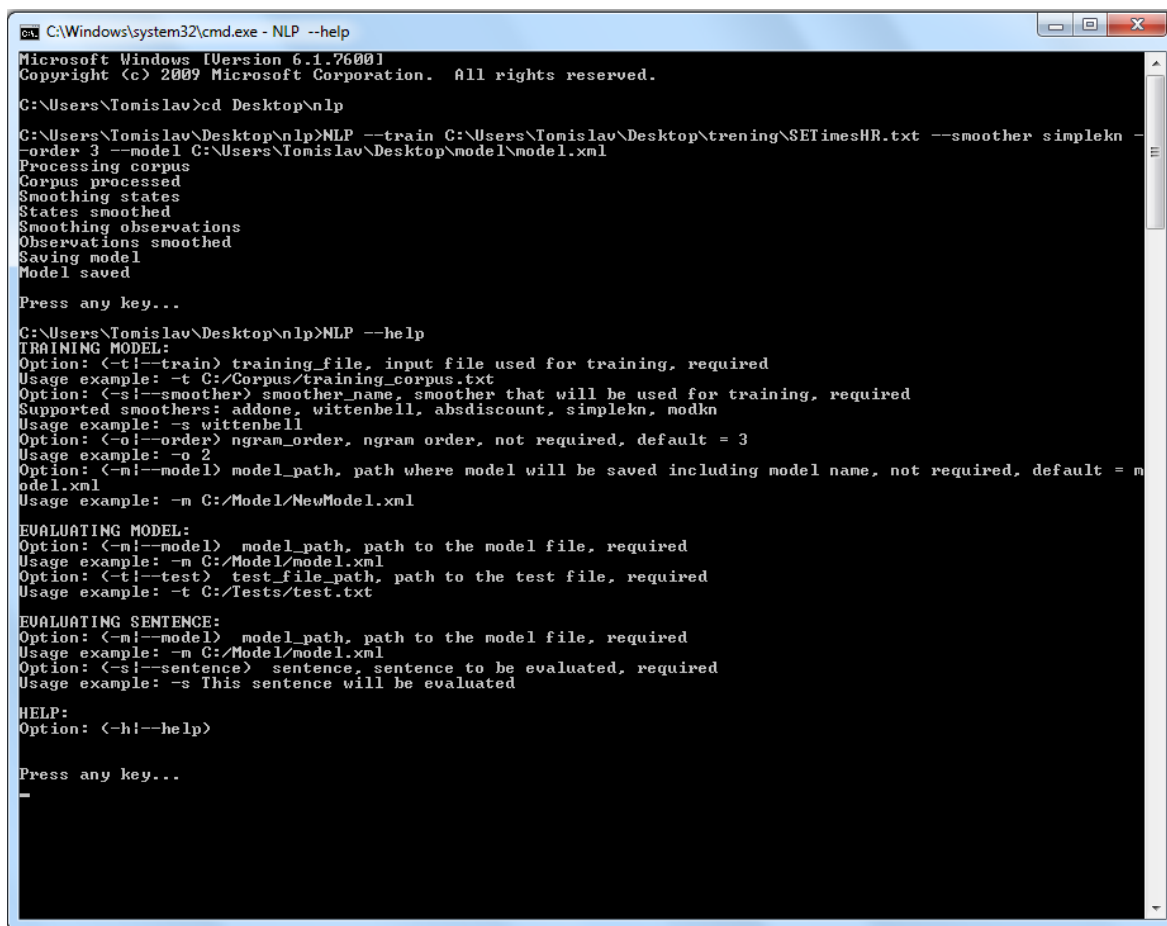
Tablica 10. Statistički podaci *set.hr.test* testnog skupa

| rečenice | oznake riječi (tokeni) | vrste riječi (tipovi) | tipovi oznaka vrste riječi |
|----------|------------------------|-----------------------|----------------------------|
| 100 | 2258 | 1265 | 11 |

Ukoliko se želi koristiti neki drugi korpus tada on mora biti u prethodno opisanom formatu što također vrijedi i za testne skupove.

6.3 Arhitektura

Aplikacija se pokreće iz komandne linije. Primjer pokretanja prikazan je slikom 6.



```
C:\Windows\system32\cmd.exe - NLP --help
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Tomislav>cd Desktop\nlp

C:\Users\Tomislav\Desktop\nlp>NLP --train C:\Users\Tomislav\Desktop\trening\SETimesHR.txt --smoother simplekn --
--order 3 --model C:\Users\Tomislav\Desktop\model\model.xml
Processing corpus
Corpus processed
Smoothing states
States smoothed
Smoothing observations
Observations smoothed
Saving model
Model saved

Press any key...

C:\Users\Tomislav\Desktop\nlp>NLP --help
TRAINING MODEL:
Option: (-t|--train) training_file, input file used for training, required
Usage example: -t C:/Corpus/training_corpus.txt
Option: (-s|--smoother) smoother_name, smoother that will be used for training, required
Supported smoothers: addone, wittenbell, absdiscount, simplekn, modkn
Usage example: -s wittenbell
Option: (-o|--order) ngram_order, ngram order, not required, default = 3
Usage example: -o 2
Option: (-m|--model) model_path, path where model will be saved including model name, not required, default = m
odel.xml
Usage example: -m C:/Model/NeuModel.xml

EVALUATING MODEL:
Option: (-m|--model) model_path, path to the model file, required
Usage example: -m C:/Model/model.xml
Option: (-t|--test) test_file_path, path to the test file, required
Usage example: -t C:/Tests/test.txt

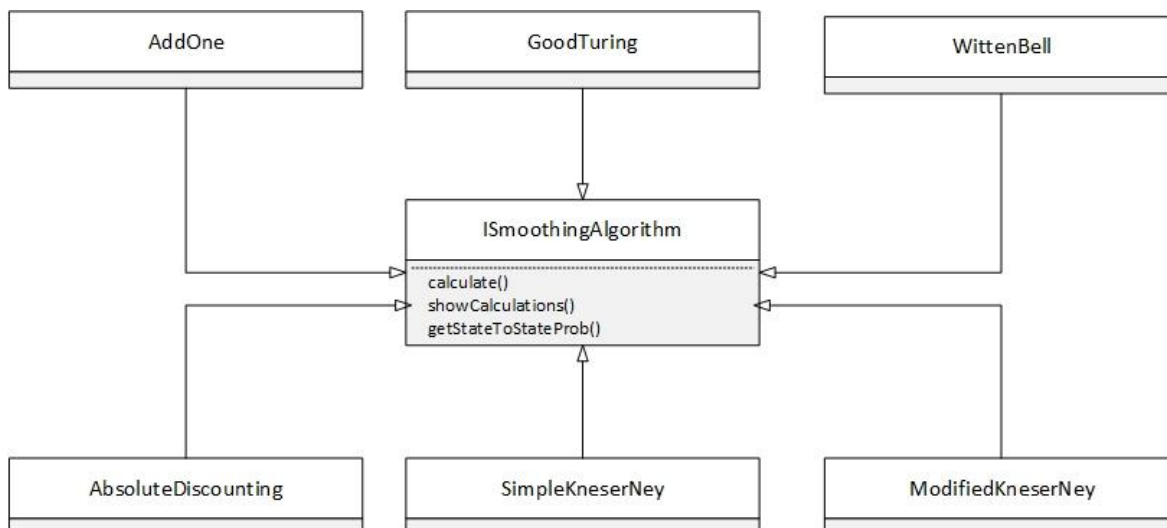
EVALUATING SENTENCE:
Option: (-m|--model) model_path, path to the model file, required
Usage example: -m C:/Model/model.xml
Option: (-s|--sentence) sentence, sentence to be evaluated, required
Usage example: -s This sentence will be evaluated

HELP:
Option: (-h|--help)

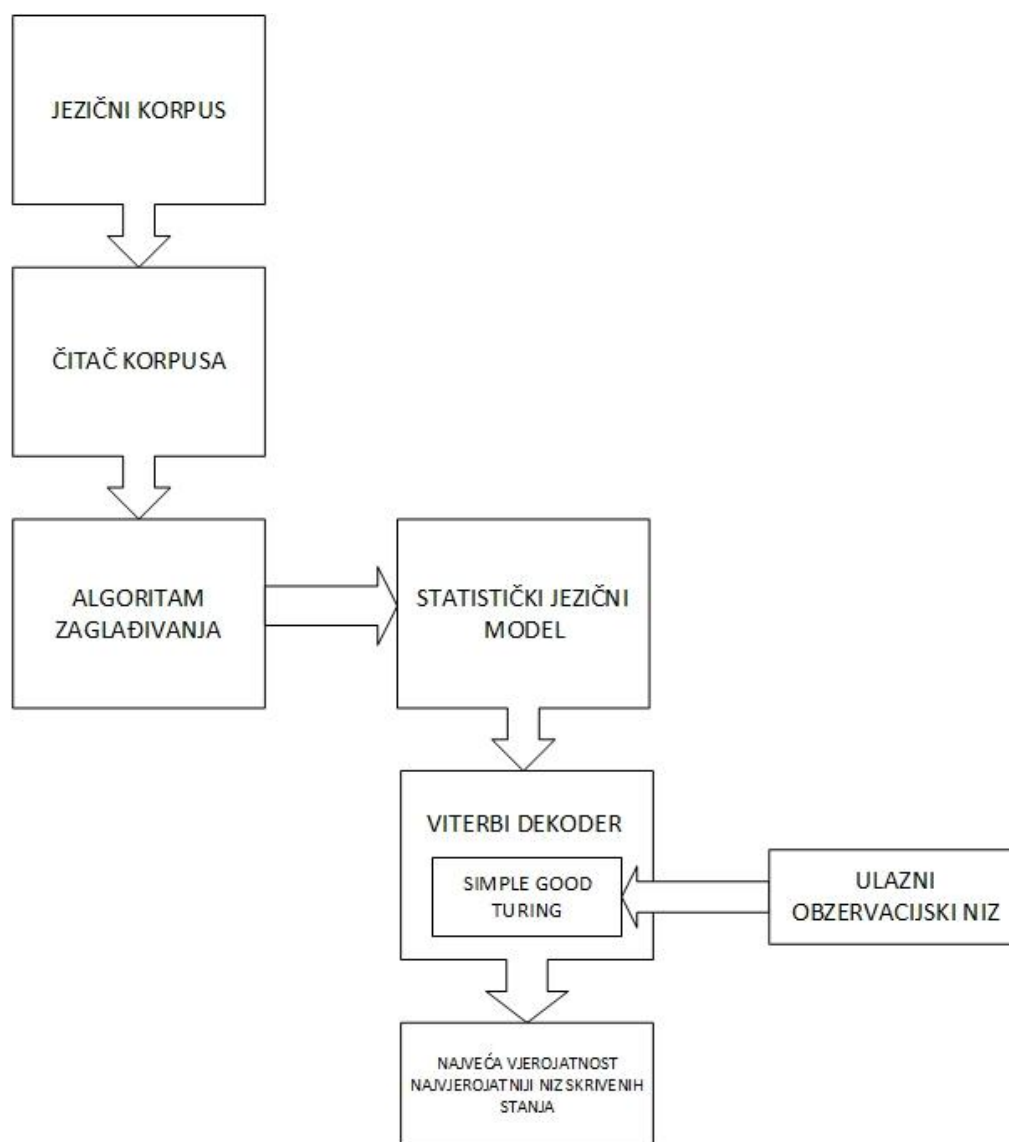
Press any key...
```

Slika 6. Pokretanje aplikacije iz komandne linije

Pomoć pri korištenju kao i prikaz svih mogućnosti aplikacije s pojašnjenjima može se dobiti pomoću parametra `--help`. Aplikacija se sastoji od nekoliko komponenata od kojih su važnije sljedeće: za učitavanje jezičnog korpusa i pohranu podataka u odgovarajuće strukture korsiti se čitač korpusa (klasa *CorpusReader*). Čitač korpusa je u mogućnosti vršiti učitavanje N-grama proizvoljnog reda no razvijeni algoritmi su definirani na način da mogu obraditi N-grame do uključivo trigramu stoga će redovi učitavanja biti $N = \{1,2,3\}$. Pomoćna klasa *Mapper* vrši preslikavanje obzervacija i stanja u cjelobrojne vrijednosti radi lakše obrade. Za zaglađivanje na raspolaganju je šest algoritama zaglađivanja koji su spomenuti u 6.1. Za određivanje najvjerojatnijeg puta kroz skriveni Markovljev model koristi se klasa *Viterbi*. Način na koji su definirani algoritmi prikazan je na slici 7.



Slika 7. Organizacija algoritama zaglađivanja



Slika 8. Osnovni način rada i arhitektura aplikacije

Na slici 8. prikazan je način rada aplikacije. Čitač korpusa čita jezični korpus s obzirom na odabranu razinu N-grama, $1 \leq N \leq 3$ te tako obrađene podatke pohranjuje u odgovarajuće podatkovne strukture. Podatkovne strukture koristi odabrani algoritam zaglađivanja čijim se izvršavanjem stvara statistički jezični model. Jezični model koristi Viterbijev dekođer koji određuje najvjerojatniji niz stanja skrivenog Markovljevog modela za ulazni obzervacijski niz. Ulazni obzervacijski niz je testni skup. Testni skup se sastoji od niza riječi koje čine rečenice u formatu u kojem je opisan jezični korpus gdje je oznaka ujedno i jedno od stanja skrivenog Markovljevog modela. Testni skup sadrži oznake stanja kako bi se na osnovu izlaznog niza stanja mogao izračunati WER. Za svako stanje skrivenog Markovljevog modela se računa vjerojatnost pojave obzervacije u tom stanju što je potrebno za Viterbijev algoritam. Frekvencije pojava obzervacija u stanjima Markovljevog modela su pohranjene u odgovarajućoj strukturi podataka koju je prethodno pripremio čitač korpusa. Te frekvencije se zaglađuju algoritmom zaglađivanja *Simple Good Turing* na način koji je opisan u 2.2 poglavlja o algoritmima zaglađivanja. Viterbijev dekođer čiji je način rada objašnjen u poglavlju 4. će potom odrediti najveću moguću vjerojatnost i najbolji put kroz model tj. najvjerojatniji niz skrivenih stanja modela. Na osnovu tih vrijednosti za statistički jezični model računaju se vrijednosti perpleksije i WER.

6.4 Rezultati

U testnom skupu zaglađivani su tipovi oznaka riječi na temelju čega je izgrađen statistički jezični model. Provedbom Viterbijeva algoritma and ulaznim testnim skupom dobiveni su sljedeći rezultati:

Tablica 11. Testni skup *set.hr.test*

| | WittenBell | Apsolutno snižavanje | Kneser-Ney | Mod Kneser-Ney |
|--------------------|------------|-------------------------|------------|----------------|
| Perpleksija | 6.1685 | 5.3384 | 5.3479 | 5.3480 |
| WER | 8.7688% | 8.8131% | 8.7688% | 8.7688% |

Također izgrađeni su jezični modeli u kojima su zaglađivani N-grami vrsta riječi, koje za Viterbijev algoritam predstavljaju ulazni obzervacijski niz. Testiranjem modela dobiveni su sljedeći rezultati:

Za bigrame:

Tablica 12. Vrijednosti perpleksija za modele bigrama

| Algoritam zaglađivanja | Perpleksija |
|-------------------------|-------------|
| Witten-Bell | 511.3186 |
| Apsolutno snižavanje | 402.7251 |
| Kneser-Ney | 374.2508 |
| Modificirani Kneser-Ney | 380.1155 |

Za trigrame:

Tablica 13. Vrijednosti perpleksija za modele trigrama

| Algoritam zaglađivanja | Perpleksija |
|-------------------------|-------------|
| Witten-Bell | 381.3808 |
| Apsolutno snižavanje | 290.9503 |
| Kneser-Ney | 265.6475 |
| Modificirani Kneser-Ney | 267.2888 |

U tablicama niža vrijednost perpleksije predstavlja bolji rezultat. To se može također definirati kao mjera koliko je model iznenađen testnim skupom. Uspoređujući rezultate testiranja modela vidljivo je da algoritmi Kneser-Ney i modificirani Kneser-Ney daju najbolje rezultate nakon čega slijede apsolutno snižavanje te Witten-Bell algoritam s odstupanjem. Zaključak je također da modeli N-grama viših razina daju bolje rezultate od N-grama nižih razina što se može vidjeti na ovom primjeru bigrama i trigrama.

Zaključak

Jezični modeli su poveznica mnogi znanstvenih disciplina koje se bave obradom prirodnog jezika. Kako bi se opisala sva kompleksnost jezika te kako bi se njegovo ponašanje moglo modelirati potrebno je imati širok spektar znanja o jeziku. Jedan od osnovnih problema koji se javlja je predviđanje sljedeće riječi odnosno vjerojatnosti pojave određenog niza riječi. Za rješavanje tog problema koriste se statistički jezični modeli N-grama. N-grami se grade iz jezičnog korpusa. Problem koji se pritom uvijek javlja je podatkovna rijetkost zbog konačnosti korpusa. Bez obzira koliko velik bio jezični korpus on nije u stanju obuhvatiti sve jezične konstrukte jezika. To za posljedicu ima da se veliki broj N-grama koji se mogu pojaviti u jeziku ne nalazi u jezičnom korpusu te je njihova vjerojatnost 0. Kako bi obuhvatili i takve N-grame koristimo se algoritmima zaglađivanja. Svaki od tih algoritama ima određenu primjenu te se vrlo često ne koristi sam u postupku zaglađivanja. Algoritmi zaglađivanja se također susreću s brojnim problemima ovisno o načinu na koji su definirani pa stoga ponekad nije moguće primjenom samo jednog algoritma dobiti zadovoljavajući rezultat. Razina složenosti algoritama i način njihova korištenja ovisit će o specifičnom području primjene. Također velika većina algoritama uključuje brojne parametre čije vrijednosti nije moguće pretpostaviti jednako za sve moguće primjene već su oni podložni podešavanju na konkretnom zadatku. Statistički jezični model također vrlo ovisi o jezičnom korpusu te je njegovom odabiru potrebno posvetiti posebnu pažnju jer će o tome ovisiti buduće performanse modela. Od algoritama koji su korišteni u ovom radu rezultatima su se posebno istaknuli izvorni i modificirani Kneser-Ney algoritam. Isto tako jezični model izgrađen nad trigramima je polučio bolje rezultate od onoga izgrađenog nad bigramima.

Literatura

- [1] Jurafsky D., Martin J. H. (1999.), An Introduction to Natural Language, Computational Linguistics and Speech Recognition, Prentice Hall
- [2] Chen S. F., Goodman J. (1998.), An Empirical Study of Smoothing Techniques for Language Modeling, Center for Research in Computing Technology Harvard University Cambridge, Massachusetts
- [3] Jelinek F., Mercer R. (1985.), Probability distribution estimation from sparse data, IBM Technical Disclosure Bulletin
- [4] Nádas A. (1985.), On Turing's formula for word probabilities, IEE Transactions on Acoustics, Speech and Signal Processing
- [5] Heafield K., Pouzyrevsky I., Clark J. H., Koehn P., Scalable Modified Kneser-Ney Language Model Estimation, University of Edinburgh, Carnegie Mellon University, Yandex
- [6] Katz S. M. (1987.), Estimation of probabilities from sparse data for the language model component of a speech recognizer, In IEEE Transactions on ASSP
- [7] Andrés-Ferrer J., Ney H., Extensions of absolute discounting (Kneser-Ney method), Universidad Politécnica de Valencia, Valencia, Spain, Lehrstuhl fuer Informatik 6 RWTH Aachen University, Aachen, Germany
- [8] Gale W. A., Good-Turing Smoothing Without Tears, AT&T Bell Laboratories
- [9] James F. (2000.), Modified Kneser-Ney Smoothing of n-gram Models, Research Institute for Advanced Computer Science, Technical Report 00.07
- [10] Good I. J. (1953.), The population frequencies of species and the estimation of population parameters, Biometrika
- [11] Jelinek F., Mercer R. (1980.), Interpolated estimation of markov source parameters from sparse data. In Proceedings of the Workshop on Pattern Recognition in Practice
- [12] Ghahramani Z. (2001.), An Introduction to Hidden Markov Models and Bayesian Networks, Gatsby Computational Neuroscience Unit, University College London

- [13] Li J., Hidden Markov Model notes, Department of Statistics The Pennsylvania State University
- [14] Stamp M., (2012.), A Revealing Introduction to Hidden Markov Models, Department of Computer Science, San Jose State University
- [15] Chatterjee S., Russell S, A temporally abstracted Viterbi algorithm, Computer Science Division University of California, Berkeley
- [16] Agić Ž. Ljubešić N., The SETIMES.HR Linguistically Annotated Corpus of Croatian, Linguistics Department, University of Potsdam, Germany, Department of Information and Communication Sciences, Faculty of Humanities and Social Sciences, University of Zagreb, Croatia
- [17] Agić, Ž.; Ljubešić, N.; Merkler, D. (2013.) Lemmatization and Morphosyntactic Tagging of Croatian and Serbian. In *Proceedings of the 4th Biennial International Workshop on Balto-Slavic Natural Language Processing (BSNLP 2013)*. Sofia, Bulgaria, Association for Computational Linguistics, 2013, pp. 48–57.
- [18] Agić, Ž.; Merkler, D. (2013.) Three Syntactic Formalisms for Data-Driven Dependency Parsing of Croatian. In *Text, Speech and Dialogue. Lecture Notes in Computer Science*. Berlin, Heidelberg, Springer, 2013, pp. 560–567.
- [19] Ljubešić, N.; Stupar, M.; Jurić, T.; Agić, Ž. (2013.) Combining Available Datasets for Building Named Entity Recognition Models of Croatian and Slovene. *Slovenščina 2.0: empirical, applied and interdisciplinary research*, in press.

Sažetak

APLIKACIJA ZA AUTOMATIZIRANO STVARANJE STATISTIČKOG JEZIČNOG MODELA OBRADOM JEZIČNOG KORPUSA

Ovaj diplomski rad daje uvid u problematiku predviđanja niza riječi prilikom obrade prirodnog jezika. Statistički jezični model baziran je na N-gramima i stvara se obradom jezičnog korpusa brojanjem riječi i nizova riječi. Jezični model pretpostavlja postojanje Markovljevih svojstava među uzastopnim riječima prirodnog jezika. S obzirom na podatkovnu rijetkost korpusa javlja se problem N-grama vjerojatnosti nula zbog njihovog nepojavljivanja u jezičnom korpusu. Kako bi se taj problem ublažio koriste se tehnike zaglađivanja koje će i takvim N-gramima pridjeliti određenu vjerojatnost na temelju procjene frekvencija postojećih N-grama. U tu svrhu razvijeno je puno algoritama zaglađivanja od kojih su neki opisani u ovom radu. Neki algoritmi se također obično ne koriste samostalno već upareni s nekim drugim algoritmima zaglađivanja. Ovisno o samoj implementaciji algoritmi prilikom zaglađivanja također mogu naići na probleme te svaki od njih ima određene prednosti i mane pa se s toga odabiru ovisno o vrsti problema i području primjene. Kako bi se odredio najvjerojatniji niz riječi na temelju ulaznih podataka i izgrađenog jezičnog modela koristi se Viterbijev algoritam koji će dati odgovor koji je najvjerojatniji niz stanja skrivenog Markovljevog modela. Za usporedbu kvalitete jezičnog modela najčešće se koristi mjera perpleksije. U ovom radu kao najkvalitetniji model pokazao se onaj izgrađen Kneser-Ney algoritmom zaglađivanja.

Ključne riječi: statistički jezični model, N-grami, algoritmi zaglađivanja, skriveni Markovljev model, Viterbijev algoritam

Summary

APPLICATION FOR AUTOMATED STATISTICAL LANGUAGE MODEL CONSTRUCTION BY PROCESSING LANGUAGE CORPUS

This master's thesis gives an insight into the issues of predicting word sequences in natural language processing. Statistical language model is based on n-grams and it is created by language corpus processing (counting the words and word sequences). Language model assumes the existence of Markov properties among the consecutive words in a natural language. Due to the data sparsity of the corpus there is a problem of zero probability n-gram because of their nonappearance in the language corpus. In order to alleviate the problem we use smoothing techniques which will assign a specific probability to those n-grams based on the estimate of frequencies of the existing n-grams. For that purpose many smoothing algorithms have been developed, some of which have been described in this thesis. Some algorithms are not usually used independently but they are paired with some other smoothing algorithms. Depending on the implementation, the algorithms can come across some problems while smoothing and each of them has certain advantages and disadvantages and therefore they are chosen based on the type of the problem and scope of application. In order to determine the most probable word sequence based on the input data and the created language model, we use Viterbi algorithm that will give an answer which is the most probable state sequence of the hidden Markov model. In order to compare the quality of the language model the measure of perplexity is most frequently used. In this thesis the best model was the one that was created by Kneser-Ney smoothing algorithm.

Keywords: statistical language model, n-grams, smoothing algorithms, hidden Markov model, Viterbi algorithm