TakeLab

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

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1526

Ispitivanje vektorskih reprezentacija riječi hrvatskoga jezika

Filip Čulinović

Zagreb, srpanj 2017.

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

Zagreb, 3. ožujka 2017.

Predmet: **Analiza i pretraživanje teksta**

DIPLOMSKI ZADATAK br. 1526

Pristupnik: **Filip Čulinović (0036472908)**
Studij: Računarstvo
Profil: Računarska znanost

Zadatak: **Ispitivanje vektorskih reprezentacija riječi hrvatskoga jezika**

Opis zadatka:

Računalna semantika ima važnu ulogu u sustavima za obradu i razumijevanje prirodnoga jezika. Distribucijski semantički modeli značenje riječi prikazuju kontekstnim vektorima izgrađenima na temelju korpusa. Jedna od boljki takvih vektorskih reprezentacija jest prikazivanje značenja višeznačnih (polisemnih) riječi, koja su kod standardnih modela različita značenja superponirana su u jedan distribucijski vektor, što narušava kvalitetu modela.
U okviru diplomskog rada potrebno je proučiti modele vektorskih reprezentacija riječi, s naglaskom na modele neuronskih reprezentacija i modele prilagođene modeliranju višeznačnosti riječi, poput multiprototipnog modela Huanga i dr. (2012). Razviti učinkovitu računalnu implementaciju nekoliko odabranih modela te ih primijeniti na korpusima tekstova na hrvatskome jeziku. Provesti iscrpno eksperimentalno vrednovanje modela na nekoliko standardnih leksičkosemantičkih zadataka, uključivo zadatcima prepoznavanja semantičke sličnosti, sinonimije i analogije, koristeći raspoložive skupove podataka za hrvatski jezik. Provesti i dodatno vrednovanje modela na odgovarajućim referentnim skupovima na engleskome jeziku. Načiniti detaljnu analizu pogrešaka te statističku obradu rezultata. Radu priložiti izvorni i izvršni kod razvijenog sustava, skupove podataka i programsku dokumentaciju te citirati korištenu literaturu.

Zadatak uručen pristupniku: 10. ožujka 2017.
Rok za predaju rada: 29. lipnja 2017.

Mentor:

Izv. prof. dr. sc. Jan Šnajder

Djelovođa:

Doc. dr. sc. Tomislav Hrkać

Predsjednik odbora za diplomski rad profila:

Prof. dr. sc. Siniša Srbljić

Zahvaljujem svojoj obitelji na svoj podršci koju su mi pružili tokom dosadašnjeg obrazovanja

SADRŽAJ

1. Uvod

U ovom radu opisan je pristup problemu stvaranja vektorskih reprezentacija riječi hrvatskoga jezika pomoću modela strojnog učenja. Ljudi tijekom cijelog svog života ponekad i nesvjesno uče značenja riječi s kojima se susreću. S obzirom da je računalima riječ prirodnog jezika samo niz znakova, potrebno je pronaći neki način reprezentacije riječi koji nosi značenje računalu. Obrada prirodnog jezika (engl. *natural language processing, NLP*) grana je računalne znanosti koja se bavi obradom i razumijevanjem teksta. Razumijevanje teksta kreće od razumijevanja jedne riječi. Riječ se analizira kako bi se odredila njena morfološka struktura i priroda riječi poput značenja te korištenih frazema. Sljedeća viša jezična struktura je rečenica koja je po svojoj prirodi niz riječi. Njoj možemo odrediti poredak riječi, analizirati gramatiku te sintaksno stablo. Pomoću tih informacija te poznatog značenja riječi dolazimo do semantike rečenice, a sve veće tekstne oblike možemo kasnije gledati na jednak način kao skupove rečenica te njihove strukture i značenja. Liddy (1998) i Feldman (1999) predlažu da mogućnost razumijevanja prirodnih jezika dolazi preko razlikovanja sljedećih sedam međuzavisnih razina koje ljudi koriste za ekstrakciju značenja iz teksta ili govora: (1) fonetička ili fonološka razina, (2) morfološka razina, (3) leksička razina, (4) sintaksna razina, (5) semantička razina, (6) komunikacijska razina i (7) pragmatična razina. Kao što smo u prethodnom primjeru vidjeli, za svaku od ovih razina postoje sustavi čijom se izradom bavi područje analize prirodnog jezika. U ovom radu poseban fokus je na semantičkoj razini te kreiraju računalnih reprezentacija riječi [10, 7]. S obzirom da su riječi osnovni nositelji značenja, potrebno je računalu prilagoditi strukturu te način reprezentiranja riječi unutar računalnih sustava. Ukoliko bismo usporedili ljudsku percepciju riječi te računalnu, možemo lako vidjeti da su riječi niz znakova za koje mi kroz svoje iskustvo znamo značenje. To iskustvo je upravo poznavanje prirode riječi koje računalo nema, pa vidimo da računalnom sustavu niz znakova nije optimalan način za prijenos istih informacija kao među ljudima. Najjednostavniji mogući pristup je kreirati *1-od-N* (engl. *1-of-N, one-hot*) vektor duljine broja svih riječi u vokabularu sa samo jednim poljem na kojemu se nalazi broj 1 koji je unikatan svakoj poznatoj riječi. Uzmimo u obzir

da se naš vokabular sastoji od samo pet riječi: *kralj*, *kraljica*, *muškarac*, *žena* i *dijete*. Prethodnom metodom riječima bismo dodijelili vektore tako da redom svakoj riječi dajemo vektor s jedinicom na indeksu njenog pojavljivanja. Time bismo riječ kralj kodirali pomoću niza 10000, riječ kraljica sa nizom 01000, itd. Iako smo ovim načinom uspješno unikatno predstavili sve riječi, iz njihovog međusobnog odnosa kao dva vektora ne može se prenijeti nikakva dodatna značajna informacija. Također moramo uzeti u obzir da veličina vokabulara nekih prirodnih jezika lako premašuje sto tisuća riječi, što nas brzo dovodi do velike memorijske neefikasnosti te potrebe za ponovnim kreiranjem svih vektora riječi prilikom upoznavanja nove.

Iz tog razloga postoji područje istraživanja koje se naziva distribucijska semantika čiji je zadatak kvantizacija te opisivanje semantičkih veza među lingvističkim objektima pomoću njihove distribucije u vrlo velikim skupovima teksta. Naziv distribucijska semantika dolazi od načina na koji se reprezentira značenje riječi. Smanjenjem dimenzija reprezentacije *1-od-N* s veličine vokabulara na neki proizvoljno manji broj gubimo lak način razlikovanja riječi, ali više ne koristimo samo jednu dimenziju za reprezentaciju riječi već distribuiramo značenje na sve dimenzije vektora te time možemo prenijeti značajno više informacija. U ovom radu ćemo iskoristiti metode strojnog učenja kako bismo naučili vektore riječi hrvatskog jezika. Ispitat ćemo predikcijske modele – *kontinuirana vreća riječi*, *skip-gram*, modele *FastText* te modele bazirane na matrici supojavljivanja riječi – *SN* i *GloVe*. U prirodnim jezicima postoje i polisemične riječi, što znači da imaju više mogućih značenja. Postoji više pristupa za reprezentaciju višeznačnih riječi od kojih je osnovna podjela na one koji koriste više vektora (za svako značenje riječi po jedan) do onih koji koriste samo jedan vektor i neke atribute [3, 4]. U ovom radu ćemo ispitati algoritam *k-SVD* koji koristi samo jedan vektor za reprezentaciju rješenja.

Rad se sastoji od dva glavna zadatka. Prvi zadatak je korištenje više modela za izradu vektora riječi hrvatskog jezika te njihova evaluacija. Modele ćemo ispitati na zadacima odabira sinonima te analogijskih pitanja. Na svim modelima ispitat ćemo različite korpuse za učenje te parametre modela. Drugi zadatak je primjena metode navedene u [4] na prethodno naučene vektore riječi kako bismo reprezentirali polisemične riječi. Ove polisemične reprezentacije riječi ispitat ćemo na zadatku odabira značenja riječi.

2. Modeli

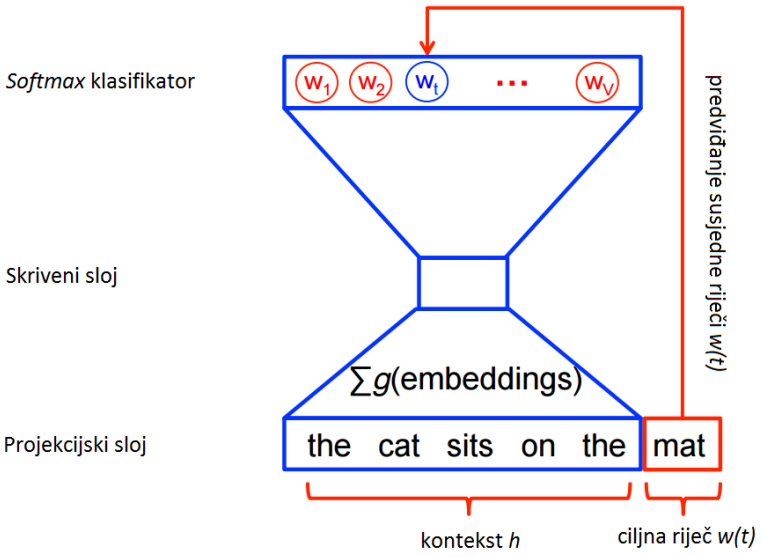
Modeli za izračun vektorskih reprezentacija riječi dijele se na dvije osnovne skupine. Prva skupina su prediktivni modeli poput modela *CBoW* i *Skip-gram*, koji uče vektorske reprezentacije pomoću zadatka predviđanja riječi. Druga skupina modela uče vektorske reprezentacije koristeći matricu supojavljivanja (engl. *co-occurrence matrix*), koja sadrži zapise o supojavljivanju dvije riječi unutar nekog kliznog prozora (engl. *window*) u cijelom tekstu. U nastavku ćemo opisati modele za dobivanje vektorskih reprezentacija riječi korištenih u ovom radu.

2.1. Kontinuirana vreća riječi

Kontinuirana vreća riječi (engl. *Continuous Bag-of-Words, CBoW*) je jednostavan model za izračun vektorskih reprezentacija riječi proizvoljne veličine. Naziv vreća riječi (engl. *bag-of-words*) dolazi od načina na koji se riječi koriste u modelu. Za ciljnu riječ uzimamo prozor od proizvoljnog broja riječi oko nje, ali nam njihov redoslijed nije bitan. S obzirom da imamo više riječi za koje u ovom trenutku više ne znamo poredak, možemo to simbolizirati bacanjem riječi u vreću. Model je kontinuiran jer koristi kontinuiranu i distribuiranu reprezentaciju konteksta. Vektorske reprezentacije riječi model uči pomoću predviđanja trenutačne riječi prema dostupnom kontekstu [14].

Uzmimo za primjer rečenicu *"The cat sits on the mat"*. Ako bismo za potrebe treniranja postavili parametar veličine prozora na 5 te koristeći asimetrični prozor koji se prostire samo prije riječi, dobili bismo kontekst oko ciljne riječi *mat* kao što je prikazano na slici 2.1 gdje *embeddings* predstavlja vektorske reprezentacije riječi.

Model je zapravo potpuno povezana neuronska mreža s jednim skrivenim slojem. Svi neuroni u skrivenom sloju su linearni neuroni. Ulazni sloj sastoji se od toliko neurona koliko riječi se nalazi u vokabularu za učenje, V. Veličina skrivenog sloja upravo je jednaka proizvoljnoj dimenziji vektora riječi N. Iz tog razloga težine između ulaznog i skrivenog sloja možemo prikazati pomoću matrice W dimenzija $V \times N$ gdje je svaki red N-dimenzionalna vektorska reprezentacija v_w pripadajuće riječi w u



Slika 2.1: Pojednostavljeni prikaz modela kontinuirane vreće riječi [8]

ulaznom sloju. Ulazne vrijednosti u mrežu su *1-od-N* kodovi riječi iz prozora veličine C oko riječi čiji vektor učimo. Vrijednost skrivenog sloja h postaje:

$$h = \frac{1}{C} W \cdot (\sum_{i=1}^{C} x_i)$$

gdje je x_i *1-od-N* vektor dimenzije veličine vokabulara, a h je vektor reprezentacije konteksta čija je vrijednost prosjek vektora riječi konteksta. Vektori riječi koje učimo upravo su elementi matrice W koje odabiremo pomoću vektora x_i. Matricu težina prema izlaznom sloju možemo opisati matricom W' čije su dimenzije $N \times V$. Koristeći te težine možemo dobiti vrijednost za svaku riječ u vokabularu:

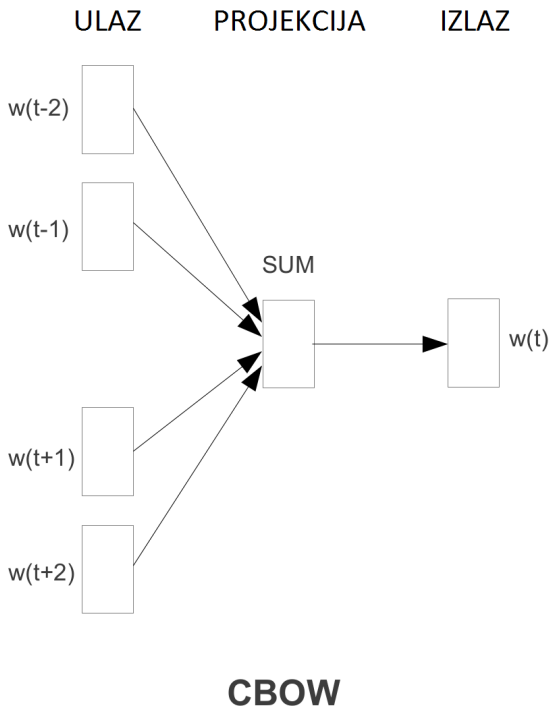
$$u_j = v'_{w_j} \cdot h$$

gdje je v'_{w_j} j-ti stupac matrice W'. Dobivena vrijednost u_j mjera je podudaranja između konteksta i slijedeće riječi te je dobivena skalarnim produktom između predviđenih reprezentacija v'_{w_j} i reprezentacije konteksta h. Sada koristeći *softmax* klasifikacijski model dobivamo aposteriornu distribuciju riječi:

$$y_j = p(w_{y_j}|w_1, ..., w_C) = \frac{exp(u_j)}{\sum_{j'=1}^{V} exp(u_{j'})}$$

gdje je y_j izlazna vrijednost j-te jedinice izlaznog sloja. Koristeći vrijednosti dobivene u izlaznom sloju te poznatu ciljnu riječ, model uči vektorske reprezentacije riječi unutar matrice W kao obične parametre neuronske mreže. Cjelokupni prikaz modela iz

originalnog rada sa slike 2.2 prikazuje nam način rada mreže sa simetričnim prozorom veličine pet, gdje $w(t)$ označava riječ koju predviđamo, a na ulazu se nalaze težine koje odgovaraju dvije prethodne i dvije slijedeće riječi [14].



Slika 2.2: Prikaz modela *CBoW* sa simetričnim prozorom veličine 5 [14]

Uzmimo za primjer vokabular koji se sastoji od pet riječi, a vektori i veličina prozora jednaka je tri. Ciljna riječ je druga riječ u vokabularu s kodom $y = [0, 0, 1, 0, 0]$ te dvije kontekstne riječi $x_1 = [0, 1, 0, 0, 0]$ te $x_2 = [0, 0, 0, 1, 0]$. Uz matrice W i W'

$$W = \begin{bmatrix} 7 & 7 & 1 \\ 3 & 7 & 6 \\ 0 & 4 & 2 \\ 3 & 3 & 2 \\ 3 & 3 & 0 \end{bmatrix}, W' = \begin{bmatrix} 3 & 5 & 2 & 7 & 4 \\ 3 & 2 & 0 & 0 & 5 \\ 3 & 9 & 8 & 1 & 7 \end{bmatrix} \tag{2.1}$$

za skriveni sloj h dobivamo vrijednost:

$$
\begin{aligned}
h &= \frac{1}{2} W \sum_{i=1}^{2} x_i \\
&= \frac{1}{2} W [0, 1, 0, 1, 0] \\
&= [6, 10, 8]
\end{aligned}
\tag{2.2}
$$

Distribucija vjerojatnosti na izlazu zatim postaje:

$$
\begin{aligned}
y' &= softmax(W'h) \\
&= softmax([72, 122, 76, 50, 130]) \\
&\approx [0, 0.003, 0, 0, 0.9997]
\end{aligned}
\tag{2.3}
$$

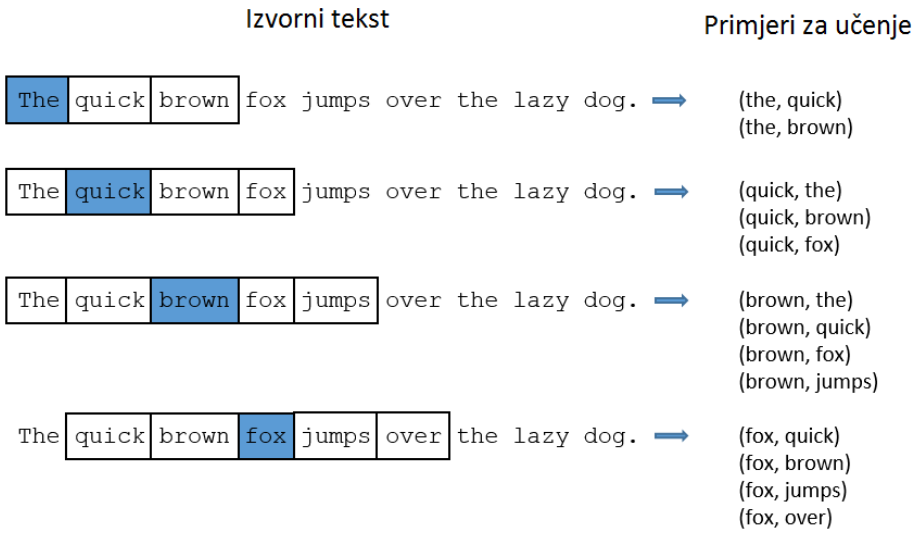
S obzirom da je naše točno rješenje vrijednost y, pomoću metode gradijentnog spusta nad funkcijom gubitka između dobivenog rezultata y' i y optimiziraju se matrice W i W' kako bi se smanjila greška.

2.2. Model *skip-gram*

Model *skip-gram* je u svojoj osnovnoj ideji zrcalno suprotan modelu kontinuirane vreće riječi. Kao i model kontinuirane vreće riječi, on uzima kontekst oko ciljane riječi, ali umjesto da pomoću konteksta predviđa riječ, on pomoću riječi predviđa kontekst. Tako za rečenicu *"The quick brown fox jumps over the lazy dog."* možemo vidjeti kreiranje ulaznih primjera na slici 2.3 pomoću simetričnog prozora veličine pet. Ciljna riječ označena je plavom bojom dok su riječi konteksta uokvirene. [14]

Kako bismo trenirali neuronsku mrežu, na njen ulaz ne možemo dovesti riječi već ih kao i u prethodnom modelu kodiramo *1-od-N* kodovima. Izlaz neuronske mreže je također vektor veličine vokabulara s vjerojatnostima da su te riječi upravo riječi iz konteksta ciljane riječi. Mreža se, kao i u slučaju modela kontinuirane vreće, riječi sastoji od samo jednog skrivenog sloja bez aktivacijske funkcije te završava *softmax* funkcijom nad izlaznim slojem. Mrežu možemo na jednak način podijeliti na matrice težina W i W'. Upravo će matrica W postati naša pregledna (engl. *look-up*) tablica za vektore riječi [14].

S obzirom da su ulazne vrijednosti strogo definirane kao *1-od-N* vektori, omogućeno nam je tretirati matricu W kao poglednu tablicu. Kada bismo imali vokabular veličine 10000 te dimenziju vektora 300, množenjem *1-od-N* vektora veličine 1×10000



Slika 2.3: Kreiranje ulaznih primjera za skip-gram model [13]

$$[0 \quad 0 \quad 0 \quad 1 \quad 0] \times \begin{bmatrix} 17 & 24 & 1 \\ 23 & 5 & 7 \\ 4 & 6 & 13 \\ 10 & 12 & 19 \\ 11 & 18 & 25 \end{bmatrix} = [10 \quad 12 \quad 19]$$

Slika 2.4: Odabir vektora riječi [13]

matricom W dimenzija 10000×300 efektivno smo samo izdvojili redak matrice koji odgovara indeksu na kojem se nalazi broj jedan u ulaznom vektoru. Odabir vektora pomoću *1-od-N* vektora vidljivo je na slici 2.4.

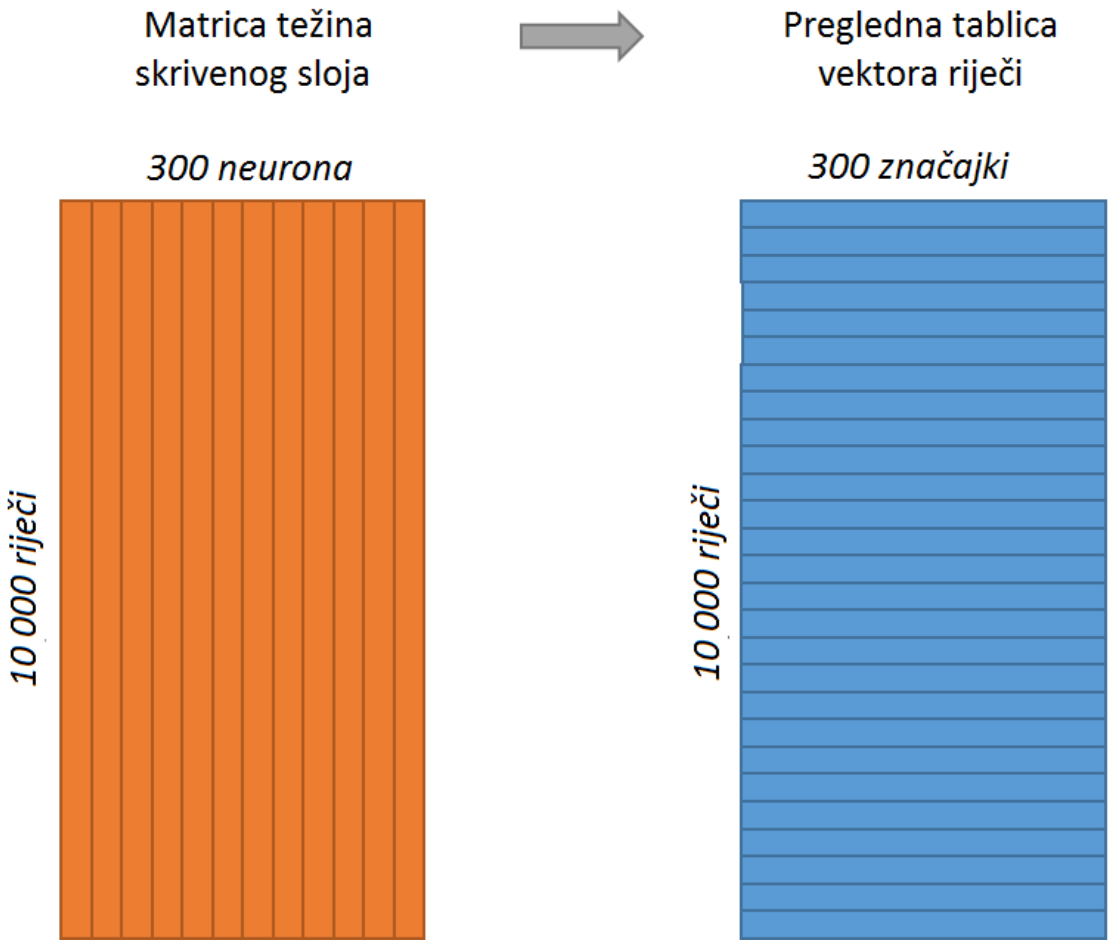
Iz tog razloga možemo definirati funkciju skrivenog sloja h kao:

$$h = x^T W = W_{(k,.)} := v_w$$

gdje je x ulazni *1-od-N* vektor ciljne riječi, a W matrica težina skrivenog sloja čime smo dobili upravo v_w, vektor riječi w kodirane ulazom x. Izlazne vrijednosti dobivaju se formulom:

$$u_{c,j} = v'_{w_j} \cdot h$$

gdje je j indeks čvora c-te izlazne riječi. S obzirom da izlazni sloj dijeli težine za svaku izlaznu riječ, vrijedi $u_{c,j} = u_j$. Primjenom *softmax* funkcije dobivamo multinomijalnu distribuciju:

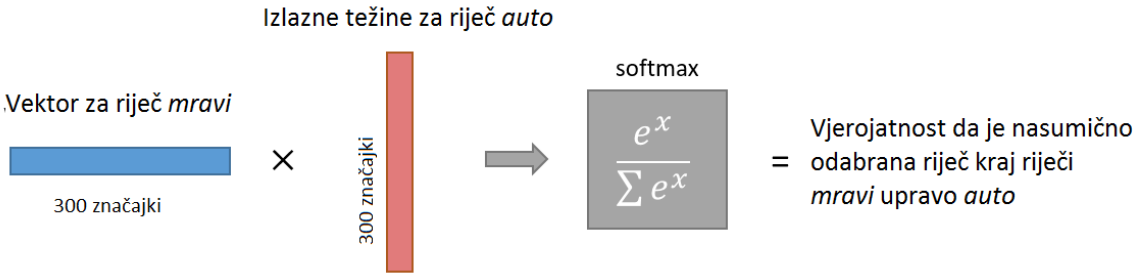


Slika 2.5: Prikaz matrice težina kao pregledne tablice za vektore riječi [13]

$$p(w_{c,j} = w_{O,c}|w_I) = y_{c,j} = \frac{exp(u_{c,j})}{\sum_{j'=1}^{V} exp(u_{j'})}$$

Vrijednost ove funkcije je upravo izračunata vjerojatnost za sve riječi da se nalaze u kontekstu ulazne riječi w te se prikaz tog izračuna za jedan par riječi nalazi na slici 2.6. Na slici 2.7 možemo vidjeti cjelokupnu arhitekturu te unaprijedni prolaz kroz mrežu.

Uzmimo za primjer vokabular koji se sastoji od pet riječi, a vektori i veličina prozora jednaka je tri. Uzmimo za naš ulazni primjer da je ciljna riječ bila druga u vokabularu s kodom $x = [0, 0, 1, 0, 0]$ te dvije kontekstne riječi $y_1 = [0, 1, 0, 0, 0]$ te $y_2 = [0, 0, 0, 1, 0]$. Uz matrice W i W'



Slika 2.6: Izračun izlazne vjerojatnosti [13]

$$W = \begin{bmatrix} 7 & 7 & 1 \\ 3 & 7 & 6 \\ 0 & 4 & 2 \\ 3 & 3 & 2 \\ 3 & 3 & 0 \end{bmatrix}, W' = \begin{bmatrix} 3 & 5 & 2 & 7 & 4 \\ 3 & 2 & 0 & 0 & 5 \\ 3 & 9 & 8 & 1 & 7 \end{bmatrix} \tag{2.4}$$

za skriveni sloj h dobivamo vrijednost:

$$\begin{aligned} h &= x^T W \\ &= [0, 0, 1, 0, 0]W \\ &= [0, 4, 2] \end{aligned} \tag{2.5}$$

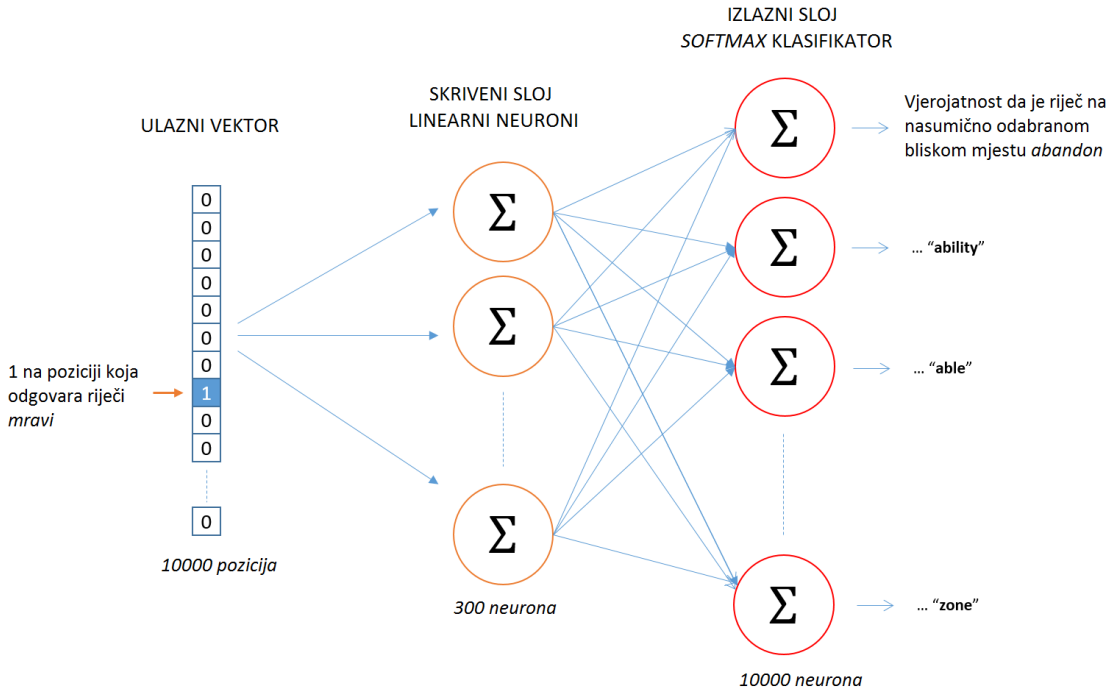
Distribucija vjerojatnosti na izlazu zatim postaje:

$$\begin{aligned} y' &= softmax(W'h) \\ &= softmax([18, 26, 16, 2, 34]) \\ &\approx [0, 0.003, 0, 0, 0.997] \end{aligned} \tag{2.6}$$

Naša točna rješenja su vrijednsti y_1 te y_2, pomoću metode gradijentnog spusta nad funkcijom gubitka za svaki par između dobivenog rezultata y' i y_i optimiziraju se matrice W i W' kako bi se smanjila greška.

2.2.1. Unaprijeđenja modela

Problemi prethodnih modela prvenstveno su dimenzije neuronske mreže. Vratimo se na prethodni primjer s vokabularom od 10000 riječi te vektore riječi dimenzije 300. Oba prethodna modela sastoje od matrica težina W i W', čije su dimenzije 300×10000, čime dolazimo do tri milijuna težina za svaku od tih matrica.



Slika 2.7: Arhitektura modela skip-gram [13]

Mreža takve veličine vrlo se sporo trenira pomoću gradijentnog spusta (engl. *gradient descent*) te bi zahtijevala ogromne količine podataka za učenje. S obzirom na veličinu mreže povećan je i njen kapacitet učenja te treba pažljivo trenirati mrežu da bi se spriječilo pretreniranje (engl. *overfitting*) [15].

Iz tog razloga, autori donose tri inovacije u njihovom drugom radu [15], koje ne samo da ubrzavaju treniranje modela, već i poboljšavaju kvalitetu rezultirajućih vektora riječi:

1. tretiranje uobičajenih parova riječi ili fraza kao jednu "riječ" u modelu,
2. poduzorkovanje (engl. *subsampling*) čestih riječi da bi se smanjio broj primjera za učenje,
3. modifikacija optimizacijskog cilja pomoću tehnike koju su nazvali negativno uzorkovanje (engl. *negative sampling*), koja dopušta da svaki primjer za učenje ažurira samo mali postotak svih težina modela.

2.2.2. Fraze kao riječi

S obzirom da u ovom radu razmatramo samo vektore riječi, a ne i vektore n-grama, prva inovacija nije direktno vezana za ovaj rad. Ona je proširenje modela na način da

izraze od više riječi reprezentiramo jednim vektorom, npr., mogli bismo očekivati da vektor za naziv *Krila Oluje* ima drastično različite vrijednosti od vektora za same riječi *krila* i *oluje*. Ova proširenje omogućava nam da očuvamo informacije učestalih fraza u korpusu bez međusobnog utjecaja s vektorima riječi od kojih se fraza sastoji.

2.2.3. Poduzorkovanje

Kako bismo objasnili razloge za drugu inovaciju, pogledajmo rečenicu *"Danas je teško biti lud"*. Kada bismo iz te rečenice izvukli ulazne primjere za naš model, jedan od njih bi bio par riječi *(biti, lud)*. Postoje dva glavna problema za uobičajene riječi poput riječi *biti*:

1. par *(biti, lud)* nam ne govori previše o značenju riječi *lud*. Dodatan razlog je što se riječ *biti* nalazi u kontekstu gotovo svih riječi,
2. postojat će puno više primjera *(biti, ...)* nego što nam je potrebno da bismo naučili dobar vektor za riječi *biti*.

Iz tog razloga uvedeno je poduzorkovanje čestih riječi što ih efektivno briše iz teksta. Vjerojatost da izbacimo riječ u relaciji je s njenom frekvencijom. Kada bismo imali prozor veličine 10, mičući samo tu jednu instancu riječi *biti* dobili smo dva poboljšanja koja upravo odgovaraju prethodno navedenim problemima:

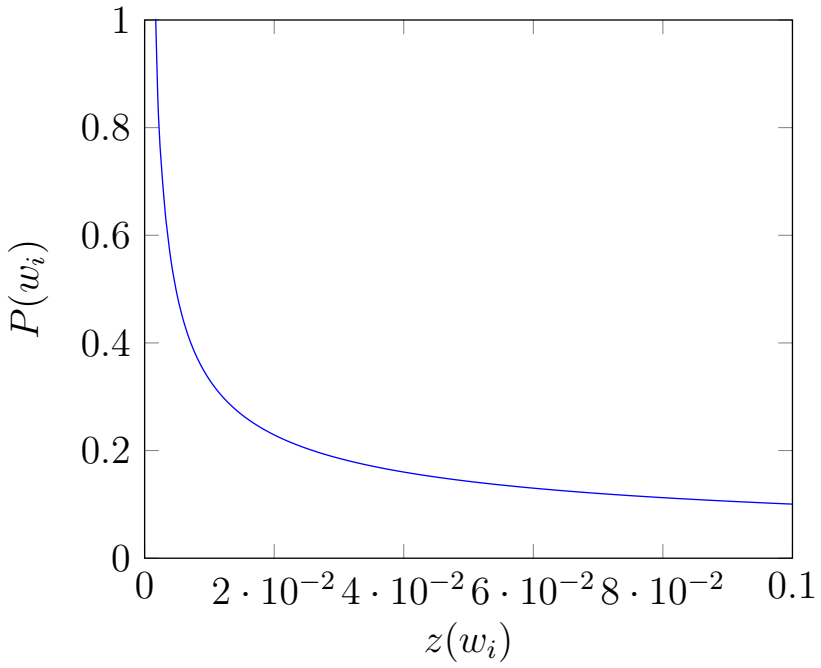
1. treniranje ostalih riječi izbjeći će pojavljivanje riječi *biti* u svom kontekstu,
2. imat ćemo deset primjera za učenje manje gdje je *biti* ulazna riječ.

Uzmimo u obzir da je w_i riječ, a $z(w_i)$ udio te riječi u korpusu. Kada bismo uzeli neku nasumičnu riječ, npr., *ispit*, koja se pojavljuje 1000 puta u korpusu od jedne milijarde riječi, njen udio bi bio $z('ispit') = 1e^{-6}$.

Iz tog razloga jedan od parametara modela je i mjera uzorkovanja *sample*, koja utječe na to koliko manje uzorkovanja će se događati. Tako dolazimo do formule vjerojatnosti da zadržimo riječ:

$$P(w_i) = \left(\sqrt{\frac{z(w_i)}{sample}} + 1\right) \cdot \frac{sample}{z(w_i)}$$

gdje je 0.001 vrijednost parametra uzorkovanja u originalnom radu. Iz formule je moguće vidjeti da se smanjivanjem vrijednosti parametra *sample* smanjuje i vjerojatnost zadržavanja riječi [15].



Slika 2.8: Graf vjerojatnosti zadržavanja riječi

S obzirom da u praksi niti jedna riječ ne čini značajno velik dio korpusa, na grafu vjerojatnosti zadržavanja riječi možemo gledati samo manje vrijednosti udjela riječi $z(w_i)$. Neke od zanimljivih točaka ove funkcije s originalnom vrijednosti parametra *sample* su:

- $P(w_i) = 1.0$, tj. riječ će sigurno biti zadržana za vrijednosti $z(w_i) \leqslant 0.0026$ što znači da će samo riječima čiji je udio veći od 0.26% korpusa će biti smanjen broj uzoraka
- $P(w_i) = 0.5$, tj. 50% vjerojatnosti za sačuvanje za vrijednost $z(w_i) = 0.00746$
- $P(w_i) = 0.033$, tj. 3.3% vjerojatnosti za vrijednost $z(w_i) = 1.0$ što bi značilo da je cijeli korpus izgrađen samo od riječi w_i što je u praksi nemoguće.

2.2.4. Negativno uzorkovanje

Treniranje neuronske mreže podrazumijeva uzimanje primjera za učenje te modificiranje težina mreže na način da se smanji iznos funkcije gubitka tog primjera pomoću metode gradijentnog spusta. Drugim riječima, svaki primjer za učenje modificira sve težine u mreži. S obzirom da je veličina vokabulara u prirodnim jezicima ogromna, treniranje bi iziskivalo modifikaciju težina za svaku riječ u vokabularu. Negativno uzorkovanje rješava ovaj problem tako da za svaki primjer za učenje modificiramo

samo malen postotak težina, a ne sve [15].

Uobičajenim postupkom, u primjeru za učenje oznaka je također definirana kao *1-od-N* vektor, što znači da je vrijednost samo jednog člana 1, dok je za sve druge jednaka 0. Pomoću negativnog uzorkovanja nećemo gledati sve te tisuće nula, već ćemo uzeti neki mali broj negativnih uzoraka za koje ćemo prilagoditi težine. Uzevši u obzir dimenzije izlaznog sloja iz prethodnih primjera, 300×10000, ovom promjenom ćemo modificirati samo težine naše pozitivne riječi te pet negativnih primjera čiji je izlaz jednak nuli. Time smo smanjili broj potrebnih gradijenata na samo šest izlaznih neurona, tj. 1800 težina, što je samo 0.06% od 3 milijuna težina u izlaznom sloju. U skrivenom sloju modificiraju se samo težine za ulaznu riječ, na što negativno uzorkovanje nema utjecaja [15].

Negativni uzorci izabrani su pomoću distribucije zadane formulom:

$$P(w_i) = \frac{f(w_i)^{3/4}}{\sum_{j=0}^{n} f(w_j)^{3/4}}$$

gdje $f(w_i)$ označava frekvenciju riječi što znači da će česte riječi također češće biti izabrane kao negativni uzorci. Razlog eksponenciranje frekvencije na $3/4$ je empirijski te je obrazložen boljim rezultatima u originalnom radu.

U dosadašnjem slučaju modela *Skip-gram*, cilj je bio maksimizirati log-izglednost funkcije:

$$\sum_{t=1}^{T} \sum_{c \in C_t} \log\ p(w_c|w_t)$$

gdje je C_t skup indeksa riječi iz konteksta riječi w_t, a T veličina kontekstnog prozora. Ako uzmemo u obzir funkciju bodovanja s koja preslikava parove riječi (riječ, kontekst) bodovima u $\mathbb{R}$, mogući izbor za definirati vjerojatnost kontekstne riječi je funkcija *softmax*:

$$p(w_c|w_t) = \frac{e^{s(w_t,w_c)}}{\sum_{j=1}^{W} e^{s(w_t,j)}}$$

Ovaj model implicira da za danu riječ w_t predviđamo samo jednu riječ konteksta w_c. Ovaj problem se može postaviti i kao skup nezavisnih binarnih klasifikacijskih zadataka gdje je cilj predvidjeti prisutnost ili neprisutnost kontekstnih riječi. Za riječ na poziciji t i kontekstom c nova negativna log-izglednost je:

$$\log(1 + e^{-s(w_t, w_c)}) + \sum_{n \in N_{t,c}} \log(1 + e^{s(w_t, w_n)})$$

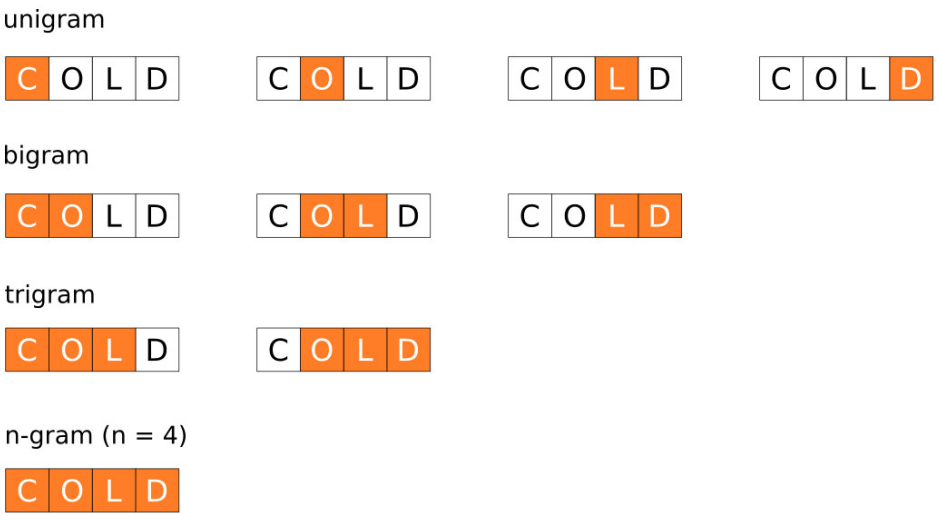
gdje je $N_{t,c}$ skup negativnih primjera koji su uzorkovani iz vokabulara. Uvođenjem logističke funkcije gubitka $l : x \mapsto \log(1 + e^{-x})$ dobivamo funkciju cilja:

$$\sum_{t=1}^{T} \Big(\sum_{c \in C_t} l(s(w_t, w_c)) + \sum_{n \in N_{t,c}} l(-s(w_t, n)) \Big)$$

Prirodna parametrizacija funkcije bodovanja između riječi w_t i kontekstne riječi w_c je korištenje skalarnog produkta između vektora riječi i konteksta $s(w_t, w_c) = u_{w_t}^T \cdot v_{w_c}$ kao što je bila i u prethodnim modelima [15].

2.3. FastText

FastText modeli razvijeni su kao pokušaji poboljšanja osnovnih modela *skip-gram* te modela kontinuirane vreće riječi. Postoji mnogo jezika koji su morfološki bogati poput turskog, finskog pa i hrvatskog. Takvi jezici sadrže puno riječi koje se pojavljuju rijetko, što otežava učenje dobrih reprezentacija na nivou riječi. U ovim modelima predloženo je učenje reprezentacija n-grama te posljedično reprezentacija riječi kao sume vektora n-grama. Nizovi od n elemenata koji se pojavljuju u nekom skupu, u ovom slučaju slova, nazivaju se n-grami. Primjer kako su riječi reprezentirane unigramima, bigramima te trigramima prikazan je na slici 2.9. Na taj način dosadašnji modeli prošireni su informacijom na nivou nižem od riječi [6].



Slika 2.9: Prikaz n-grama unutar riječi *cold* [5]

Koristeći jedinstvenu vektorsku reprezentaciju svake riječi, model *skip-gram* ignorira unutarnju strukturu riječi. Iz tog razloga predložena je drugačija funkcija bodovanja s kako bi se iskoristila i ta informacija. Za danu riječ w označimo s $G_w \subset \{1, ..., G\}$ skup n-grama koji se pojavljuju u riječi w gdje je G broj n-grama koji postoje u našem vokabularu. Svakom n-gramu prirodaje se vektorska reprezentacija z_g. Sada možemo izraziti riječ kao sumu vektorskih reprezentacija njezinih n-grama. Nova funkcija bodovanja je:

$$s(w, c) = \sum_{g \in G_w} z_g^T \cdot v_c \tag{2.7}$$

U skup n-grama uvijek se pridodaje i riječ w da bi se naučila i vektorska reprezentacija svake riječi, što znači da skup n-grama postaje nadskup vokabulara. Različite vektorske reprezentacije se pridodaju se riječima te n-gramima koji dijele isti niz znakova. Primjerice, riječi *te* i bigramu *te* iz riječi *test* biti će dodijeljeni različiti vektori. Ovaj jednostavan model omogućava dijeljenje reprezentacija među riječi što omogućuje učenje pouzdanih reprezentacija za rijetke riječi [6].

Ovaj model je vrlo jednostavan te dopušta dizajnerske odluke prilikom kreiranja skupa n-grama G_w. U originalnom članku iskorišteni su n-grami duljine od tri do šest znakova uključivo. Model dopušta i druge načine kreiranja n-grama, primjerice prefiksi i sufiksi riječi. Kako bi se razlikovali prefiksi i sufiksi riječi, u te n-grame se na početak za prefikse i kraj za sufikse dodaje poseban znak za identifikaciju [6].

Kako bi se smanjila memorijska potrošnja algoritma, koristi se *hash* funkcija koja preslikava n-grame na cijele brojeve od 1 do K, gdje je K u originalnoj implementaciji jednak dva milijuna. Kako bi se poboljšala efikasnost modela, za najčešćih P riječi u vokabularu ne koriste se n-grami za njihov prikaz. Ovaj potez može i negativno utjecati na model te autori upozoravaju da smanjenje broja P može doprinijeti kvaliteti, ali i smanjiti računalne performanse modela [6].

2.4. GloVe

Model *GloVe* (engl. *Global Vectors*) je regresijski model koji spaja dvije najpoznatnije obitelji modela – globalne metode za faktorizaciju matrica te metode lokalnog kontekstnog prozora. Ovaj model efikasno uči statističku informaciju učeći samo iz matrice supojavljivanja koristeći elemente matrice koji nisu jednaki nuli, umjesto na cijeloj rijetkoj matrici ili samo na individualnim kontekstnim prozorima [17].

Primarni izvor informacija za sve modele nenadziranog učenja vektora riječi je

statistika pojavljivanja riječi u korpusu. Glavno pitanje je kako dobiti značenje riječi iz te informacije te je iz tog razloga kreiran ovaj model koji direktno obuhvaća globalnu statistiku korpusa [17].

Uzmimo da je matrica supojavljivanja riječi (engl. *word-word cooccurrence matrix*) označena s X, gdje X_{ij} označava broj koliko se puta riječ j pojavila u kontekstu riječi i. Iz tog razloga s $X_i = \sum_k X_{ik}$ možemo označiti broj pojavljivanja svih riječi u kontekstu riječi i. Zatim možemo vjerojatnost pojavljivanja riječi j u kontekstu riječi i izraziti formulom $P_{ij} = P(j|i) = X_{ij}/X_i$ [17].

Neka aspekti značenja vidljivi su već i iz matrice vjerojatnosti supojavljivanja, što ćemo pokazati primjerom u tablici 2.1. Uzmimo u obzir dvije riječi i i j, koje iskazuju određeno područje interesa. Primjerice za područje termodinamičke faze uzmimo $i = led$ i $j = para$. Odnos između tih dviju riječi može se dobiti proučavanjem omjera njihovih vjerojatnosti supojavljivanja s raznim riječima ispitivanja, k. Za riječi k koje su vezane s riječi *led* poput *krutina* očekujemo da će omjer P_{ik}/P_{jk} biti velik. Također, iz istog razloga očekujemo da će za riječ $k = plin$, koja je vezana za paru, ali ne i za led, omjer biti malen. Za riječi poput *voda* i *moda*, koje su ili vezane za obje riječi ili za nijednu, očekujemo da je omjer što bliži jedan. U usporedbi sa sirovim vjerojatnostima, omjeri bolje prikazuju bitne riječi (*krutina* i *plin*) od irelevantnih (*voda* i *moda*) [17].

Vjerojatnost i omjer	$k = krutina$	$k = plin$	$k = voda$	$k = moda$
$P(k\|led)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|para)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|led)/P(k\|para)$	8.9	8.5×10^{-2}	1.36	0.96

Tablica 2.1: Tablica vjerojatnosti supojavljivanja za riječi *led* i *para*

Iz ovoga možemo zaključiti da su omjeri vjerojatnosti supojavljivanja dobar početak za kreiranje vektora riječi. Uzimajući u obzir da omjer P_{ik}/P_{jk} ovisi o tri riječi: i, j i k, najopćenitiju formu modela možemo zapisati kao:

$$F(w_i, w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}} \tag{2.8}$$

gdje su $w \in \mathbb{R}^d$ vektori riječi, a $\tilde{w} \in \mathbb{R}^d$ odvojeni kontekstni vektori. Desnu stranu ove jednadžbe možemo lako ekstrahirati iz korpusa. Za funkciju F postoji više mogućnosti, ali uzevši u obzir nekoliko preduvjeta dolazimo do jedinstvenog izbora. Željeli bismo da funkcija F enkodira informacije prisutne u omjeru P_{ik}/P_{jk} u vektorskom

prostoru riječi. S obzirom da su vektorski prostori inherentno linearne strukture, najprirodniji način je pomoću razlike vektora. Iz tog razloga možemo funkcije F ograničiti na funkcije razlike dvije riječi:

$$F(w_i - w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}} \tag{2.9}$$

Također uzmimo u obzir da su argumenti funkcije F vektori dok je rezultat s desne strane skalar. Iako bi funkcija F mogla biti parametrizirana na različite načine, npr., neuronskom mrežom, time bismo zamutili linearnu strukturu koju želimo obuhvatiti. Kako bismo izbjegli ovaj problem, možemo uzeti skalarni produkt argumenata

$$F((w_i - w_j)^T \tilde{w}_k) = \frac{P_{ik}}{P_{jk}} \tag{2.10}$$

što sprječava funkciju F od miješanja vektorskih dimenzija na bilo koji način. Kod tablice supojavljivanja riječi, nije moguće razlikovati običnu riječ od kontekstne riječi te ih slobodno mijenjamo. Kako bismo to dosljedno mogli činiti, moramo zamijeniti ne samo $w \leftrightarrow \tilde{w}$ već i $X \leftrightarrow X^T$. Konačan model morao bi biti invarijantan na takve zamjene, što trenutačno nije podržano posljednjom formulom. Simetričnost se može povratiti u dva koraka. Prvi korak je zahtjev da funkcija F bude homomorfizam između dvije grupe $(\mathbb{R}, +)$ i $(\mathbb{R}_{>0}, \times)$:

$$F((w_i - w_j)^T \tilde{w}_k) = \frac{F(w_i^T \tilde{w}_k)}{F(w_j^T \tilde{w}_k)} \tag{2.11}$$

što je pomoću jednadžbe 2.10 rješivo s:

$$F(w_i^T \tilde{w}_k) = P_{ik} = \frac{X_{ik}}{X_i} \tag{2.12}$$

dobivamo rješenje jednadžbe 2.11 pomoću funkcije $F = \exp$:

$$w_i^T \tilde{w}_k = \log(P_{ik}) = \log(X_{ik}) - \log(X_i) \tag{2.13}$$

Iz jednadžbe 2.13 vidljivo je da bismo postigli simetričnost ukoliko ne bi postojao član $\log(X_i)$, ali s obzirom da je taj član neovisan o k, može se apsorbirati u pristranost b_i za w_i. Napokon, dodavanjem pristranosti osigurana je simetričnost:

$$w_i^T \tilde{w}_k + b_i + \tilde{b}_k = \log(X_{ik}) \tag{2.14}$$

Ova jednadžba loše je definirana s obzirom da logaritam divergira u slučaju da je njegov argument nula. Jedno rješenje ovog problema je dodatak aditivnog pomaka u

logaritmu, $\log(X_{ik}) \rightarrow \log(1 + X_{ik})$, koji omogućava očuvanje rijetkosti matrice, ali izbjegava divergenciju logaritma. Ideja faktorizacije matrice supojavljivanja bliska je metodi LSA (engl. *latent semantic analysis*). Glavna mana tog modela je što smatra da su sva supojavljivanja jednaka, čak i ona koja se događaju rijetko ili se ne događaju nikada. Takva supojavljivanja nose svoj šum te sadrže manje informacija od onih čestih iako nul-elementi konstituiraju 75-90% podataka u X, ovisno o korpusu i vokabularu [17].

U ovom modelu predložen je novi težinski regresijski model najmanjih kvadrata koji adresira te probleme. Prikazujući jednadžbu 2.14 kao problem najmanjih kvadrata te uvođenjem težinske funkcije $f(X_{ij})$ u funkciju troška, dobivamo model:

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - logX_{ij})^2 \tag{2.15}$$

gdje je V veličina vokabulara. Težinska funkcija mora poštovati sljedeća pravila:

1. $f(0) = 0$. Ako na f gledamo kao kontinuiranu funkciju, morala bi nestati s $x \rightarrow 0$ dovoljno brzo da je $\lim_{x \rightarrow 0} f(x) \log^2 x$ je konačan,
2. $f(x)$ mora biti nepadajuća kako rijetka supojavljivanja ne bi imala velike težine,
3. $f(x)$ mora biti relativno malen za velike vrijednosti x kako česta supojavljivanja ne bi imala preveliku težinu.

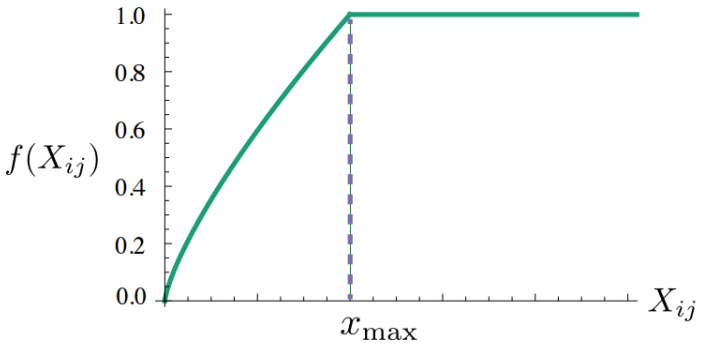
Iako velik broj funkcija ispunjava ove uvjete, autori su se odlučili za obitelj funkcija definiranih kao:

$$f(x) = \begin{cases} (x/x_{max})^\alpha & \text{ako } x < x_{max} \\ 1 & \text{inače} \end{cases} \tag{2.16}$$

U radu su korištene vrijednosti $x_{max} = 100$ te $\alpha = 3/4$, koje su empirijski određene. Na slici 2.10 nalazi se prikaz funkcije f za prethodne vrijednosti, gdje vidimo kako ova funkcija zadovoljava sve uvjete.

2.5. Model *SN*

Model *SN* posljednji je model u nizu koji će biti ispitan u ovom radu. Njegovo ime dolazi od engleskog naziva ***S**quared **N**orm*, koji se odnose na njegovu normalizaciju težina vektora riječi u funkciji cilja [3].



Slika 2.10: Graf funkcije f za $\alpha = 3/4$ [17]

Postoje neke nelinearne metode za izračun novih težina matrice supojavljivanja, poput uzajamne informacije točke (engl. *pointwise mutual information, PMI*). Najjednostavnija verzija kreće od matrice gdje je svaki redak i stupac indeksiran jednom riječju. Sadržaj matrice za par (w, w') jest:

$$PMI(w, w') = \log \frac{p(w, w')}{p(w)p(w')} \tag{2.17}$$

gdje je $p(w, w')$ empirijska vjerojatnost da se riječi w i w' nalaze unutar prozora određene veličine u korpusu, a $p(w)$ je marginalna vjerojatnost riječi w. Empirijski se pokazalo da je matricu PMI moguće blisko aproksimirati pomoću matrice puno manjeg ranga, čime dolazimo do tvrdnje:

$$\langle w, w' \rangle \approx PMI(w, w')$$

Model tretira korpus kao dinamički proces u kojem se t-ta riječ proizvodi u koraku t. Proces pokreće nasumična šetnja (engl. *random walk*) vektora diskursa $c_t \in \mathbb{R}^d$. Koordinate vektora diskursa su reprezentacija onoga o čemu se govori u tekstu, tj. teme. Svaka riječ ima vremenski invarijantan latentni vektor $v_w \in \mathbb{R}^d$ koji obuhvaća njezinu povezanost s vektorom diskursa. Ovakvo ponašanje modelira se log-linearnim modelom produkcije riječi:

$$\Pr[w \text{ emitiran u vrijeme } t | c_t] \propto \exp(c_t, v_w) \tag{2.18}$$

Log-linearni model je matematički model koji je reprezentiran funkcijom čiji logaritam je linearna kombinacija parametara:

$$\exp\left(c + \sum_i w_i f_i(X)\right) \tag{2.19}$$

Vektor diskursa radi sporu nasumičnu šetnju, što znači da dobivamo c_{t+1} dodavanjem malog nasumičnog vektora na vektor c_t tako da se bliske riječi generiraju uz sličan vektor diskursa. S obzirom na to da nas zanimaju vjerojatnosti supojavljivanja dvije riječi, povremeni veliki skokovi tokom nasumične šetnje su dopušteni jer imaju zanemariv utjecaj na te vjerojatnosti. Model se temelji na dvije jednadžbe:

$$\log\ p(w, w') = \frac{||v_w + v_{w'}||_2^2}{2d} - 2\log Z \pm \epsilon, \tag{2.20}$$

$$\log\ p(w) = \frac{||v_w||_2^2}{2d} - \log Z \pm \epsilon \tag{2.21}$$

što zajednički implicira:

$$PMI(w, w') = \frac{\langle v_w, v_{w'} \rangle}{d} \pm O(\epsilon) \tag{2.22}$$

Uzmimo da je $X_{w,w'}$ broj supojavljivanja riječi u prozoru određene veličine u korpusu, a vjerojatnosti su nam poznate iz prethodnih formula. Uspješni uzorci nasumične šetnje nisu nezavisni. Ukoliko se nasumična šetnja miješa prilično brzo (vrijeme miješanja ovisi o logaritmu veličine vokabulara), onda distribucija $X_{w,w'}$ postaje vrlo bliska multinomijalnoj distribuciji Mul$(L, \{p(w, w)'\})$, gdje je $L = \sum_{w,w'} X_{w,w'}$ ukupan broj parova riječi. Pomoću ove aproksimacije dolazimo do vrijednosti najveće izglednosti za vektore riječi koje dobivamo optimizacijom funkcije cilja:

$$\min_{\{v_w\},C} \sum_{w,w'} X_{w,w'}(log(X_{w,w'}) - ||v_w + v_{w'}||_2^2 - C)^2 \tag{2.23}$$

Upravo ova funkcija cilja naziva se kvadratnom normom. Empirijski je pokazano da je za učestale riječi potrebno ograničiti $X_{w,w'}$ srezivanjem na $\min(X_{w,w'}, X_{max})$, gdje je X_{max} jednak 100 [3].

3. Ispitivanje modela

U ovom poglavlju napravit ćemo odabir korpusa te parametarsku pretragu prostora za prethodnih šest modela. Ovo ispitivanje temelji se na [9] te ispituje parametre kao što su odabir korpusa, dimenzija vektora, veličina vektorskog prostora te redukcija dimenzionalnosti. S druge strane, neke parametarske opcije poput usmjerenja i načina odabira konteksta nisu ispitani. Popis svih parametara nalazi se u tablici 3.1. Parametre ćemo ispitivati slijedno kako su navedeni u tablici. Najbolje modele iz svakog koraka koristit ćemo kao temelj optimizacije sljedećeg parametra zbog čega ovo dijeli svojstva s pohlepnim pretraživanjem parametarskog prostora.

Parametar	Vrijednosti
Skup podataka za učenje	hrWaC2, wiki
Dimenzija vektora riječi	100, 250, 500, 1000
Veličina kontekstnog prozora	3, 5, 7, 9, 10, 15
Broj reduciranih dimenzija	0, 1, 3, 5

Tablica 3.1: Vrijednosti korištene za parametarsku pretragu modela

Ispitivanje ćemo napraviti pomoću tri načina za evaluaciju vektora riječi. Prvi način evaluacije je izabrati nekoliko nasumičnih riječi te pomoću modela izvući neki broj najsličnijih riječi prema naučenom vektoru. Iako ovaj način evaluacije nije formaliziran te ga ne možemo koristiti za usporedbu kvalitete modela, svejedno iz njega možemo dobiti povratnu informaciju je li model naučio neke konkretne sličnosti riječi.

3.1. Odabir sinonima

Drugi način evaluacije koji je korišten je izbor sinonima. Korišten je skup podataka *hr-synonym-choice* koji sadrži 3000 pitanja [18]. Pitanja su podijeljena u tri područja – po jedno za imenice, glagole i pridjeve. Za danu riječ ponuđene su četiri druge riječi između kojih se mora odabrati sinonim. Pristup koji je korišten za evaluaciju na ovom

skupu podataka je kosinusna sličnost riječi i potencijalnih sinonima. Potencijalno rješenje je ona riječ koja nosi maksimalnu kosinusnu sličnost vektora s početnom riječi te je mjera koja je korištena za evaluaciju točnost na ovom skupu. Primjer iz ovog skupa podataka može se vidjeti u tablici 3.2. U tablicama koje prikazuju točnosti modela na ovom skupu dodatno su prikazani intervali 95%-tne pouzdanosti za točnosti dva najbolja parametra po modelu i razlike u njihovim preciznostima dobivene *bootstrap* ispitivanjem u 1000 iteracija te razlika u točnosti dva najbolja parametra po modelu.

Vrsta Riječi	Upitna riječ	Ponuđene riječi	Indeks odgovora
Imenice	razred	iredentizam, strpljivost, učionica, služnost	2
Glagoli	micati	izdubiti, emigrirati, maknuti, poviti	2
Pridjevi	kaotičan	ciklički, zbrkan, zadrt, brijači	1

Tablica 3.2: Primjer iz skupa *hr-synonym-choice*

3.2. Analogije među riječima

Treći način evaluacije koji se koristi za evaluaciju modela je jedan od najuobičajenih – analogijska pitanja. Konkretni skup podataka za hrvatski jezik koji je korišten za ovu evaluaciju je *croanalogy* [19]. Sastoji se od dva podskupa, jedan koji se sastoji od glavnih gradova i država te drugi s komparacijom pridjeva koji ukupno čine 856 analogijskih pitanja. Analogijsko pitanje sastoji se od četiri riječi u odnosu $a : b = c : d$ gdje tražimo riječ d pomoću vektora najsličnijeg vektoru $\tilde{d} = b - a + c$. Primjer iz ovog skupa podataka može se vidjeti u tablici 3.3.

Vrsta Riječi	a	b	c	d
Gradovi i države	Atena	Grčka	Bagdad	Irak
Komparacija pridjeva	bogat	bogatiji	debeo	deblji

Tablica 3.3: Primjer skupa *croanalogy*

3.3. Slične riječi

S obzirom da prvi način evaluacije ne možemo direktno iskoristiti za usporedbu modela, navest ćemo samo njegov primjer za jednu grupu modela.

Model	*ispit*
CBoW	kolokvij, prijemni, prijamni, predispit, predrok
Skip-gram	kolokvij, predrok, kolokviranje, prijemni, predispit
FastText CBoW	ecdl-ispit, međuispit, podispit, prispit, među-ispit
FastText Skip-gram	predispit, kolokvij, predbolonja, kolokvirati, podispit
GloVe	kolokvij, semestar, polagati, matura, završen
SN	semestar, kolokvij, matura, polagati, prijemni

Tablica 3.4: Prikaz 5 najsličnijih riječi za danu riječ *ispit* po modelima

Model	*sunce*
CBoW	mjesečina, zalazeći, svjetlost, svijetlost, zubato
Skip-gram	sunčev, zraka, zalazeći, zapadajući, zalazeči
FastText CBoW	@sunce, sunce35, sunce55, sunce.ne, sunceve
FastText Skip-gram	sunče, sunčev, sunčeva, sunčav, sunčen
GloVe	nebo, sunčev, zraka, svjetlost, obasjati
SN	nebo, svjetlost, zraka, obasjati, vjetar

Tablica 3.5: Prikaz 5 najsličnijih riječi za danu riječ *sunce* po modelima

Iz tablica 3.4 i 3.5 vidljivo je da su svi modeli na neki način naučili vektore riječi. Većina ih je naučila sinonime (koji se nalaze u sličnim kontekstima kao i zadana riječ) te riječi koje su usko povezane za radnje vezane s tom riječi (polagati, kolokvirati, itd.) ili pak nepravilne varijacije te riječi. Iz modela *FastText* jasno je vidljiv utjecan n-grama u modelu. Sve najsličnije riječi sadrže n-grame od 4–5 slova koji su upravo jednaki originalnoj riječi, a s obzirom da svi n-grami preko sume u jednakoj mjeri doprinose značenju riječi, ovakvo ponašanje je očekivano za te modele.

3.4. Skupovi podataka za učenje

U početnom eksperimentu tražen je skup podataka koji donosi najbolje rezultate za učenje modela. Korištena su dva različita skupa podataka u šest varijanti – *hrWaC2* [11] te članci hrvatske wikipedije. Iz ova dva osnovna skupa podataka pretprocesiranjem su kreirana četiri nova skupa podataka od kojih se dva odnosne na lematizirane skupove podataka dok su druga dva lematizirani te označeni POS (engl. *Part-of-speech*) oznakama pomoću alata za pretprocesiranje. Lematizacija je proces transformiranja inflekcijskih oblika riječi u njihov originalni, rječnički format. Time dobivamo značajno smanjenje vokabulara te imamo više primjera za učenje istih riječi. POS oznake riječi su gramatičke oznake njenih osobina te opisuju vrstu rijel, rod, broj te lice riječi. Dodatkom POS oznaka povećava se vokabular korpusa jer se neke riječi pojavljaju u raznim oblicima. Za lematizaciju i dobivanje POS oznaka korišten je *Reldi Tagger* [12]. Skup podataka *hrWaC2* dobiven je akumuliranjem web sadržaja hrvatskog jezika. Sadrži oko $63M$ rečenica na hrvatskom jeziku. Skup podataka $wiki$ preuzet je sa službene stranice *Wikipedije* te se sastoji od 13751 članka na hrvatskom jeziku. Iz tablica 3.6 i 3.7 moguće je vidjeti koliko je povećanje vokabulara korpusa smanjilo moć modela jer za mnogo riječi koje se nalaze u ispitnim skupovima nije postojao dovoljan broj ponavljanja u korpusu da se ubroje u vokabular.

Osim veličine vokabulara te pokrivenosti testova, za odabir skupa podataka za učenje bitni su i rezultati modela na samim ispitnim skupovima. Rezultati su prikazani na tablicama 3.8 i 3.9.

Iz rezultata na skupu *hr-synonym-choice* vidljivo je da je da *hrWaC2* skup primjera za učenje daje bolje rezultate od skupa *wiki*. S obzirom da ima značajno više primjera za učenje može mnogo točnije odrediti vektore riječi dok lematizacija dodatno doprinosi smanjenju vokabulara te učenju vektora. U usporedbi dva najbolja parametra svakog modela uvijek su bili primjerci *hrWaC2* skupa, ali je statistički pokazano da je razlika među njima nesignifikantna. U ovom ispitivanju pokazalo se da *FastText*

Skup primjera za učenje	Veličina vokabulara
wiki	270 160
wiki lema	161 302
wiki lema + POS	326 580
hrWaC2	1 520 167
hrWaC2 lema	1 024 906
hrWaC2 lema + POS	2 254 005

Tablica 3.6: Veličina vokabulara po skupu podataka za učenje s minimalnim brojem pojavljivanja $min\_count = 5$

Skup primjera za učenje	hr-synonym-choice	croanalogy
wiki	12.47	92.76
wiki lema	51.03	99.30
wiki lema + POS	6.97	29.32
hrWaC2	95.40	**100**
hrWaC2 lema	**98.67**	**100**
hrWaC2 lema + POS	6.97	56.54

Tablica 3.7: Pokrivenost ispitnih skupova podataka u postotcima

	CBoW	**Skip-gram**	**FT CBoW**	**FT Skip-gram**	**GloVe**	**SN**
hrWaC2	76.14	79.11	67.92	81.48	73.31	59.92
hrWaC2 lema	**81.15**	**82.03**	**71.99**	**81.86**	**83.92**	**74.53**
hrWaC2 lema + POS	73.20	80.35	64.97	81.56	73.37	56.96
wiki	58.56	63.90	59.36	70.86	50.53	39.30
wiki lema	66.36	66.34	60.81	69.37	61.66	50.36
wiki lema + POS	58.37	63.16	44.50	72.73	50.24	38.28
Najbolji model	[79.88, 82.60]	[80.80, 83.56]	[70.37, 73.58]	[80.26, 83.07]	[82.80, 85.24]	[73.23, 75.97]
Drugi najbolji	[74.64, 77.63]	[78.67, 81.93]	[66.14, 69.44]	[80.13, 82.88]	[71.56, 75.13]	[58.23, 61.66]
Interval razlike	[3.69, 6.62]	[0.14, 3.37]	[2.51, 5.71]	[-1.00, 1.64]	[8.70, 12.50]	[12.91, 16.20]
Razlika	5.01	1.68	4.07	0.3	10.55	14.61

Tablica 3.8: Točnost modela po skupu za učenje na skupu *hr-synonym-choice* u postotcima

inovacije ne doprinose boljem rezultatu u odnosu na originalne CBoW i Skip-gram modele.

	CBoW	**Skip-gram**	**FT CBoW**	**FT Skip-gram**	**GloVe**	**SN**
hrWaC2	22.78	29.67	22.78	28.50	18.46	18.34
hrWaC2 lema	35.16	35.98	34.70	35.98	35.05	35.05
hrWaC2 lema + POS	14.67	23.55	27.89	56.54	19.83	16.32
wiki	3.02	13.48	13.60	14.10	0.88	3.02
wiki lema	**58.60**	**58.80**	**59.40**	**59.80**	**58.40**	**58.00**
wiki lema + POS	4.38	5.58	44.62	36.25	0.40	3.19

Tablica 3.9: Točnost modela po skupovima podataka za učenje na skupu *croanalogy* u postotcima

Na skupu *croanalogy* pokazalo se da za sve modele skup primjera za učenje *wiki* s lematizacijom ipak daje bolje rezultate. Mogući razlog za bolje performanse učenjem na skupu hrvatske Wikipedije je razlika u veličini vokabulara te strukturiranosti skupa za učenje. Vokabular modela učenih na lematiziranom skupu *hrWaC2* više od šest puta je veći od vokabulara modela učenih na lematiziranoj Wikipediji. Time se za isti broj dimenzija vektora u modelu dobiva puno gušće naseljen vektorski prostor, što smanjuje preciznost metoda temeljenih na sličnosti vektora. S obzirom da su na prvom ispitnom skupu svi modeli pokazali značajno bolje rezultate na lematiziranom skupu *hrWaC2* te je pokrivenost oba ispitna skupa najbolja za taj skup primjera za učenje, u daljnjem ispitivanju koristit će se upravo taj.

3.5. Dimenzija vektora

Odabir dimenzije vektora jedan je od najbitnijih parametara modela. Odabirom premale dimenzije vektora onemogućavamo modelu učenje bitnih informacija dok odabirom prevelike dimenzije ne dobivamo značajna poboljšanja u kvaliteti, ali smanjujemo performanse modelima koji se kasnije vežu na vektore riječi.

	CBoW	**Skip-gram**	**FT CBoW**	**FT Skip-gram**	**GloVe**	**SN**
50	77.70	78.41	69.63	78.01	80.41	72.40
100	81.15	82.03	71.99	81.86	83.92	74.53
250	**84.12**	**84.83**	73.45	83.61	85.34	**74.93**
500	83.61	84.02	73.73	84.49	**86.39**	74.86
1000	83.99	83.45	**74.90**	**85.24**	86.28	**74.93**
Najbolji model	[82.88, 85.34]	[83.43, 86.00]	[73.51, 76.47]	[84.16, 86.43]	[85.26, 87.47]	[73.58, 76.36]
Drugi najbolji	[82.93, 85.29]	[82.44, 85.29]	[72.09, 75.30]	[83.22, 85.67]	[85.03, 87.32]	[73.45, 76.34]
Interval razlike	[-0.68, 0.78]	[0.13, 1.48]	[0.51, 1.96]	[-0.01, 1.48]	[-0.61, 0.74]	[-0.54, 0.54]
Razlika	0.13	0.81	1.17	0.75	0.11	0.00

Tablica 3.10: Točnosti modela po dimenziji vektora na skupu *hr-synonym-choice* u postotcima

	CBoW	**Skip-gram**	**FT CBoW**	**FT Skip-gram**	**GloVe**	**SN**
50	34.93	35.86	33.53	35.40	34.58	34.34
100	35.16	**35.98**	34.70	**35.98**	35.05	35.05
250	**35.51**	35.04	**34.81**	34.69	**35.16**	**35.16**
500	34.70	34.35	34.23	34.35	34.46	35.05
1000	34.35	34.35	34.35	34.35	34.35	34.70

Tablica 3.11: Točnosti modela po dimenziji vektora na skupu *croanalogy* u postotcima

U tablicama 3.10 i 3.11 vidljivi su rezultati modela učenih s različitim dimenzijama vektora. Na oba skupa podataka i kroz većinu modela vidljivo je da su rezultati najbolji upravo za veličinu vektora riječi od 250 dimenzija što pripada u neku srednju veličinu. Jedini modeli koji su se pokazali bolji s najvećom dimenziju su upravo *FastText* modeli kojima veća dimenzija omogućava preciznije učenje n-grama (kojih ima mnogo više od riječi) te na taj način doprinosi rezultatu modela. Ipak, statističkim ispitivanjem pokazalo se da su razlike među najbolja dva parametra svakog modela na skupu *hr-synonym-choice* u svim slučajevima nesignifikantne.

Zbog dobivenih točnosti te uzimajući u obzir da povećanje broja dimenzija negativno utječe na performanse modela koji će se nadovezivati na vektore riječi, za daljnje

	CBoW	Skip-gram	FT CBoW	FT Skip-gram	GloVe	SN
3	**84.56**	**85.91**	**75.88**	**85.41**		
5	84.19	85.00	74.53	84.63	85.14	73.18
7	84.12	84.83	73.45	83.61	85.34	74.93
9	83.89	84.22	73.07	82.74	85.81	74.86
10					85.57	76.15
15					**86.52**	**77.13**
Najbolji model	[83.44, 85.87]	[84.58, 87.16]	[74.25, 77.30]	[84.21, 86.77]	[85.11, 87.76]	[75.73, 78.70]
Drugi najbolji	[82.89, 85.69]	[83.83, 86.34]	[73.01, 76.11]	[83.43, 85.91]	[84.56, 87.21]	[74.82, 77.61]
Interval razlike	[-0.27, 0.88]	[0.17, 1.63]	[0.61, 2.03]	[0.07, 1.42]	[-0.27, 1.52]	[0.10, 1.72]
Razlika	0.37	0.91	1.35	0.78	0.71	0.98

Tablica 3.12: Točnosti modela po veličini prozora na skupu *hr-synonym-choice* u postotcima

ispitivanje i održavanje ravnopravnosti među modelima odabrani su vektori veličine 250.

3.6. Veličina prozora

Veličina prozora (engl. *window size*) također je bitan parametar modela. Ona određuje širinu konteksta oko ciljne riječi koju uzimamo u obzir pri izračunu vektora riječi. Proširenje konteksta može i poboljšati i pogoršati performanse modela. Iako povećanjem povećavamo broj riječi u kontekstu te dopuštamo da se i u malo širem kontekstu pojave bitne riječi za značenje ciljne riječi, također dopuštamo i drugim riječima koje nisu dio konteksta te su zapravo šum da uđu u kontekst i utječu na vektor. S obzirom da je u [17] navedena veličina prozora od 15, a u [3] veličina 10, ispitane su i te dvije veličine za te faktorizacijske modele. Iznosi veličine prozora koji nisu u općenitoj uporabi za modele nisu ispitani te su u odgovarajućim tablicama obojani sivom bojom.

Po rezultatima iz tablice 3.12 na skupu *hr-synonym-choice* pokazalo se da za modele *CBoW* i *Skip-gram* te njihove *FastText* nadogradnje bolje rezultate donosi prozor manje veličine. S druge strane metodama *GloVe* te *SN* veći prozor je donio bolje rezultate, što je u skladu s tvrdnjama autora iz originalnih radova. Ipak, statističkim ispitivanjem pokazalo se da su razlike među najbolja dva parametra za svaki model statistički nesignifikantna.

Rezultati tablice 3.13 na skupu *croanalogy* pokazali su malo nepravilnije rezultate. Za sve modele osim modela *FastText* pokazalo se da veći kontekstni prozor donosi bolje rezultate za analogijska pitanja. S druge strane, modeli *FastText* pokazali su bolje rezultate za što manji prozor, unatoč tome što je *FastText Skip-gram* pokazao

	CBoW	Skip-gram	FT CBoW	FT Skip-gram	GloVe	SN
3	**35.28**	**34.81**	**35.63**	**34.81**		
5	35.63	34.93	35.28	34.70	34.58	**34.81**
7	35.51	35.04	34.81	34.69	35.16	35.16
9	35.75	35.16	34.35	34.81	35.16	35.40
10					34.81	35.28
15					**35.40**	**35.86**

Tablica 3.13: Točnosti modela po veličini prozora na skupu *croanalogy* u postotcima

jednake rezultate za najmanji i najveći ispitani prozor. Prvenstveno možemo vidjeti da su rezultati na skupu *croanalogy* vrlo bliski te unutar 1% bez obzira na veličinu korištenog prozora.

3.7. Postprocesiranje vektora

U članku [16] autori predlažu metode za poboljšanje vektora riječi postprocesiranjem. Autori su eksperimentima pokazali da svi ispitani modeli vektora riječi imaju srednju vrijednost različitu od nule te dijele velik zajednički vektor čija norma postiže vrijednost i do pola ukupne norme prosječnog vektora riječi. Micanjem zajedničkog vektora prosječne vrijednosti reprezentacije su daleko od izotropnih. Većina energije vektora riječi sadržana je u malom vektorskom podprostoru, npr., 8 dimenzija od 300.

S obzirom da sve riječi dijele zajednički vektor i imaju zajedničke dominirajuće smjerove koji utječu na reprezentacije riječi, autori predlažu micanjem tih smetnji u dva koraka:

1. Oduzimanje vektora srednjih vrijednosti od svih vektora riječi kako bi se smanjila energija,

2. Projiciranje reprezentacija od dominantnih smjerova D, čime je efektivno smanjena dimenzija vektora.

Dominantni smjerovi vektora riječi dobivaju se izračunom analize glavnih komponenti (engl. *Principal component analysis, PCA*), koja određuje smjerove u vektorskom prostoru za koje vektori imaju najveću varijancu. Nove vektore $v'(w)$ dobivamo sljedećim koracima:

1. Izračunati srednju vrijednost vektora $v(w), w \in V$

$$\mu \leftarrow \frac{1}{|V|} \sum_{w \in V} v(w); \quad \tilde{v}(w) \leftarrow v(w) - \mu$$

2. Izračunati PCA-komponente:

$$u_1, ..., u_d \leftarrow PCA(\{\tilde{v}(w), w \in V\})$$

3. Postprocesiranje reprezentacija riječi:

$$v'(w) \leftarrow \tilde{v} - \sum_{i=1}^{D} (u_i^T v(w)) u_i$$

gdje je D broj glavnih komponenti kojih se želimo riješiti. U originalnom radu preporučena vrijednost $D = d/100$, gdje je d dimenzija vektora riječi. S obzirom da u našem slučaju radimo s vektorima veličine 250, uspoređene su manje vrijednosti parametra d u tablicama 3.14 i 3.15.

	CBoW	**Skip-gram**	**FT CBoW**	**FT Skip-gram**	**GloVe**	**SN**
0	84.56	85.91	75.88	85.41	**86.52**	77.13
1	84.32	86.76	80.34	87.16	86.15	**85.44**
3	85.84	86.79	81.32	86.89	85.34	79.59
5	**86.05**	**87.60**	**82.94**	**87.84**	85.64	78.72

Tablica 3.14: Rezultati modela po broju eliminiranih dimenzija na skupu *hr-synonym-choice* u postotcima

Rezultati iz tablice 3.14 pokazali su da za modele koji uče vektore riječi pomoću predviđanja drugih riječi postprocesiranje donosi malen doprinos na zadacima izbora sinonima. S druge strane za model *GloVe* pokazalo se da smanjenje dimenzija smanjuje i ekspresivnost modela te se rezultati pogoršavaju. Za model *SN* pokazalo se da je eliminacije 1 dimenzije značajno doprinijela rezultatu na skupu odabira sinonima.

S druge strane postprocesiranje vektora pokazalo je značajno lošije rezultate na zadatku analogija. Svi modeli osim modela *FastText Skip-gram* pokazali su značajno bolje rezultate sa svojim originalnim dimenzijama. Modeli *GloVe* te *SN* eliminacijom glavnih dimenzija gube svu svoju ekspresivnost na ovom zadatku. Dodatnim ispitivanjem pokazalo se da gubitkom više od 1 dimenzije na većini modela posustaje i sličnost

	CBoW	Skip-gram	FT CBoW	FT Skip-gram	GloVe	SN
0	**35.28**	**34.81**	**35.63**	34.81	**35.40**	**35.86**
1	29.79	23.48	24.30	34.81	0.00	0.00
3	26.51	24.07	18.57	35.28	0.00	0.00
5	21.14	20.56	1.75	**35.51**	0.00	0.00

Tablica 3.15: Rezultati modela po broju eliminiranih dimenzija na skupu *croanalogy* u postotcima

riječi, tj. da za vektore slične vektoru riječi više ne dobivamo semantički slične riječi već nasumične.

U usporedbi ovih rezultata s rezultatima iz rada [9] vidljivo je da je odabir korpusa te veličine kontekstnog prozora u oba rada dao pozitivan doprinos modelima. Ipak, u ovom radu pokazalo se da redukcijom dimenzionalnosti vektora nisu postignuta poboljšanja na našim ispitnim podacima što je suprotno od rezultata dobivenih u radu [9].

4. Linearnost značenja riječi i polisemična primjena

Značenje polisemičnih riječi predstavljeno jednim vektorom riječi prilično je nejasno. U radu [4] je pokazano da više značenja može postojati unutar jednog vektora u obliku linearne superpozicije te ih je moguće dohvatiti jednostavnim rijetkim kodiranjem.

U tom radu predloženo je da su vektori riječi linearna kombinacija značenja, tj. za vektor v_{list} vrijedi:

$$v_{list} \approx \alpha_1 \cdot v_{list1} + \alpha_2 \cdot v_{list2} + \alpha_3 \cdot v_{list3} + \cdots \tag{4.1}$$

iako je ovo nedovoljno da se matematički odrede značenja s obzirom da se v_{list} može izraziti na bezbroj načina u tom obliku. S obzirom da značenja $list1$ i $list2$ odgovaraju različitim distribucijama riječi koje se pojavljuju oko riječi $list$, možemo ih smatrati različitim diskursima. Diskursom možemo smatrati značenje ili temu o kojoj se govori u tekstu, a u našem slučaju oni se reprezentiraju smjerovima u vektorskom prostoru kao i riječi. Pretpostavljamo da je uz riječ $list$ moguće pronaći diskurs $novine$ koji ima veliku vjerojatnost za značenje $list1$ te druge povezane riječi poput $novine$.

Iz tog razloga prethodnu jednadžbu možemo napisati na drugačiji način pomoću parametra rijetkosti k, gornje granice m te skupa jediničnih vektora $A_1, A_2, ..., A_m$ tako da vrijedi:

$$v_w = \sum_{j=1}^{m} \alpha_{w,j} A_j + \eta_w \tag{4.2}$$

gdje je najviše k koeficijenata $\alpha_{w,1}, ..., \alpha_{w,m}$ različito od nule, a η_w je vektor šuma. U ovoj jednadžbi i vektori A_j i vektori $\alpha_{w,j}$ su nepoznati što čini problem nekonveksnim. Autori članka su upravo u tome prepoznali rijetko kodiranje (engl. *sparse coding*) koje je moguće riješiti algoritmom k-SVD. Funkcija pogreške u ovom slučaju upravo je l_2-rekonstrukcijska pogreška:

$$\sum_{w} |v_w - \sum_{j=1}^{m} \alpha_{w,j} A_j|_2^2 \tag{4.3}$$

4.1. Algoritam *k-SVD*

Ova optimizacija je surogat za željenu ekspanziju v_{list} jer je moguće smatrati vektore $A_1, ..., A_m$ važnim diskursima u korpusu, koji također možemo nazivati atomima diskursa ili samo atomima. Također, ograničavajući parametar m na broj mnogo manji od broja riječi osigurava da će se isti diskurs koristiti za opis više riječi.

Jednadžba 4.3 može se izraziti kao:

$$\min_{D,X}\{||Y - DX||_F^2\} \quad \text{tako da} \quad \forall i, ||x_i||_0 \leqslant T_0 \tag{4.4}$$

gdje je D matrica atoma, X matrica reprezentacija, a T_0 uvjet rijetkosti. Algoritam *k-SVD* sastoji se od dva koraka. Prvi korak je stupanj rijetkog kodiranja gdje se fiksira D, a prethodni optimizacijski problem svodi se na potragu rijetkih reprezentacija s koeficijentima sumariziranima u matrici X. Ova funkcija cijene može biti ponovo napisana kao:

$$||Y - DX||_F^2 = \sum_{i=1}^{N} ||y_i - Dx_i||_2^2 \tag{4.5}$$

čime je problem rastavljen u N različitih problema u formi:

$$\min_{x_i}\{||y_i - Dx_i||_2^2\} \quad \text{tako da} \quad ||x_i||_0 \leqslant T_0, \quad \text{for } i = 1, 2, ..., N \tag{4.6}$$

Ovaj problem lako je rješiv algoritmima potrage poput *matching pursuit* i *orthogonal matching pursuit* te ukoliko je parametar rijetkosti T_0 dovoljno malen, njihovo rješenje je dobra aproksimacija idealnom kojeg je numerički neisplativ za izračun [1].

Drugi korak je proces izmjene rječnika atoma zajedno s koeficijentima različitim od nule. Uzmimo u obzir da su X i D fiksirani i postavimo u pitanje samo jedan stupac rječnika d_k i koeficijente koji se odnose na njega, tj. k-ti red u X, označen kao x_T^k. Ukoliko se vratimo na funkciju cijene 4.4, u ovom slučaju se može napisati kao:

$$
\begin{aligned}
||Y - DX||_F^2 &= ||Y - \sum_{j=1}^{K} d_j x_T^j||_F^2 \\
&= ||(Y - \sum_{j \neq k} d_j x_T^j) - d_k x_t^k||_F^2 \\
&= ||E_k - d_k x_T^k||_F^2
\end{aligned}
\tag{4.7}
$$

Time je množenje DX svedeno na sumu K matrica ranga 1. Među njima, $K-1$ elemenata su fiksirani, a samo k-ti ostaje upitan. Matrica E_k označava greške svih N uzoraka kada je k-ti atom maknut. Primjenom algoritma SVD u ovom trenutku uzrokovalo bi punjenje vektora X_T^k jer ovakva minimizacija pogreške ne održava uvjet rijetkosti.

Rješenje problema je ipak prilično jednostavno i intuitivno. Definirajmo ω_k kao grupu indeksa koji pokazuju na primjere $\{y_i\}$ koji koriste atom d_k, tj. one gdje je $x_T^k(i)$ različit od nule.

$$
\omega_k = \{i | 1 \leqslant i \leqslant K, x_T^k(i) \neq 0\} \tag{4.8}
$$

Definirajmo s Ω_k matricu dimenzija $N \times |\omega_k|$, gdje su jedinice na mjestima $(\omega_k(i), i)$, a sve ostalo nule. Množenjem $x_R^k = x_T^k \Omega_k$ smanjuje se vektor retka x_T^k micanjem elemenata koji odgovaraju nulama rezultirajući vektorom x_R^k duljine $|\omega_k|$. Slično, množenjem $Y_k^R = Y\Omega_k$ dobiva se matrica veličine $n \times |\omega_k|$ koja uključuje podskup primjera koji trenutno koriste atom d_k. Isti efekt se događa i s $E_k^R = E_k \Omega_k$, što selektira samo one stupce pogreške koji odgovaraju primjerima koji koriste atom d_k.

Iz tog razloga možemo modificirati jednadžbu 4.7 tako da predložimo minimizaciju s obzirom na d_k i x_T^k uz uvjet rijetkosti. To je ekvivalentno minimizaciji izraza:

$$
||E_k \Omega_k - d_k x_T^k \Omega_k||_F^2 = ||E_k^R - d_k x_R^k||_F^2 \tag{4.9}
$$

koji se ovaj put može odrediti direktno pomoću algoritma SVD.

4.2. Filtriranje atoma

Teorija predviđa da bi skup značajnih atoma trebao biti stabilan kroz različite iteracije algoritma k-SVD s obzirom da bi atomi diskursa trebali biti konstantni za neki skup podataka. Njihovi eksperimenti na ponovljenim iteracijama algoritma su to i potvrdili. Atomi dobiveni svakom iteracijom nazvani su bazama. Pokazalo se da $2/3$ atoma

u jednoj bazi ima atome u drugoj bazi. Sličnost je određena skalarnim produktom atoma između baza. Atomi su smatrani duplikatima ukoliko je njihov skalarni produkt bio veći od 0.85. Također, izvršeno je i suprotno filtriranje gdje su odbačeni nestabilni atomi, tj. atomi koji nisu imali susjede u drugim bazama sa skalarnim produktom većim od 0.2. Na taj način finalni rezultat je rezultat nekoliko iteracija k-SVD algoritma, gdje su konačni atomi dobiveni spajanjem baza iz više iteracija.

Našim ispitivanjem na atomima modela *GloVe* pokazalo se da od 10 najzastupljenijih atoma, tj. atoma koje u svojim reprezentacijama koristi najviše riječi, čak njih 7 se odnosi na atome brojeva. Neki od njih odnose se na vrijeme, neki na datum, neki na sportske rezultate i na godine. Prikaz deset najzastupljenijih atoma te pet najsličnijih riječi svakome od njih vidljiv je u tablici 4.1.

Atom	Najsličnije riječi
343	27.11.2011., 22.11.2009., 02.05.2010., 27.03.2011., 18.09.2011.
953	04.09.2008., 07.01.2010., 08.06.2011., 18.01.2011., 02.03.2011.
1615	saopštiti, evro, januar, decembar, oktobar
922	4:2, 5:2, 3:2, 11:9, 4:1
4228	14:07, 20:31, 18:49, 08:37, 16:47
1659	nista, cak, jos, vec, nesto
549	231, 217, 223, 224, 216
3634	00:58, 00:56, 00:53, 00:44, 00:46
563	1907., 1894., 1897., 1906., 1877.
1034	mackenzie, stephen, christopher, davies, jonathan

Tablica 4.1: Deset najpopularnijih atoma i njihove najsličnije riječi

4.3. Atomi

Zbog prethodnih razloga preuzeli smo autorove parametre od 2000 atoma te uvjetom rijetkosti 5. Uvjet rijetkosti znači da je svaku riječ moguće aproksimirati linearnom kombinacijom najviše 5 atoma. Nad svim vektorima riječi izvršit će se 5 iteracija algoritma k-SVD te filtriranje atoma kako bi se dobio konačan skup atoma i reprezentacija.

Za vektore riječi dobivene pomoću modela *SN* te zatim podvrgnutih algoritmu k-SVD izdvojeno je nekoliko dobivenih atoma te 5 najsličnijih riječi tim atomima u

tablici 4.2. Kroz 5 iteracija kreirano je 1809 atoma, što je u istom redu veličine kao i autorovih 2376 atoma za vektore riječi engleskog jezika.

Atom	**225**	**620**	**845**	**1536**	**1714**
	vrlo	model	graditi	narodnjak	samba
	veoma	limuzina	izgraditi	cajka	tango
	izuzetno	kompaktan	sagraditi	treštati	valcer
	iznimno	coupe	gradnja	narodnjački	salsa
	jako	koncept	izgradnja	turbofolk	cha
Oznaka	*prilozi intenziteta*	*vrste automobila*	*izgradnja*	*narodna glazba*	*plesovi*

Tablica 4.2: Prikaz 5 najsličnijih riječi atomima

Iz atoma je moguće vidjeti neku zajedničku značajku svih najsličnijih riječi u atomu te smo na nekoliko primjera i oznakom pokazali njihov zajednički diskurs. Kako je ovim rijetkim kodiranjem optimizirana i matrica reprezentacija riječi, α, tako su i neke višeznačne riječi izražene pomoću linearne kombinacije više atoma. Primjer takve riječi je riječ *križati (se)*. U tablici 4.3 vidljivo je da je riječ sastavljena kao linearna kombinacija četiri atoma koje možemo i smatrati značenjima riječi. Atom 109 pokazuje riječi *križati* u kontekstu prestrašenosti te pravljenja znaka križa na sebi. Atom 194 nije toliko jasan, ali ukazuje na operacije koje se mogu izvršiti nad nizovima podataka. Atom 848 jasno prikazuje pripadnost atomu životinja koji je s riječi *križati* povezan jer se ta riječ može pojaviti u kontekstu parenja jedinki. Posljednji atom, atom 1087 prikazuje nam riječ *križati* u kontekstu biologije. Taj atom jasno pokazuje semantiku područja genetske modifikacije i križanja vrsta.

Atom	*križati*
109	trnac, jeza, obuzimati, peckanje, vrtoglavica
194	poredati, poslagati, abecedni, posložiti, kronološki
848	svinja, govedo, krava, perad, stoka
1087	modificirati, genetički, genetski, gmo, modifikacija

Tablica 4.3: Atomi koji čine riječ *križati*

4.4. Zadatak određivanja značenja riječi

Krenuvši od prethodnih tvrdnji kako je vektor riječi linearna kombinacija značenja te riječi, ispitana je metoda k-SVD na skupu podataka *CRO36WSD*. Skup se sastoji od 36 višeznačnih riječi te 3501 pitanja koja se sastoje od rečenice, riječi za koju se određuje značenje te oznaka značenja [2]. Struktura ispitnog skupa i nekoliko primjera prikazani su u tablici 4.4. Uz pitanja dostupna je i zbirka značenja koja za svako značenje riječi navodi definiciju te primjere upotrebe. Zbirka značenja za riječ *aktivan* vidljiva je u tablici 4.5.

Riječ	**Lema**	**Rečenica**	**Značenje**
aktivnih	aktivan	Klub ima oko stotinu aktivnih članova.	A.001.01
gorjelo	gorjeti	Noćas je gorjelo nedaleko od Dubrovnika	V.002.01
normalna	normalan	Istina Martine, samo što mi nismo normalna država.	A.004.02

Tablica 4.4: Ispitni primjeri iz skupa *CRO36WSD*

Značenje	**Definicija**	**Upotreba**
A.001.01	a. koji djeluje b. koji je u radnom odnosu, opr. umirovljen, v. umiroviti	Ivan je fizički najaktivnija osoba koju znam. Ivan se nalazi na listi aktivnih zaposlenika. On više nije aktivan pilot.
A.001.02	koji ima inicijativu, koji je radišan; poduzetan, angažiran opr. pasivan	On je zaista aktivan čovjek, stalno nešto popravlja. On je u zadnje vrijeme vrlo aktivan, prijavio je šest projekata.
A.001.03	koji djeluje, koji još postoji ili djeluje u svom osnovnom svojstvu (o prirodnim pojavama) [aktivan vulkan]	Vulkan Etna je opet bio aktivan ovo ljeto. Aktivne tvari u ovom preparatu uništavaju sve klice.
A.001.04	ekon. kojem su potraživanja veća od dugova, opr. u dugu, u pasivi, pasivan	U toj propaloj banci provodila su se neadekvatna pozicioniranja aktivnih sredstava.
A.001.05	koji je uključen, u pogonu	Antivirusna zaštita uvijek bi trebala biti aktivna.

Tablica 4.5: Zbirka značenja iz skupa *CRO36WSD* za riječ *aktivan*

Jednostavno ispitivanje se izvodi u tri koraka. Prvi korak je pridjeljivanje vektora svakom značenju riječi. S obzirom da se uz svako značenje nalazi i barem jedna rečenica upotrebe, definiramo vektor značenja riječi v_{sense} kao:

$$v_{sense} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{M} \sum_{j=1}^{M} v_{w_{i,j}} \tag{4.10}$$

gdje je N broj rečenica koje su navedene za to značenje, M_i broj riječi u i-toj rečenici, a $v_{w_{i,j}}$ vektor riječi na poziciji j u i-toj rečenici.

Drugi korak je spajanje vektora značenja s atomima. Za svaki atom od kojeg se sastoji upitna riječ izračuna se kosinusna sličnost s vektorom značenja v_{sense} svih mogućih značenja te se atomu pridruži ono značenje koje donosi najveću sličnost

Treći korak je pretvaranje rečenice pitanja u vektor na isti način kao i vektore značenja, ali u ovom slučaju imamo samo jednu rečenicu po pitanju. Vektor rečenice izrazimo kao prosjek vektora riječi od kojih se sastoji:

$$v_{sent} = \frac{1}{M} \sum_{i=1}^{M} v_{w_i} \tag{4.11}$$

Rješenje dobivamo izračunom maksimalne kosinusne sličnosti između vektora rečenice v_{sent} te mogućih atoma od kojih se sastoji riječ čije značenje tražimo. Ukoliko je upravo onaj atom koji u drugom koraku odgovara značenju riječ koje je zapisano u pitanju, smatramo rješenje točnim.

U tablici 4.6 nalaze se rezultati različitih modela za reprezentaciju riječi na skupu *CRO36WSD*. U obzir su uzeta samo pitanja za koje je pomoću k-SVD algoritma dobivena riječ koja se sastoji od barem 2 značenja te se time ograničavamo na polisemične riječi čiji utjecaj možemo mjeriti.

	CBoW	**Skip-gram**	**FT CBoW**	**FT Skip-gram**	**GloVe**	**SN**
Broj pitanja	2464	2787	2583	2955	2317	1779
Točnost	0.2853	0.3348	0.2462	0.3648	0.2175	0.3333
Broj atoma	4072	4063	3887	3776	4398	1809

Tablica 4.6: Rezultati modela na skupu *CRO36WSD*

Iz rezultata je vidljivo kako su obje varijante modela *skip-gram* pokazale najbolje rezultate dok ih slijedi model *SN* sa značajno manjim brojem atoma. Model *GloVe* pokazao je značajno lošije rezultate od svih ostalih modela čak i uz uporabu najvećeg broja atoma za reprezentaciju vektora.

Rezultati na svim ispitivanjima pokazali su da varijante modela *skip-gram* te model *GloVe* daju najbolje rezultate iako se model *GloVe* nije pokazao pogodan za određivanje značenja riječi. Za modele koji uče predikcijom pokazalo se da točnost odabira sinonima raste suprotno veličini prozora. Iako se pokazalo da su vektori većih dimenzija u nekim slučajevima imali bolje rezultate, pokazalo se da vektori dimenzije 250 ne odstupaju drastično rezultatima te čak i u nekim slučajevima pobjeđuju veće vektore. Gubitak jedne dimenzije tokom postprocesiranja nije donio nikakav napredak, a gubitkom više dimenzija gubi se sličnost riječi zbog čega je metoda neuspješna.

5. Zaključak

U ovom radu bavili smo se ispitivanjem vektorskih reprezentacija riječi hrvatskog jezika. Ispitano je šest različitihi modela čije su vrijednosti ispitane na zadacima odabira sinonima, analogijskih pitanja te određivanja značenja riječi.

Pokazalo se da su svi modeli pogodni za učenje vektorskih reprezentacija riječi hrvatskog jezika u nekoj mjeri. Ispitivanjem modela pomoću traženja sličnih riječi nasumičnim riječima vidljivo je da je svaki model kao rezultat dao vektore gdje riječi sličnog značenja daju slične vektore.

U daljnjim ispitivanjima pokazalo se da veći skup primjera za učenje pozitivno utječe na rezultate odabira sinonima. Razlog tome mogao bi jednostavno biti veći broj konteksta u kojima se riječ ponavlja te time i preciznije određivanje značenja riječi. S druge strane, učenjem na manjem, ali puno strukturiranijem skupu podataka pokazalo se boljim za analogijska pitanja. Najbolje rješenje za učenje reprezentacija bi ipak bio neka mješavina ova dva skupa podataka koja bi bila strukturirana, pisana pravilnim hrvatskim jezikom te sadržavala dovoljan broj primjera za učenje.

Iako se s porastom broja dimenzija vektora riječi povećava reprezentacijska moć tih vektora, rezultati su pokazali da se najbolja svojstva postižu za vektorske reprezentacije umjerenih dimenzija. Veličina prozora se pokazala da nema drastičan učinak na kvalitetu modela, ali je ipak vidljivo kako *GloVe* te *SN* uvijek daju bolje rezultate s većim prozorom.

Postprocesiranje vektora, iako rezultatski zanimljivo na skupu *hr-synonym-choice*, nije donijelo dobre rezultate na drugim ispitivanjima. Pokazalo se da eliminacijom glavnih smjerova u vektorskom prostoru gubimo bitne informacije o riječi te se lako izgubi i relacija da slične riječi imaju i slične vektore.

Pomoću algoritma k-SVD moguće je iz vektora riječi izraziti zanimljive atome diskursa. Iako je potrebno dosta filtriranja kako bi se izbacili duplicirani atomi ili atomi koji nemaju konkretno značenje, ostatku je moguće dodijeliti neki semantički kontekst. Iako se atomi diskursa nisu pokazali posebno dobrima za zadatak odabira značenja riječi, moramo uzeti u obzir i jednostavnost korištenog modela.

U budućem radu bilo bi dobro ispitati kombinaciju oba korpusa za učenje vektorskih reprezentacija te pronaći još neke zadatke ili druge mjere za evaluaciju vektora riječi. Također, atomi diskursa pružaju još jedan smjer istraživanja. Ispitivanje različitih parametara poput broja atoma te maksimalnog broja reprezentacija moglo bi donijeti bolje rezultate. Na zadatku odabira značenja riječi vrijedilo bi također isprobati metode strojnog učenja koje bi vjerojatno postigle bolji rezultat.

LITERATURA

[1] Michal Aharon, Michael Elad, i Alfred Bruckstein. rmk-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 54(11):4311–4322, 2006.

[2] Domagoj Alagić i Jan Šnajder. Cro36wsd: A lexical sample for croatian word sense disambiguation. U *Proceedings of the 10th edition of the Language Resources and Evaluation Conference, LREC 2016*, Portorož, Slovenia, 2016. ELRA.

[3] Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, i Andrej Risteski. Random walks on context spaces: Towards an explanation of the mysteries of semantic word embeddings. *CoRR*, abs/1502.03520, 2015. URL `http://arxiv.org/abs/1502.03520`.

[4] Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, i Andrej Risteski. Linear algebraic structure of word senses, with applications to polysemy. *CoRR*, abs/1601.03764, 2016. URL `http://arxiv.org/abs/1601.03764`.

[5] Glenn De Backer. Word2vec tutorial - the skip-gram model, 2015. URL `https://www.simplicity.be/article/throwing-dices-recognizing-west-flemish-and-other-languages/`.

[6] Piotr Bojanowski, Edouard Grave, Armand Joulin, i Tomas Mikolov. Enriching word vectors with subword information. *CoRR*, abs/1607.04606, 2016. URL `http://arxiv.org/abs/1607.04606`.

[7] Susan Feldman. Nlp meets the jabberwocky. *Online*, 23(3):62–72, 1999.

[8] Google. Vector representations of words, 2016. URL `https://www.tensorflow.org/tutorials/word2vec`.

[9] Gabriella Lapesa i Stefan Evert. A large scale evaluation of distributional semantic models: Parameters, interactions and model selection. *Transactions of the Association for Computational Linguistics*, 2:531–545, 2014.

[10] Elizabeth D Liddy. Enhanced text retrieval using natural language processing. *Bulletin of the American Society for Information Science and Technology*, 24(4): 14–16, 1998.

[11] Nikola Ljubešić i Filip Klubička. {bs,hr,sr}WaC – web corpora of Bosnian, Croatian and Serbian. U *Proceedings of the 9th Web as Corpus Workshop (WaC-9)*, stranice 29–35, Gothenburg, Sweden, 2014. Association for Computational Linguistics.

[12] Nikola Ljubešić, Filip Klubička, Željko Agić, i Ivo-Pavao Jazbec. New inflectional lexicons and training corpora for improved morphosyntactic annotation of croatian and serbian. U Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk, i Stelios Piperidis, urednici, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, may 2016. European Language Resources Association (ELRA). ISBN 978-2-9517408-9-1.

[13] Chris McCormick. Word2vec tutorial - the skip-gram model, 2016. URL `http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/`.

[14] Tomas Mikolov, Kai Chen, Greg Corrado, i Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013. URL `http://arxiv.org/abs/1301.3781`.

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, i Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013. URL `http://arxiv.org/abs/1310.4546`.

[16] Jiaqi Mu, Suma Bhat, i Pramod Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. *CoRR*, abs/1702.01417, 2017. URL `http://arxiv.org/abs/1702.01417`.

[17] Jeffrey Pennington, Richard Socher, i Christopher D. Manning. Glove: Global vectors for word representation. U *Empirical Methods in Natural Language Pro-*

cessing (EMNLP), stranice 1532–1543, 2014. URL http://www.aclweb.org/anthology/D14-1162.

[18] Jan Šnajder, Sebastian Padó, i Željko Agić. Building and evaluating a distributional memory for croatian. U *51st Annual Meeting of the Association for Computational Linguistics*, stranica in press, 2013.

[19] Leo Zuanović, Mladen Karan, i Jan Šnajder. Experiments with neural word embeddings for croatian. U *TODO*, stranica TODO. TODO, 2014.

Ispitivanje vektorskih reprezentacija riječi hrvatskoga jezika

Sažetak

U ovom radu ispitani su različiti modeli za vektorsku reprezentaciju riječi na više skupova za učenje na hrvatskom jeziku. Dobiveni vektori evaluirani su na dva ispitna skupa odabira sinonima te analogija. Također ispitan je i model ekstrakcije značenja iz vektora pomoću algoritma k-SVD te njegova primjena na određivanje značenja riječi.

Ključne riječi: vektori riječi hrvatskog jezika, vektorske reprezentacije riječi, atom diskursa, polisemija

Evaluating Croatian Language Word Representations

Abstract

In this paper different models for word representations have been tested on multiple training corpora for Croatian language. Result vectors have been evaluated on two different test sets which consisted of synonym choices and analogies. Another task was to extract meanings from the vectors through k-SVD algorithm and it's application to word sense disambiguation.

Keywords: croatian word vectors, word representations, discourse atom, polisemy