Jezični model hrvatske Wikipedije temeljen na neuronskim mrežama

Illich, Luka

Undergraduate thesis / Završni rad

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Rijeka, Faculty of Engineering / Sveučilište u Rijeci, Tehnički fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:190:431360

Rights / Prava: Attribution 4.0 International/Imenovanje 4.0 međunarodna

Download date / Datum preuzimanja: **2024-07-19**

Repository / Repozitorij:

Repository of the University of Rijeka, Faculty of Engineering

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
Prijediplomski studij računarstva

Završni rad

Jezični model hrvatske Wikipedije temeljen na neuronskim mrežama

Rijeka, rujan 2023.

Luka Illich
0069087894

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
Prijediplomski studij računarstva

Završni rad

Jezični model hrvatske Wikipedije temeljen na neuronskim mrežama

Mentor: prof. dr. sc. Ivo Ipšić

Rijeka, rujan 2023.

Luka Illich
0069087894

Umjesto ove stranice umetnuti zadatak za završni ili diplomski rad

Izjava o samostalnoj izradi rada

Izjavljujem da sam samostalno izradio ovaj rad.

Rijeka, rujan 2023.

Luka Illich

Zahvala

Zahvaljujem se mentoru prof. dr. sc. Ivi Ipšiću na pomoći i uputama tijekom pisanja ovoga rada. Zahvaljujem i svojoj obitelji i najbližima na podršci tijekom studiranja.

Sadržaj

Popis slika

Popis tablica

Poglavlje 1

Uvod

Obrada prirodnog jezika (*Natural Language Processing - NLP*) je grana računarstva čiji je cilj razumijeti, interpretirati i generirati ljudski jezik korištenjem raznih algoritama. Ključan koncept u obradi prirodnog jezika je jezično modeliranje koje omogućava računalima da razumiju i generiraju tekst na način sličan ljudskom jeziku. Jedan od značajnijih alata vezanih uz jezično modeliranje je *Word2Vec* razvijen u sklopu Googleovog istraživačkog tima. Alat koristi neuronske mreže kako bi stvorio jezični model - vektorski prikaz riječi i njihove međusobne semantičke veze. Primjenu takvih modela pronalazimo u raznim situacijama kao što su: prevođenje tekstova, razumijevanje upita, prepoznavanje govora, generiranje tekstova, razumijevanje sentimenata i automatsko ispravljanje teksta.

Cilj ovog završnog rada bio je istražiti i opisati gradnju jezičnog modela korištenjem spomenutog alata, primijeniti ga u stvaranju jezičnog modela te prikazati rezultate dobivene korištenjem riječi hrvatskog jezika. Nadam se da će čitatelji kroz ovaj rad steći dublje razumijevanje jezičnog modeliranja te prepoznati važnost jezičnih modela u širem kontekstu umjetne inteligencije.

1.1 Zadatak

Zadatak završnog rada bio je prikazati postupak gradnje jezičnog modela na temelju tekstova hrvatske Wikipedije uz pomoć alata *Word2Vec*. Pomoću dobivenog modela

bilo je potrebno prikazati vjerojatnost pojavljivanja nizova riječi.

Poglavlje 2

Jezični model

Strukture i vrste jezičnih modela će u ovom poglavlju biti objašnjeni samo do razine razumijevanja konteksta potrebnog za daljnje izlaganje ovog rada, a bilo kakvo detaljnije razmatranje problematike jezičnih modela moguće je pronaći u knjizi pod nazivom "Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition"[1].

U području obrade prirodnog jezika jezični model osnovni je koncept za razumijevanje jezičnih struktura i uzoraka. Osnovna zadaća mu je predvidjeti iduću riječ koristeći vjerojatnosti pojavljivanja nizova riječi.

Jezični modeli se treniraju na velikim korpusima tekstova kako bi "naučili" značenje i odnose među riječima. Analizom riječi u rečenici ili djelu rečenice model računa koja je vjerojatnost da se nakon niza riječi pojavi određena riječ. Za računanje takve vjerojatnosti, odnosno uvjetne vjerojatnosti, koristi se formula:

$$\begin{aligned} P(w_{1:n}) &= P(w_1)P(w_2|w_1)P(w_3|w_{1:2})...P(w_n|w_{1:n-1}) \\ &= \prod_{k=1}^{n} P(w_k|w_{1:k-1}) \end{aligned} \tag{2.1}$$

gdje w označava riječ, $P(w)$ vjerojatnost pojavljivanja riječi, a vrijedi da je:

- w_n - riječ na n-tom mjestu u nizu

- $w_{m:n}$ - niz od m-te do n-te riječi
- $P(w_n)$ - vjerojatnost pojavljivanja n-te riječi
- $P(w_n|w_{1:n-1})$ - nakon pojavljivanja niza riječi $w_{1:n-1}$, vjerojatnost pojavljivanja n-te riječi.

Nedostatak ovakvom računanju nalazimo kod dugih nizova riječi. Što je niz riječi dulji, to je veća mogućnost njegovog nepostojanja u korpusu prema kojem gradimo jezični model i tada računanje vjerojatnosti za takav niz nije moguće. Rješenje ovog problema možemo naći u implementacijama raznih drugih modela od kojih su neki ukratko objašnjeni u idućim podnaslovima.

2.1 N-gram modeli

Ovakvi jednostavni jezični modeli temelje se na pretpostavci da se iduća riječ niza može predvidjeti uzimajući u obzir vrlo mali broj prethodnih riječi - Markovljev lanac prilagođen jezičnom modelu. Takvi modeli aproksimiraju vjerojatnost pojavljivanja N-te riječi s određivanjem vjerojatnosti pojavljivanja $N-1$ prethodnih riječi. Tada se koristi formula:

$$P(w_n|w_{1:n-1}) \approx P(w_n|w_{n-N+1:n-1}) \tag{2.2}$$

dok za 2-gram model formula glasi:

$$P(w_2|w_{1:2}) \approx P(w_2|w_1). \tag{2.3}$$

Najčešće se koriste 2-gram (tzv. bigram) i 3-gram (tzv. trigram) modeli. Primjerice u 3-gram modelu se procjenjuje pojavljivanje treće riječi na temelju prethodne dvije.

2.2 Rekurzivni ili neuronski jezični modeli

Rekurzivni jezični modeli su vrsta modela koji koriste rekurzivne neuronske mreže kako bi obradili jezične sekvence. Ovi modeli oslanjaju se na povratne veze kako bi naučili složenije veze među riječima i njihov kontekst u rečenicama i tekstu. Takva vrsta modela omogućuje bolje razumijevanje strukture jezika, ovisnosti između riječi i konteksta u kojem se koriste. Neke od podvrsta takvih jezičnih modela koriste:

- **Rekurzivne neuronske mreže** (*Recurrent neural network - RNN*): svaka riječ niza (sekvence) koristi se kao ulaz u RNN, rekurzivnim vezama informacije se prenose među riječima te se "stanje mreže" ažurira i propagira prema naprijed kroz sekvencu
- **Dugačke kratkoročne memorije** (*Long short-term memory - LSTM*): koriste se posebne strukture kako se ne bi "zaboravile" bitne informacije sadržane u ranijim dijelovima niza.

Rekurzivni jezični modeli mogu "zapamtiti" dugoročne ovisnosti riječi u tekstu i stoga imaju bolje razumijevanje konteksta u kojem se riječi koriste. Iako preciznije modeliraju gramatičke strukture i složenije jezične uzorke, rekurzivni modeli imaju i svoje nedostatke:

- mogu biti **računalno i vremenski zahtjevni**
- imaju problema s **nestajućim ili eksplodirajućim gradijenatima** (određenim riječima daju premalo ili previše pažnje).

2.3 Transformatorski jezični model

Osnovni koncept transformatorskih modela je *self-attention*, "pažnja prema sebi", mehanizam koji omogućuje modelu da izračuna važnost svake riječi u odnosu na sve druge riječi u sekvenci. Prilikom izračuna težine nisu međusobno ovisne tako da se mogu određivati istovremeno za više riječi. Također, težine odražavaju koliko je svaka riječ važna za razumijevanje konteksta ostalih riječi. Dodatno, transformator-

ski jezični modeli koriste *Multi-Head Attention* ("Višeglava pažnja") gdje umjesto jedne "pažnje", transformatori koriste više "pažnji", nazvanih "glave". Svaka glava "uči" različite težine i ovisnosti između riječi. Kombiniranjem rezultata pojedinih težina dobiva se konačni reprezentativni vektor.

Neke od dodatnih tehnika koje se još koriste u transformatorskim jezičnim modelima su enkodiranje i dekodiranje te rezidualno povezane neuronske mreže.

Prednosti transformatorskih jezičnih modela u odnosu na ostale jezične modele su:

- mogućnost **paralelne obrade riječi** (*self-attention*)
- uspostavljanje **dalekosežne ovisnosti** među riječima u sekvenci
- dobro **skaliranje** prema količini podataka i dubini mreže.

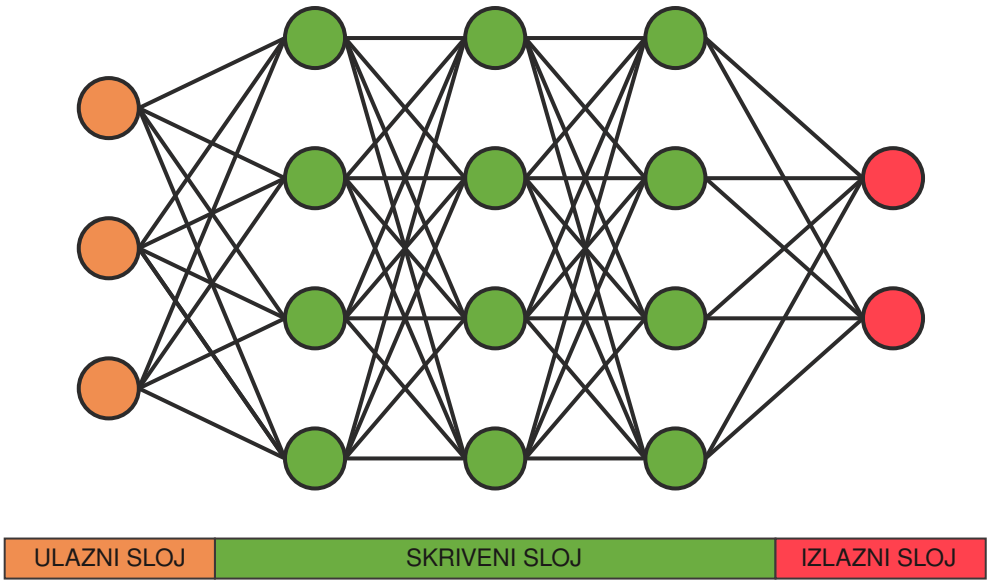
Transformatorski jezični modeli poput BERT-a, BART-a, ChatGPT-a i drugih trenutno su vodeći modeli u području obrade prirodnog jezika. Ostvarivši značajne rezultate u širokom rasponu jezičnih zadataka vrlo dobro prezentiraju njihovu svestranost i sposobnost razumijevanja konteksta u koji ih postavimo.

Poglavlje 3

Neuronske mreže

Neuronska mreža je struktura koja za svrhu ima obradu podataka. Glavna zadaća joj je da na temelju više ulaznih i izlaznih parametara modificira svoje "ponašanje" kako bi bila pogodna za daljnju uporabu, odnosno da bi nakon treniranja mogla za dane ulazne parametre ponuditi točan podatak.

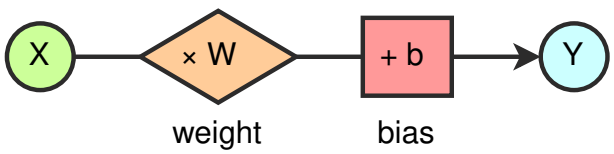
Kako i vidimo na slici 3.1, neuronska mreža se sastoji od jednog ili više skrivenih slojeva koji se nalaze između ulaznog i izlaznog sloja. Svaki čvor ulaznog sloja je obično povezan na sve čvorove prvog skriveng sloja, dok su svi čvorovi zadnjeg skrivenog sloja obično povezani na sve čvorove izlaznog sloja. Također, skriveni slojevi imaju vezu svakog svojeg čvora sa svim čvorovima susjednih skrivenih slojeva.



Slika 3.1 Shema neuronske mreže

3.1 Veze među čvorovima

Svaka moguća veza dva čvora opisana je dvama vrijednostima: *weight* - težina i *bias* - sklonost. Kako svaki čvor prosljeđuje svoju vrijednost idućem, *weight* će pomnožiti vrijednost prijašnjeg čvora svojim iznosom, dok će *bias* dodati svoj iznos vrijednosti koja kroz nj prolazi, što je i prikazano na slici 3.2.



Slika 3.2 Shema veze između dva čvora neuronske mreže te vrijednosti *weight* i *bias* među njima

Za primjer možemo uzeti dva čvora X i Y te vezu koja je određena vrijednostima *weight* $w_1 = 5$ i *bias* $b_1 = -3$. Ako na čvor X postavimo vrijednost 4, čvor Y će poprimiti vrijednost 17, po izračunu: $4 \cdot 5 - 3 = 17$.

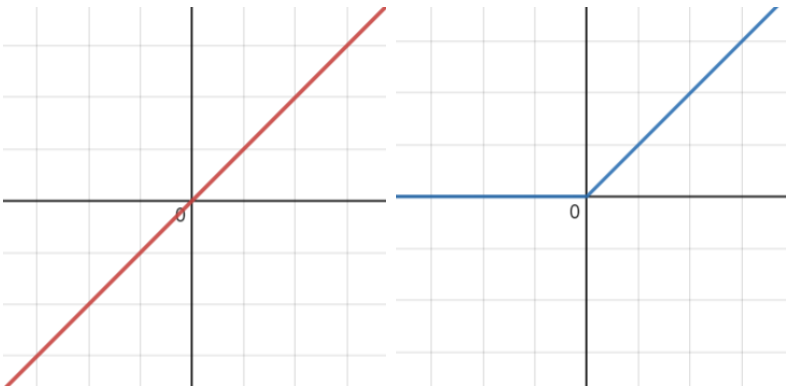
3.2 Čvorovi skrivenih slojeva

Skriveni slojevi sastoje se od čvorova u kojima se nalaze aktivacijske funkcije. Njihova je uloga od prijašnjeg čvora primiti vrijednost te je pomoću unaprijed određene funkcije pretvoriti u novu vrijednost. U slučaju da čvor skrivenog sloja prima vrijednosti više prethodnih čvorova uobičajeno je koristi sumiranje te potom sumu proslijediti aktivacijskoj funkciji.

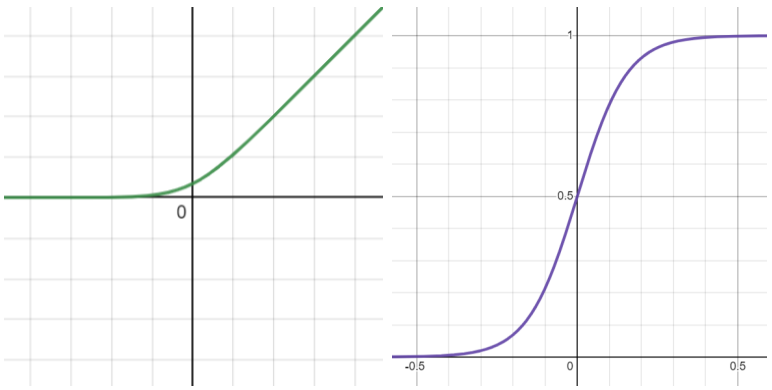
Neke od najčešćih aktivacijskih funkcija su:

- **linearna**
- **ReLU - rectified linear unit** (ispravljena linearna jedinica)
- **SoftPlus**
- **sigmoidna**,

a njihov grafički prikaz vidljiv je na slici 3.3.



(a) Graf linearne funkcije (b) Graf $ReLU$ funkcije



(c) Graf *SoftPlus* funkcije (d) Graf sigmoidne funkcije

Slika 3.3 Grafički prikaz nekih od najčešćih aktivacijskih funkcija neuronske mreže

3.3 Čvorovi izlaznog sloja

Zbog karakteristike neuronskih mreža da za ulazne parametre mogu ponuditi više mogućih rješenja, često se nakon samih izlaznih čvorova koriste funkcije za odabir jednog od više ponuđenih rješenja. Dvije najčešće funkcije su: *ArgMax* i *SoftMax*. *ArgMax* je jednostavna funkcija koja od više argumenata vraća najveću vrijednost. Njoj različita, *SoftMax* je funkcija koja normalizira sve vrijednosti izlaznog sloja na način da njihov zbroj iznosi 1 te tako zapravo dobivamo distribuciju vjerojatnosti. Nju opisujemo izrazom:

$$SoftMax(x_i) = \frac{e^{x_i}}{\sum_{i=1}^{n} e^{x_i}} \tag{3.1}$$

gdje je:

- e - Eulerov broj, konstanta, $e \approx 2.71828182...$
- x_i - i-ta vrijednost u izlaznom sloju
- n - broj izlaznih vrijednosti u izlaznom sloju.

Kako je za računanje *SoftMax* vrijednosti N rješenja potrebno provesti isto toliko operacija, $O(N)$, često se koristi matematički složenija, ali računalno efikasnija funkcija za izračun normaliziranih vrijednosti izlaznog sloja pod nazivom *Hierarchial SoftMax*. Njezina računalna kompleksnost se opisuje notacijom $O(log_2 N)$, a bitna je činjenica da se izlazni čvorovi prikazuju kao listovi stabla u kojem se ne računaju vjerojatnosti svih grana nego samo onih potrebnih za daljnju obradu.

3.4 *Backpropagation*

Ključan algoritam za optimiziranje parametara modela neuronske mreže je *Backpropagation*. Ovaj algoritam omogućuje neuronskoj mreži da "uči" prilagođavanjem svojih težinskih vrijednosti (*weight* i *bias*) kako bi se minimizirala greška između predviđenih i ciljnih vrijednosti modela. Sastoji se od sljedećih koraka:

1. ***Forward Pass* - Prolaz naprijed**: Ulazni podaci se propagiraju kroz mrežu te generiraju predviđanja na način da svaki neuron u mreži izračunava sumu svojih ulaza, primjenjuje aktivacijsku funkciju na tu sumu i prosljeđuje rezultat sljedećem sloju.

2. ***Loss Calculation* - Izračun greške**: Uspoređuju se predviđanja neuronske mreže sa stvarnim ciljnim vrijednostima, odnosno, prikazuje se greška pomoću nekih od funkcija (najčešće srednja kvadratna pogreška za regresiju ili kategorizacijska križna entropija za klasifikaciju).

3. ***Backward Pass (Backpropagation)* - Prolaz unatrag**: koristi se metoda

gradijentalnog spusta kako bi se prilagodili parametri neuronske mreže te se minimizirala greška. Gradijent funkcije gubitka po težinama se računa unatrag, počevši od zadnjeg sloja prema prvom sloju.

4. ***Weight Update* - Ažuriranje težina**: Nakon što se gradijenti izračunaju za sve težine mreže, težine se ažuriraju koristeći prilagođeni gradijentni spust. Obično se koristi metoda kao što je stohastički gradijentni spust ili njegove varijacije.

Svaki od ovih koraka se ponavlja u više iteracija kako bi se dobio što bolji model neuronske mreže.

Poglavlje 4

Alat *Word2vec*

Word2vec[2][3] je algoritam koji korištenjem neuronske mreže prikazuje riječi pomoću niza brojeva (vektora). Objavljen je 2013. godine u radovima "Efficient Estimation of Word Representations in Vector Space"[4] i "Distributed representations of words and phrases and their compositionality"[5], a nastao je u Google-u pod vodstvom Tomasa Mikolova, češkog znanstvenika iz područja računarstva. Algoritam radi na način da na temelju dovoljno velike količine teksta može prepoznati i zapamtiti semantičke odnose među riječima, definicije i kontekst prema tekstu u kojem se nalaze. Poveznice na implementacije u raznim programskim jezicima dostupne su u Wikipedijinom članku *Word2vec*[6], a vrlo dobro objašnjenje rada algoritma može se pročitati i na web-stranici `https://towardsdatascience.com/` u članku "Word2Vec Explained"[7].

Word2vec, u svojoj uobičajenoj verziji, koristi plitku neuronsku mrežu (*Shallow Neural Networks*[1]) s jednim skrivenim slojem u kojem se odvija glavna zadaća samog algoritma - tokenizacija riječi te pamćenje semantičkih i drugi veza među riječima.

4.1 Tokenizacija riječi

Kako računala u svojoj srži sve procese i podatke prikazuju u logičkom ili brojčanom obliku, potrebno je, za obradu i modeliranje, riječi prikazati brojevima. Svakoj

[1] *Shallow Neural Networks* - sadrže do 2 skrivena sloja

riječi koja ulazi u obradu algoritma dodijeljen je ulazni čvor u neuronskoj mreži, a svaki taj čvor "spojen" je na sve čvorove u skrivenom sloju. Broj čvorova u skrivenom sloju određuje broj dimenzija kojima želimo prikazati pojedinu riječ kao vektor dok je broj izlaznih čvorova isti kao i broj ulaznih. Čvorovi skrivenog sloja sastoje se od aktivacijskih funkcija za linearno transformiranje kojima prethode sume umnožaka težina i ulaznih parametara. Same težine na aktivacijskim funkcijama jesu vrijednosti koje određuju riječ kao vektor, odnosno svaka riječ za ulaz u svaku od aktivacijskih funkcija ima težinu koja je jedna od dimenzija vektora. Čvorovi izlaznog sloja sastoje se od suma umnožaka težine i izlaza iz aktivacijske funkcije. U samom izlaznom sloju vrijednosti prolaze obradu *SoftMax* funkcijom te poprimaju konačne vrijednosti. *SoftMax* funkcija:

$$SoftMax(x_i) = \frac{e^{x_i}}{\sum_{i=1}^{n} e^{x_i}} \tag{4.1}$$

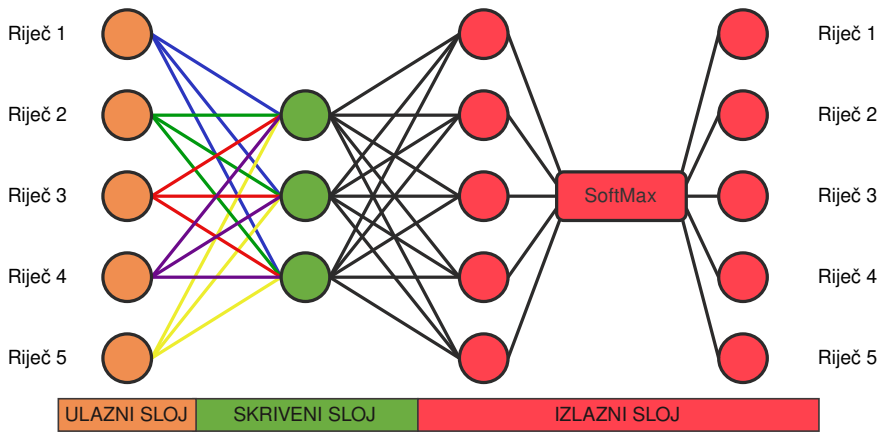
gdje je:

- e - Eulerov broj, konstanta, $e \approx 2.71828182...$
- x_i - i-ta vrijednost u izlaznom sloju
- n - broj izlaznih vrijednosti u izlaznom sloju.

Za primjer možemo uzeti tekst koji se sastoji od 5 različitih riječi koje želimo prikazati 3-dimenzionalnim vektorima. Neuronska mreža će imati po 5 ulaznih i izlaznih čvorova te 3 čvora s aktivacijskim funkcijama u skrivenom sloju. Svaka riječ ima točno 3 veze prema skrivenom sloju na kojima se nalazi po jedna težina (*weight*). Upravo te 3 težine određuju vektor kojim će riječ biti prikazana. Nakon skrivenog sloja vrijednosti sve 3 aktivacijske funkcije su usmjerene u svaki od 5 izlaznih čvorova koji vrijednosti množe sa svojim težinama i sumiraju ih u jedan broj. Shematski prikaz primjera vidljiv je na slici 4.1 gdje su različitim bojama prikazane veze svake od riječi u ulaznom sloju sa čvorovima skrivenog sloja. Svaka boja označava vektor (skup *weight* vrijednosti) koje definiraju riječ u obradi alatom *Word2vec*. Formula *Softmax* fukncije druge izlazne vrijednosti u ovom primjeru bila bi:

$$SoftMax(x_2) = \frac{e^{x_2}}{\sum_{i=1}^{3} e^{x_i}} = \frac{e^{x_2}}{e^{x_1} + e^{x_2} + e^{x_3}}. \tag{4.2}$$

Softmax funkcija dobivene rezultate za pojedinu riječ normalizira na način da oni predstavljaju distribuciju vjerojatnosti tako možemo bilo koje realne brojeve pretvoriti u vrijednosti čiji će ukupni zbroj biti 1.



Slika 4.1 Shema neuronske mreže *Word2vec* modela s riječima i prikazom težina

4.2 Odnosi među riječima

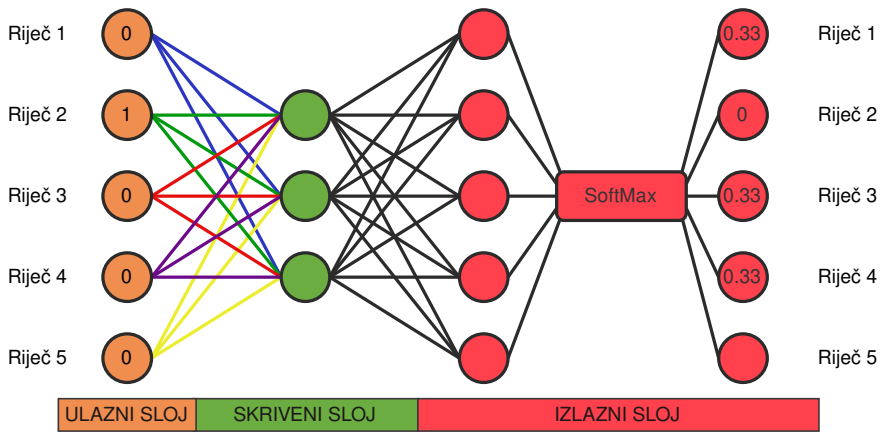
Kako se sve težine unutar neuronske mreže u prvoj fazi određuju slučajnim odabirom, a želimo da te težine određuju riječi na takav način da se slične riječi nalaze blizu u vektorskom prostoru, potrebno je korištenjem procesa *back propagation* optimizirati težine čime će se i vektori grupirati. U slučaju algoritma *Word2Vec* korištenjem algoritma postupnog opadanja (*Gradient Descent*) ažuriramo težine veza između čvorova čime i vrijednosti vektora postaju optimalne.

4.3 Implementacija

Prilikom samog treniranja neuronske mreže koristi se jedna od ove dvije strategije: *Skip-gram* i *Continuous Bag Of Words* (skraćeno: *CBOW*).

4.3.1 *Skip-gram*

Korištenje ove metode podrazumijeva da na temelju određene riječi želimo dobiti okolne riječi. Tada za ulazni parametar postavljamo vrijednost 1 za riječ prema kojoj želimo odrediti ostale riječi, dok u izlaznom sloju postavljamo vrijednost svim riječima koje želimo da budu povezane kao okolne. Vrijednost koja se postavlja u izlaznom sloju je $\frac{1}{N}$ gdje je N broj riječi koje predviđamo (slika 4.2). Tako bi za 4 riječi koje želimo predvidjeti, u izlaznom sloju postavili svakoj vrijednost 0.25.

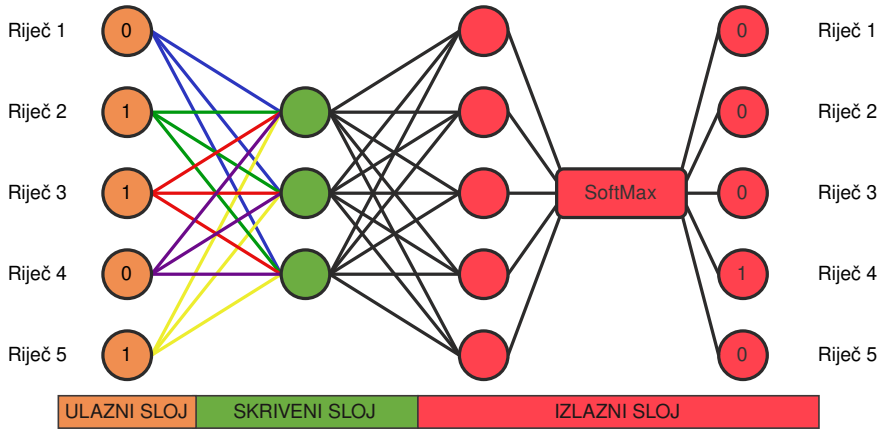


Slika 4.2 Shema neuronske mreže *Word2vec* prilikom korištenja *skip-gram* modela

4.3.2 *Continuous Bag Of Words*

U slučaju korištenja ove strategije pri treniranju neuronske mreže pokušavamo predvidjeti središnju riječ na temelju riječi koje ju okružuju. Na ulaz neuronske mreže postavljamo vrijednost 1 za sve riječi koje se nalaze prije i poslije riječi koju želimo

da neuronska mreža "pogodi", dok na izlaz postavljamo vrijednost 1 za riječ koju želimo dobiti (slika 4.3).



Slika 4.3 Shema neuronske mreže *Word2vec* prilikom korištenja *CBOW* modela

4.4 Zaključak algoritma

Pomoću predstavljene tokenizacije, *Word2vec* je u stanju, u svom vektorskom prostoru, prikazati i grupirati po značenju slične riječi. U konkretnom primjeru korištenja hrvatskih riječi, uz grupiranje sinonima i sličnih riječi, možemo očekivati grupiranje riječi s istim korijenom, primjerice imenice u različitim padežima ili pridjeve prilagođene različitim rodovima.

Poglavlje 5

Ostali korišteni alati

5.1 Hrvatska Wikipedija

Wikipedia[8] je internetska enciklopedija koja omogućava korisnicima da slobodno uređuju i dopunjuju članke na različitim jezicima. Osnovana 2001. godine ubrzo je postala jedan od najvažnijih izvora informacija na internetu, a danas ima preko 15 milijuna članaka. Wikipedija se temelji na načelu suradničkog uređivanja gdje korisnici diljem svijeta mogu dodavati nove članke, uređivati postojeće ili ispravljati netočnosti.

Hrvatska Wikipedija[9] je verzija Wikipedie na hrvatskom jeziku. Osnovana je 16. veljače 2003. godine i od tada je postala bogat izvor informacija o različitim temama na hrvatskom jeziku. Danas broji više od 200 tisuća članka te se pridržava istih načela suradničkog uređivanja kao i globalna Wikipedija, što znači da svatko može sudjelovati u stvaranju i uređivanju članaka.

Jedna od ključnih karakteristika Wikipedie, pa tako i Hrvatske Wikipedije, jest otvorenost i transparentnost. Svi članci i njihove promjene su javno dostupni i mogu se pratiti, što omogućava korisnicima da prate razvoj i kvalitetu informacija. Hrvatska Wikipedija igra važnu ulogu u očuvanju i širenju znanja na hrvatskom jeziku te omogućava korisnicima da pristupe informacijama o različitim temama bez obzira na njihovu lokaciju ili vremensku dostupnost.

5.2 *Python*

Python[10] je interpretativni programski jezik koji je popularan zbog svoje čitljive i jednostavne sintakse. Razvio ga je Guido van Rossum i prvi put je javno objavljen 1991. godine. *Python* se često naziva "jezikom za početnike", ali je također veoma moćan i koristi se u različitim područjima, uključujući *web development*, *data science*, razvoj umjetne inteligencije, automatizaciju, sistemske administracije i mnoge druge. Nekoliko ključnih karakteristika koje *Python* čine posebnim su:

- **Čitljivost i jednostavnost**: *Python* se ponosi svojom čitljivom sintaksom koja omogućava programerima da izraze svoje ideje na jasan način. Korištenjem minimalnog broja znakova i naglašavanjem strukture, *Python* kod je često lak za razumijevanje čak i osobama koje nisu iskusni programeri.

- **Dinamičko tipiziranje**: *Python* je dinamički tipiziran jezik, što znači da se tipovi varijabli određuju automatski prema njihovoj vrijednosti. Ova karakteristika olakšava pisanje koda i smanjuje potrebu za eksplicitnim određivanjem tipova varijabli.

- **Interpretirani jezik**: *Python* se izvršava kroz interpreter, što omogućava interaktivno testiranje i brzu izmjenu prilikom razvoja. Ovo je posebno korisno prilikom učenja i eksperimentiranja

- **Bogata biblioteka knjižica**: *Python* dolazi s obimnom standardnom bibliotekom koja pokriva širok spektar funkcionalnosti, uključujući rad s nizovima, datotekama, mrežnim protokolima, reguliranim izrazima, matematičkim operacijama i još mnogo toga. Ova značajka ubrzava razvoj aplikacija jer je mnoge uobičajene zadatke moguće obaviti koristeći već postojeće module i *framework*-e koji proširuju mogućnosti jezika za specifične namjene (npr. *Django*, *Flask*, *TensorFlow*, *pandas*).

- Podrška za **objektno orijentirano programiranje (OOP)**: *Python* podržava koncepte objektno orijentiranog programiranja, što omogućava organizaciju koda u klase i objekte radi bolje modularnosti i mogućnosti ponovnog korištenja.

Ukratko, *Python* je popularan programski jezik zbog svoje jednostavnosti, svestranosti i bogatog sustava biblioteka i *framework*-a te nudi razne mogućnosti za razvoj različitih vrsta programa bez obzira na iskustvo u području programiranja.

5.2.1 *wikipediaapi*

Python modul *wikipediaapi*[11] omogućava jednostavan pristup podacima sa Wikipedije putem *Python* skripti. Ovaj modul olakšava programerima da pretražuju, čitaju i analiziraju članke sa Wikipedije koristeći *Python* kod.

Modul se može instalirati koristeći alat *pip*. Dovoljno je izvršiti naredbu

```
pip install wikipedia-api
```

u terminalu, a nakon instalacije, modul se može učitati u *Python* skriptu koristeći naredbu

```
import wikipediaapi.
```

Za korištenje funkcionalnosti modula potrebno je kreirati instancu objekta *wikipediaapi.Wikipedia*:

```
wp = wikipediaapi.Wikipedia('[user_agent]', 'hr').
```

`"[user_agent]"` je oznaka po kojoj se prate sve programske obrade na Wikipediji.

Za dohvat članaka moguće je koristeći metodu page() čiji je argument naslov članka, a tekstu članka moguće je pristupiti pozivanjem atributa text:

```
1  page = wp.page('Python (programski jezik)')
2  print(page.text).
```

Rezultat gore prikazanog koda vidljiv je na slici 5.1b.

Uz ove i razne druge funkcionalnosti ovaj modul olakšava programerima pristup informacijama s Wikipedije te ih potiče da koriste te podatke za razne svrhe kao što su analize, istraživanje i još mnogo toga.

Python (programski jezik)

107 jezika

Stranica Razgovor Čitaj Uredi **Uredi kôd** Vidi povijest Pomagala

»Python« preusmjerava ovamo. Za druga značenja, pogledajte Python (razdvojba).

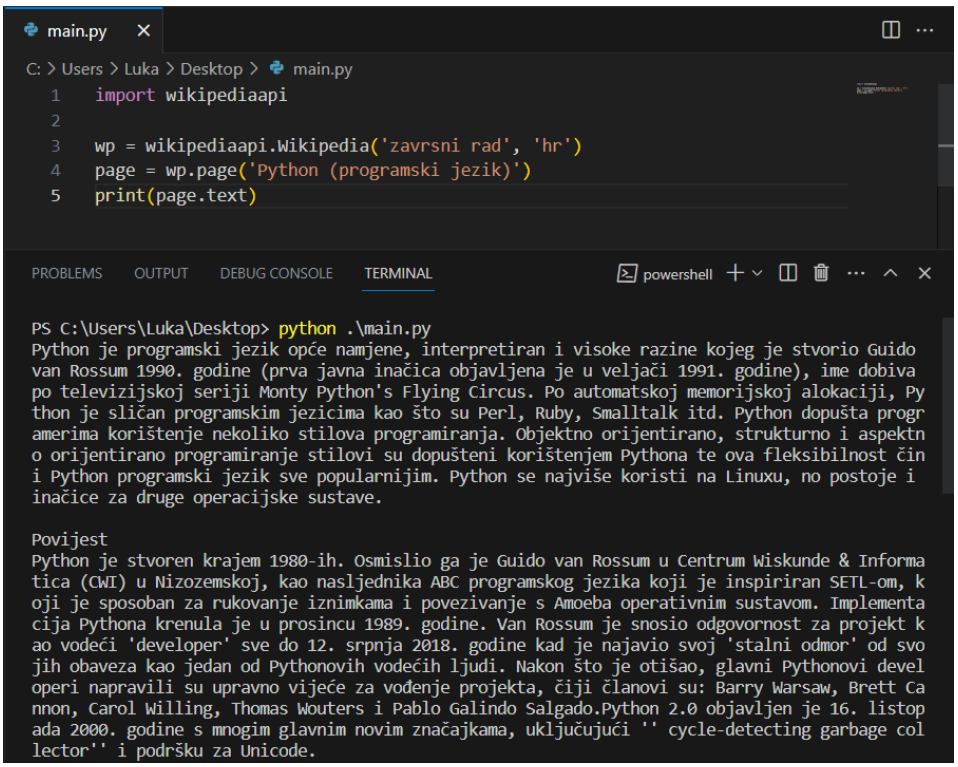
Python je programski jezik opće namjene, interpretiran i visoke razine kojeg je stvorio Guido van Rossum 1990. godine (prva javna inačica objavljena je u veljači 1991. godine),[1] ime dobiva po televizijskoj seriji Monty Python's Flying Circus. Po automatskoj memorijskoj alokaciji, Python je sličan programskim jezicima kao što su Perl, Ruby, Smalltalk itd. Python dopušta programerima korištenje nekoliko stilova programiranja. Objektno orijentirano, strukturno i aspektno orijentirano programiranje stilovi su dopušteni korištenjem Pythona te ova fleksibilnost čini Python programski jezik sve popularnijim. Python se najviše koristi na Linuxu, no postoje i inačice za druge operacijske sustave.

Povijest [uredi | uredi kôd]

Python je stvoren krajem 1980-ih.[2] Osmislio ga je Guido van Rossum u Centrum Wiskunde & Informatica (CWI) u Nizozemskoj, kao nasljednika ABC programskog jezika koji je inspiriran SETL-om,[3] koji je sposoban za rukovanje iznimkama i povezivanje s Amoeba operativnim sustavom. Implementacija Pythona krenula je u prosincu 1989. godine. Van Rossum je snosio odgovornost za projekt kao vodeći 'developer' sve do 12. srpnja 2018. godine kad je najavio svoj 'stalni odmor' od svojih obaveza kao jedan od Pythonovih vodećih ljudi. Nakon što je otišao, glavni Pythonovi developeri napravili su upravno vijeće za vođenje projekta, čiji članovi su: Barry Warsaw, Brett Cannon, Carol Willing, Thomas Wouters i Pablo Galindo Salgado.[4]

Python 2.0 objavljen je 16. listopada 2000. godine s mnogim glavnim novim značajkama, uključujući " cycle-detecting garbage collector" i podršku za Unicode.[5] Python 3.0 objavljen je 3. prosinca 2008. To je bila velika revizija jezika koja nije potpuno unatrag kompatibilna. Mnoge njegove značajke vraćene su u verzije Python 2.6.x i 2.7.x. Izdanja Pythona 3 uključuju pomoćni program 2to3, koji automatizira (barem djelomično) prijevod Python 2 koda na Python 3. Datum završetka Pythona 2.7 prvotno je bio određen za 2015. godinu, a zatim je odgođen za 2020. godinu zbog zabrinutosti da se veliki dio postojećeg koda ne može lako prenijeti na Python 3. Za njega neće biti objavljene više sigurnosne zakrpe niti druga poboljšanja. S isteklim vijekom trajanja

(a) Prikaz članka sa stranice hrvatske Wikipedije



main.py

C: > Users > Luka > Desktop > main.py

```
1  import wikipediaapi
2
3  wp = wikipediaapi.Wikipedia('zavrsni rad', 'hr')
4  page = wp.page('Python (programski jezik)')
5  print(page.text)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL powershell

```
PS C:\Users\Luka\Desktop> python .\main.py
Python je programski jezik opće namjene, interpretiran i visoke razine kojeg je stvorio Guido
van Rossum 1990. godine (prva javna inačica objavljena je u veljači 1991. godine), ime dobiva
po televizijskoj seriji Monty Python's Flying Circus. Po automatskoj memorijskoj alokaciji, Py
thon je sličan programskim jezicima kao što su Perl, Ruby, Smalltalk itd. Python dopušta progr
amerima korištenje nekoliko stilova programiranja. Objektno orijentirano, strukturno i aspektn
o orijentirano programiranje stilovi su dopušteni korištenjem Pythona te ova fleksibilnost čin
i Python programski jezik sve popularnijim. Python se najviše koristi na Linuxu, no postoje i
inačice za druge operacijske sustave.

Povijest
Python je stvoren krajem 1980-ih. Osmislio ga je Guido van Rossum u Centrum Wiskunde & Informa
tica (CWI) u Nizozemskoj, kao nasljednika ABC programskog jezika koji je inspiriran SETL-om, k
oji je sposoban za rukovanje iznimkama i povezivanje s Amoeba operativnim sustavom. Implementa
cija Pythona krenula je u prosincu 1989. godine. Van Rossum je snosio odgovornost za projekt k
ao vodeći 'developer' sve do 12. srpnja 2018. godine kad je najavio svoj 'stalni odmor' od svo
jih obaveza kao jedan od Pythonovih vodećih ljudi. Nakon što je otišao, glavni Pythonovi devel
operi napravili su upravno vijeće za vođenje projekta, čiji članovi su: Barry Warsaw, Brett Ca
nnon, Carol Willing, Thomas Wouters i Pablo Galindo Salgado.Python 2.0 objavljen je 16. listop
ada 2000. godine s mnogim glavnim novim značajkama, uključujući '' cycle-detecting garbage col
lector'' i podršku za Unicode.
```

(b) Prikaz tekstualnog dijela članka preuzetog pomoću modula *wikipediaapi*

Slika 5.1 Prikaz članka "*Python* (programski jezik)"

5.3 C

C[12] je popularan i moćan programski jezik koji je nastao u ranim 1970-ima. Razvio ga je Dennis Ritchie u laboratorijima Bell Telephone. C je visokorazinski jezik koji omogućava programerima da implementiraju efikasne i brze programe. Jedna od ključnih karakteristika jezika C je njegova bliskost sa *assebmlerom* - strojnim jezikom, što omogućava programerima da precizno upravljaju memorijom i resursima računalnog sustava.

C je poznat po svojoj sintaksi i setu osnovnih operacija. Programiranje u C-u zahtijeva od programera ručno upravljanje memorijom, poput alokacije i oslobađanja, što može biti izazovno, ali pruža veću kontrolu nad resursima. Sam jezik je osnova mnogim drugim tehnologijama kao što su operativni sustav *Unix* te programskim jezicima C++ i C#.

Poglavlje 6

Priprema ulaznih parametara

6.1 Prethodni rad

Ovaj je rad nastavak na projekt koji se bavio prebrojavanjem riječi hrvatske Wikipedije gdje je bilo potrebno preuzeti sve tekstove hrvatske Wikipedije te prebrojati pojedine riječi. Kako ne bi došlo do zabune, bitno je napomenuti da tekstovi hrvatske Wikipedije, osim hrvatskih riječi, sadrže i određenu količinu riječi stranih jezika i znakova koji nisu dio hrvatske abecede te brojeva koji su također ušli u riječnik. Primjerice, česti su znakovi ćirilice, azijskih pisama, posebni znakovi za prikaz matematičkih izraza te ostalih pisanih oblika (slika 6.1). Cilj tog projekta nije bilo stvoriti vokabular hrvatskih riječi nego samo prebrojati koliko se koji skup (niz) znakova, odvojen prazninom ili nekim interpunkcijskim znakom, ponavlja u tekstovima.

6.1.1 Stvaranje korpusa i vokabulara

Prije samog prebrojavanja riječi bilo je potrebno preuzeti sve tekstove hrvatske Wikipedije. Za to je korištena Python knjižica *wikipeadiaapi* koja omogućava preuzimanje teksta pojedinog članka. Na stranicama Wikipedije[13] može se pronaći datoteka koja sadrži sve naslove članaka na hrvatskom jeziku. Kombiniranjem tih dviju značajki, napravljena je skripta koja je, prolazivši naslove članaka na hrvatskom jeziku, preuzela tekst članka te ga spremila u datoteku s ostalim tekstovima. Dobivena je

```
2306319  Juhnov (ruski: Юхнов) — je upravno središte Juhnovskog
         rajona Kaluške oblasti u Rusiji.
2306320
2306321  Općenito
2306322  Kod OKATO za ovaj grad je 29250501. Brzoglasni pozivni broj
         je (+7) 48436.
2306323  Po stanju od ožujka 2007. (podatak sa ru.wiki),
         gradonačelnik je Nikolaj Sergejevič Utkin.
2306324
2306325  Zemljopisni položaj
2306326  Nalazi se na zapadu Kaluške oblasti, na desnoj obali rijeke
         Ugre, na auto-cestovnoj prometnici Moskva-Roslavlj (A101),
         35 km od željezničke postaje Mjatlevske linije
         Kaluga-Vjazma, 85 km od Kaluge. Zemljopisne koordinate su
         mu 54°45' sjeverne zemljopisne širine i 35°14' istočne
         zemljopisne dužine.
2306327  Vremenska zona: Moskovsko vrijeme (UTC+3, ljeti +4).
2306328
2306329  Povijest
2306330  Za ovaj grad se zna od 15. stoljeća, od utemeljenja
         juhnovskog kazanskog muškog samostana (Juhnovska pustinja,
         ruski: Юхновская Пустынь). Uzima se da je utemeljen 1410.
         godine.
```

Slika 6.1 Primjer dijela korpusa koji sadrži znakove koji ne pripadaju hrvatskoj abecedi

datoteka koja je sadržavala sve članke hrvatske wikipedije u tekstualnom obliku te je zauzimala 615 MB diskovnog prostora.

Za stvaranje vokabulara bilo je potrebno napraviti program koji će čitati riječi iz korpusa, provjeriti zadovoljava li riječ određene zahtjeve i u polje riječi povećati brojač za riječ koja već postoji ili je zapisati na kraj polja i postaviti brojač na 1. Korišten je C programski jezik te je za pohranu riječi i brojača svake od riječi korišteno polje struktura *identity* koja je sadržavala varijablu polje znakova *word* (za pohranu riječi) i varijablu za pohranu cjelobrojnih vrijednosti *counter* koja je sadržavala broj pojavljivanja riječi u korpusu što je vidljivo u kodu ispod.

```
6   #define SIZE 200
7   #define NUM_OF_EL 400000
8
9   typedef struct identity{
10      char word[SIZE];
11      int counter;
12  } identity;
```

U sljedećem dijelu koda prikazano je "čitanje" riječi iz ulazne datoteke (korpusa). Ako je riječ pronađena u polju *id*, inkrementiramo brojač te riječi, dok u suprotnom

novu riječ upisujemo na kraj polja *id* i njezin brojač postavljamo na vrijednost 1 te povećavamo varijablu *inscribed* koja sadrži broj upisanih riječi u polju.

```
44      identity *id = malloc(NUM_OF_EL * sizeof(identity));
45      while (fscanf(input, "%s", buffer) != EOF) {
46          counter++;
47          int i = 0;
48          while (1) {
49              if (!strcmp(id[i].word, buffer)) {
50                  id[i].counter++;
51                  break;
52              }
53              if (i == inscribed) {
54                  strcpy(id[i].word, buffer);
55                  id[i].counter = 1;
56                  inscribed++;
57                  break;
58              }
59              i++;
60          }
61      }
```

Kako se radilo o 90 milijuna riječi koje je trebalo sortirati i prebrojiti, korpus je podijeljen u 90 datoteka. U tim datotekama su zasebno prebrojavane riječi, a rezultati su zapisivani također u 90 različitih datoteka. Svaka datoteka sadržavala je više redova riječi i njihov broj ponavljanja. Svaka riječ nije smjela biti duža od 200 znakova (konstanta *SIZE* je iznosila 200) i svi interpunkcijski znakovi su služili kao granica između dviju riječi.

Nakon obrade cijelog korpusa, zasebnim programom implemetirano je povezivanje, potom sortiranje te na kraju ispis rezultata u završnu datoteku. Datoteka je sastavljena na način da je u svakom redu bila zapisana riječ, broj ponavljanja i postotak zastupljenosti u odnosu na cijeli korpus. Neki od bitnijih rezultata tog zadatka su:

- 89 937 040 riječi (korpus)

- 1 551 295 jedinstvenih riječi (vokabular)
- riječ "je", jedna od najučestalijih, pojavljuje se u korpusu 3 099 282 puta.

6.2 Filtriranje vokabulara i korpusa

Budući da je u stvorenom vokabularu primijećena poveća količina riječi koja ne bi doprinosila stvaranju jezičnog modela pristupilo se postupku filtriranja. Riječi koje su ušle u novu verziju vokabulara morale su zadovoljavati sljedeće:

- riječ se u korpusu mora pojavljivati **više od 100 puta**
- riječ mora sadržavati samo **znakove hrvatske abecede**
- riječ **ne smije sadržavati nikakvu vrstu navodnika**.

Kako je u ovom dijelu programske obrade vokabulara primijećeno da postoje riječi koje su se u korpusu nalazile na početku ili na kraju neke vrste citata ili navoda bilo je potrebno maknuti sve vrste navodnika ukoliko su bili sadržani kao prvi ili zadnji znak riječi. Primjerice *»riječ* i *riječ“* nisu bili dozvoljeni izrazi. Stoga je napravljena provjera i prepravak riječi koje su sadržavale znakove: „ “ » « ‘ ’ ¨ “. Uz navodnike, u ovoj fazi je eliminiran i česti znak: —.

Nakon filtriranja na razini znakova, bilo je potrebno ponovo "povezati" riječi koje su se pojavljivale više puta u vokabularu, zbrojiti iznose njihovih brojača i ponoviti sortiranje svih riječi. Tako je lista riječi za izradu jezičnog modela reducirana s otprilike 1.5 milijuna na 60295 riječi.

Kako je primijećeno da se alat *Word2vec* ne ponaša očekivano prilikom pokretanja prema danom vokabularu, što je testirano i u Windows i u Linux konfiguraciji, filtriran je i korpus na način da su iz njega izbačene riječi koje se nisu pojavile u završnoj verziji vokabulara. Također, u ovoj fazi je primijećeno da nijedna riječ nije dulja od 50 znakova što implicira da su riječi vokabulara i korpusa dobro probrane.

Poglavlje 7

Stvaranje modela i prikaz rezultata

U ovom će poglavlju biti predstavljen postupak te rezultati izgradnje jezičnog modela korištenjem hrvatske Wikipedije i alata *Word2Vec*. Analizirane su i dvije ključne metrike: udaljenost među riječima (*distance*) i analogije odnosa među riječima (*word-analogy*) primjenom dostupnih alata uz *Word2Vec*.

7.1 Stvaranje modela

Izvorni kod *Word2vec* alata preuzet je sa stranice *Google Code Archive*[3] te je izvršna datoteka dobivena pokretanjem naredbe

```
gcc word2vec.c -o word2vec -lm
```

u terminalu. Prilikom pokretanja kompajliranog programa *Word2vec* program ispisuje sve parametre (slika 7.1) kojima se određuje način, efikasnost i uvjeti modeliranja. Argumenti programa su:

- **naziv datoteke s tekstom, korpusom**
- **naziv datoteke u koju će model biti pohranjen**
- **veličina vektora** koji će predstavljati pojedinu riječ, broj dimenzija vektora
- **okvir** - broj riječi prije i poslije ciljane riječi koje ulaze u obzir prilikom obrade

- **prag pojavljivanja** u korpusu, brojač riječi s većim brojem pojavljivanja biti će nasumično reduciran unutar zadanog praga pojavljivanja
- korištenje algoritma **Hierarchical Softmax**
- **negative sampling** - broj nasumično odabranih riječi koje se koriste u treniranju, a nisu kontekstualno povezane s ciljanom riječi → na taj način model prilagođava parametre da maksimizira vjerojatnost predviđanja ciljane riječi
- **broj dretvi** prilikom stvaranja modela
- **broj iteracija** prilikom stvaranja modela
- **minimalni broj pojavljivanja** riječi - odbacuje riječi s manje pojavljivanja
- **learning rate** - broj koji određuje koliko će se brzo model prilagođavati, odnosno konvergirati težine prema ciljanim vrijednostima, predviđeno za *Skip-gram* 0.025, za *CBOW* 0.05
- način ispisa - **vektori ili klase**
- **debug** način - ispis više informacija prilikom obrade
- način zapisa izlazne datoteke - **tekstualno ili binary**
- **naziv datoteke iz koje se čita vokabular**
- **naziv datoteke u koju će biti spremljen vokabular modela**
- način stvaranja modela - ***Skip-gram*** **ili** ***CBOW***.

Kako je već spomenuto, *Word2vec* nudi dva načina stvaranja jezičnog modela, korištenjem *Skip-gram* ili *CBOW* metode. Prilikom stvaranja modela *Skip-gram* metodom korišteni parametri bili su:

```
./word2vec -train corpus_v3_spaces.txt -output
vectors_2023-08-13_skip-gram.bin -size 200 -window 5 -sample
1e-3 -hs 1 -negative 5 -threads 4 -iter 5 -min-count 1 -aplha
0.025 -binary 1 -cbow 0
```

dok je prilikom pokretanja *CBOW* metode korištena naredba bila:

```
./word2vec -train corpus_v3_spaces.txt -output
vectors_2023-08-13_cbow.bin -size 200 -window 5 -sample 1e-3
```

```
luka@luka-vb:~$ cd luka/word2vec
luka@luka-vb:~/luka/word2vec$ ./word2vec
WORD VECTOR estimation toolkit v 0.1c

Options:
Parameters for training:
        -train <file>
                Use text data from <file> to train the model
        -output <file>
                Use <file> to save the resulting word vectors / word clusters
        -size <int>
                Set size of word vectors; default is 100
        -window <int>
                Set max skip length between words; default is 5
        -sample <float>
                Set threshold for occurrence of words. Those that appear with higher frequency in the training data
                will be randomly down-sampled; default is 1e-3, useful range is (0, 1e-5)
        -hs <int>
                Use Hierarchical Softmax; default is 0 (not used)
        -negative <int>
                Number of negative examples; default is 5, common values are 3 - 10 (0 = not used)
        -threads <int>
                Use <int> threads (default 12)
        -iter <int>
                Run more training iterations (default 5)
        -min-count <int>
                This will discard words that appear less than <int> times; default is 5
        -alpha <float>
                Set the starting learning rate; default is 0.025 for skip-gram and 0.05 for CBOW
        -classes <int>
                Output word classes rather than word vectors; default number of classes is 0 (vectors are written)
        -debug <int>
                Set the debug mode (default = 2 = more info during training)
        -binary <int>
                Save the resulting vectors in binary moded; default is 0 (off)
        -save-vocab <file>
                The vocabulary will be saved to <file>
        -read-vocab <file>
                The vocabulary will be read from <file>, not constructed from the training data
        -cbow <int>
                Use the continuous bag of words model; default is 1 (use 0 for skip-gram model)

Examples:
./word2vec -train data.txt -output vec.txt -size 200 -window 5 -sample 1e-4 -negative 5 -hs 0 -binary 0 -cbow 1 -iter 3

luka@luka-vb:~/luka/word2vec$ S
```

Slika 7.1 Prikaz mogućih parametara prilikom pokretanja *Word2vec* alata

–hs 1 –negative 5 –threads 4 –iter 5 –min–count 1 –aplha 0.05 –binary 1 –cbow 1.

Oba postupka stvaranja modela rezultirala su datotekama veličine 46.2 MB na disku.

7.2 Udaljenost među riječima

Udaljenost među riječima (*distance*) je metrika koja mjeri semantičku sličnost između riječi u vektorskom prostoru. Vrijednosti se pojavljuju u rasponu od 0 do 1 te niža vrijednost udaljenosti ukazuje na manju semantičku sličnost između riječi, dok veći iznos ukazuje na veću sličnost.

Za analizu udaljenosti među riječima, korišten je alat dostupan uz *Word2Vec* biblioteku. Rezultati su prikazani u obliku matrice udaljenosti koja pruža uvid u semantičke veze između riječi (slika 7.2 i slika 7.3). Na osnovu ovih rezultata moguće je odrediti koliko su različite riječi povezane i grupirane u vektorskom prostoru.

Drugi primjeri rezultata *distance* metrike dostupni su u dodacima Appendix A.

```
Word: brat  Position in vocabulary: 1078

                                              Word       Cosine distance
------------------------------------------------------------------------
                                             nećak                0.846677
                                              otac                0.843387
                                             stric                0.827506
                                               sin                0.802487
                                          polubrat                0.799093
                                             rođak                0.794168
                                              unuk                0.785550
                                            bratić                0.764448
                                             mlađi                0.763549
                                              ujak                0.743295
                                              djed                0.739537
                                           praunuk                0.715327
                                            suprug                0.686489
                                           pradjed                0.680190
                                         naslijedio                0.679405
                                            sinovac                0.677400
                                         nasljednik                0.674075
                                            stariji                0.653587
                                          nasljedio                0.643063
                                           suvladar                0.640638
                                          prijatelj                0.640554
```

(a) *Skip-gram* metoda

```
Word: brat  Position in vocabulary: 1078

                                              Word       Cosine distance
------------------------------------------------------------------------
                                               sin                0.813370
                                              otac                0.778946
                                          polubrat                0.753991
                                             nećak                0.746336
                                             rođak                0.733470
                                              stric               0.726241
                                         nasljednik                0.699223
                                              ujak                0.692408
                                            bratić                0.671395
                                              unuk                0.663754
                                              djed                0.650322
                                            suprug                0.608492
                                           praunuk                0.594401
                                          prijatelj                0.593838
                                            sinovac                0.592493
                                           pradjed                0.583535
                                           suvladar                0.581236
                                               zet                 0.554706
                                         prethodnik                0.544695
                                          ljubavnik                0.526939
                                            potomak                0.521991
```

(b) *CBOW* metoda

Slika 7.2 Ispis riječi sličnih riječi "brat"

```
Word: abeceda  Position in vocabulary: 23417

                                        Word       Cosine distance
--------------------------------------------------------------------
                                   gramatika              0.588430
                                     abecede              0.578408
                                     znakova              0.551933
                                       slova              0.546937
                                   gramatike              0.486882
                                  glagoljici              0.478805
                                      nizova              0.478775
                                    latinica              0.468919
                                    sintaksa              0.465694
                                    latinici              0.464407
                                    formalna              0.460321
                                     alfabet              0.459777
                                  kontekstno              0.454444
                                       slovo              0.453190
                                    stringova              0.450667
                                   definicija              0.449180
                                    gramatiku              0.448541
                                   suglasnika              0.447059
```

(a) *Skip-gram* metoda

```
Word: abeceda  Position in vocabulary: 23417

                                        Word       Cosine distance
--------------------------------------------------------------------
                                   gramatika              0.495052
                                       slova              0.422943
                                     abecede              0.395785
                                   algoritam              0.391568
                                      oznaka              0.387374
                                      valuta              0.384013
                                       norma              0.380367
                                       šifra              0.374956
                                   varijanta              0.364553
                                  trobojnica              0.364369
                                   razdvojba              0.363285
                                  biblioteka              0.363071
                                     notacija              0.359781
                                   definicija              0.357250
                                      znakova              0.353448
                                  televizija              0.351498
                                       slovo              0.350396
                                    tradicija              0.345676
```

(b) *CBOW* metoda

Slika 7.3 Ispis riječi sličnih riječi "abeceda"

7.3 Analogije odnosa među riječima

Još jedna važna metrika koja omogućava ispitivanje semantičkih veza između riječi je i analogija odnosa među riječima (*Word-analogy*). Ova se metrika često koristi za evaluaciju jezičnih modela kako bi se provjerila njihova sposobnost da prepoznaju analogije.

Kroz ovaj eksperiment analizirane su analogije među riječima primjenjujući dostupan alat *word-analogy* raspoloživ uz *Word2vec*. Rezultati su predstavljeni u obliku analogija po principu: riječi "muškarac" i "žena" imaju istu analogiju kao riječi "kralj" i "kraljica" gdje su vektorske reprezentacije riječi "muškarac", "žena" i "kralj" ulazni parametri, a "kraljica" vrijednost koji želimo dobiti (slika 7.4 i slika 7.5). Ovakve analogije pružaju uvid u to je li model uspješno prepoznao semantičke odnose između različitih riječi.

```
Enter three words (EXIT to break): metar m kilometar

Word: metar  Position in vocabulary: 5365

Word: m  Position in vocabulary: 108

Word: kilometar  Position in vocabulary: 14410

                                          Word              Distance
------------------------------------------------------------------------
                                            km              0.606795
                                     kilometara             0.574846
                                         južno              0.564509
                                       sjeverno             0.564360
                                     kilometra              0.553309
                                  sjeverozapadno            0.530236
                                        istočno             0.529583
                                    jugoistočno             0.523370
                                         zapadno            0.520478
                                  sjeveroistočno            0.517118
                                         metara             0.515497
```

(a) *Skip-gram* metoda

```
Enter three words (EXIT to break): metar m kilometar

Word: metar  Position in vocabulary: 5365

Word: m  Position in vocabulary: 108

Word: kilometar  Position in vocabulary: 14410

                                          Word              Distance
------------------------------------------------------------------------
                                            km              0.515808
                                     kilometara             0.488699
                                         metara             0.442776
                                     kilometra              0.442639
                                          milja             0.418984
                                       sjeverno             0.398518
                                         južno              0.380617
                                        zapadno             0.368349
                                              n             0.357010
                                              e             0.348638
```

(b) *CBOW* metoda

Slika 7.4 Ispis mogućih rješenja za odnose vektora riječi "metar", "m" i "kilometar"

Drugi primjeri rezultata *word-analogy* metrike dostupni su u dodacima Appendix A.

```
Enter three words (EXIT to break): muškarac žena kralj

Word: muškarac  Position in vocabulary: 7525

Word: žena  Position in vocabulary: 517

Word: kralj  Position in vocabulary: 298

                                          Word              Distance
-------------------------------------------------------------------
                                      kraljica              0.590283
                                            ii              0.573716
                                           iii              0.570050
                                         karlo              0.556732
                                     jagelović              0.555195
                                         vilim              0.546821
                                        henrik              0.544695
                                            iv              0.540656
```

(a) *Skip-gram* metoda

```
Enter three words (EXIT to break): muškarac žena kralj

Word: muškarac  Position in vocabulary: 7525

Word: žena  Position in vocabulary: 517

Word: kralj  Position in vocabulary: 298

                                          Word              Distance
-------------------------------------------------------------------
                                      kraljica              0.436475
                                            ii              0.431707
                                           iii              0.431652
                                            iv              0.424696
                                     vladavina              0.413093
                                           car              0.408942
                                      kraljevi              0.395619
                                        regent              0.391969
```

(b) *CBOW* metoda

Slika 7.5 Ispis mogućih rješenja za odnose vektora riječi "muškarac", "žena" i "kralj"

7.4 *Cosine Distance*

Kosinusna udaljenost (engl. "*cosine distance*") je mjera sličnosti između dva vektora u višedimenzionalnom prostoru, a također se naziva i kosinusna sličnost. Ova mjera se često koristi za određivanje koliko su dva vektora usmjereni u istom smjeru, bez obzira na njihovu duljinu. Kosinusna sličnost računa se kao kosinus kuta između njih

u višedimenzionalnom prostoru te se postiže sljedećom formulom:

$$CosineSimilarity = \frac{A \cdot B}{\|A\|\|B\|} \tag{7.1}$$

gdje su:

1. A i B - vektori koje uspoređujemo
2. $\cdot$ - skalarni produkt vektora
3. $\|A\|$ - duljina (norma) vektora.

Kosinusna sličnost ima vrijednost između -1 (potpuno suprotni vektori) i 1 (potpuno isti vektori).

Poglavlje 8

Zaključak

U ovom istraživanju detaljno je razmotren proces izgradnje jezičnog modela pomoću alata *Word2Vec* te su u tom procesu korišteni članci hrvatske Wikipedije. Ovo završno poglavlje sažima glavne spoznaje i rezultate dobivene tijekom istraživanja ističući važnost i doprinos koji je metoda *Word2Vec* donijela u izgradnji jezičnih modela.

U prvom dijelu istraživanja površinski je analiziran koncept jezičnih modela i raznolikim vrstama jezičnih modela koji su prisutni u području obrade prirodnog jezika. Razumijevanje kako ovi modeli funkcioniraju i kako se koriste za različite jezične zadatke ključno je za ispravno primjenjivanje alata kao što je *Word2Vec*.

Detaljno je zatim istražen radni mehanizam alata *Word2Vec*. Uočeno je kako se temelji na distribuiranom predstavljanju riječi, odnosno na ideji da riječi koje se često pojavljuju zajedno imaju slična semantička značenja. Kroz proces kontekstualne predikcije i iterativno ažuriranje vektorskih reprezentacija riječi, *Word2Vec* generira kvalitetne vektore koji prepoznaju semantičke veze između riječi.

U sklopu procesa izgradnje jezičnog modela pomoću *Word2Vec*-a opisani su koraci potrebni za prikupljanje te obradu korpusa i vokabulara. Naglašena je važnost čišćenja teksta, tokenizacije i izgradnje konteksta za svaku riječ kako bi se osiguralo da model dobije relevantne informacije iz teksta. Kroz primjenu metrika *distance* i *word-analogy* pokazano je kako model može "hvatati" semantičke odnose i sličnosti među riječima.

Na temelju rezultata koji su postignuti, može se zaključiti da je *Word2Vec* moćan

alat za izgradnju jezičnih modela. Kroz primjenu na hrvatske Wikipedijine članke, prikazana je mogućnost prepoznavanja semantičkih veza između riječi te je ilustrirano kako model uspješno "hvata" kontekstualne značajke jezika. Ovaj rad pruža osnovu za daljnja istraživanja u domeni jezičnih modela kao i za primjenu tih modela u različitim jezičnim poput pretraživanja, klasifikacije i generiranja teksta.

U konačnici, *Word2Vec* predstavlja važan korak prema boljem razumijevanju i obradi jezika. Njegova sposobnost hvatanja semantičkih odnosa između riječi otvara put prema sveobuhvatnijem razumijevanju ljudskog jezika i omogućava razvoj naprednih jezičnih aplikacija koje će unaprijediti način na koji komuniciramo i koristimo jezik u digitalnom svijetu.

Bibliografija

[1] D. Jurafsky and J. H. Martin, *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Upper Saddle River, N.J.: Pearson Prentice Hall, 2009. , s Interneta, http://www.amazon.com/Speech-Language-Processing-2nd-Edition/dp/0131873210/ref=pd_bxgy_b_img_y

[2] Word2vec GitHub Repozitorij. , s Interneta, https://github.com/tmikolov/word2vec , srpanj 2017.

[3] Word2vec Google Code Archive. , s Interneta, https://code.google.com/archive/p/word2vec/ , srpanj 2013.

[4] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[5] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," 2013.

[6] Wikipedija članak Word2vec. , s Interneta, https://en.wikipedia.org/wiki/Word2vec , kolovoz 2023.

[7] Članak Word2Vec Explained. , s Interneta, https://towardsdatascience.com/word2vec-explained-49c52b4ccb71 , srpanj 2021.

[8] Wikipedia. , s Interneta, https://www.wikipedia.org/

[9] Wikipedija. , s Interneta, https://hr.wikipedia.org/wiki/Glavna_stranica

[10] Python Documentation. , s Interneta, https://www.python.org/

[11] Wikipedia-API. , s Interneta, https://pypi.org/project/Wikipedia-API/ , studeni 2017.

[12] The GNU C Reference Manual. , s Interneta, https://www.gnu.org/software/gnu-c-manual/gnu-c-manual.html , studeni 2008.

[13] hrwiki dump progress. , s Interneta, https://dumps.wikimedia.org/hrwiki/20230801/ , kolovoz 2023.

Sažetak

Ovaj rad temelji se na procesu izgradnje jezičnog modela uz primjenu alata *Word2Vec* na članke hrvatske Wikipedije. U uvodnom dijelu istražene su različite vrste jezičnih modela te njihova ulogu u obradi prirodnog jezika. Fokusirajući se na alat *Word2Vec* analiziran je njegov mehanizam distribuiranog predstavljanja riječi koji je temeljen na semantičkim sličnostima među riječima u kontekstu.

U praktičnom dijelu rada opisani su koraci izgradnje jezičnog modela pomoću alata *Word2Vec*. Proces je uključivao čišćenje i pripremu teksta, tokenizaciju te izgradnju konteksta za svaku riječ. Kroz iterativno ažuriranje vektorskih reprezentacija, model je usvojio semantičke značajke riječi, što smo potvrdili analizom metrika *distance* i *word-analogy*.

Rezultati dobiveni primjenom *Word2Vec*-a na članke hrvatske Wikipedije pokazali su da model uspješno "hvata" semantičke odnose između riječi doprinoseći razumijevanju jezičnih struktura. Ovaj rad pruža osnovu za daljnje istraživanje u području jezičnih modela te naglašava važnost alata *Word2Vec* u procesu izgradnje i razvoja jezičnih resursa.

Kroz ovaj rad jasno je prikazano da primjena *Word2Vec*-a omogućuje napredak u razumijevanju jezičnih obrazaca, a time i bolje iskorištavanje jezika u različitim aplikacijama. Ovakva istraživanja otvaraju vrata prema daljnjim inovacijama u području obrade prirodnog jezika potičući razvoj u načinu na koji komuniciramo i koristimo jezik u suvremenom digitalnom okruženju.

Ključne riječi — **Word2vec, Wikipedija, Jezični model, Obrada prirodnog jezika, Umjetne neuronske mreže, Distance, Word-analogy, Skip-gram, Continuous Bag Of Words**

Abstract

This paper is based on the process of building a language model using the *Word2Vec* tool on articles from the Croatian Wikipedia. In the introductory section, various types of language models and their role in natural language processing are explored.

Focusing on the *Word2Vec* tool, its mechanism of distributed word representation based on semantic similarities among words in context is analyzed.

The practical part of the work describes the steps involved in constructing a language model using the *Word2Vec* tool. The process included text cleaning and preparation, tokenization, and context creation for each word. Through iterative updates of vector representations, the model acquired semantic features of words, which were confirmed through the analysis of *distance* and *word-analogy* metrics.

The results obtained by applying *Word2Vec* to articles from the Croatian Wikipedia demonstrated that the model successfully captures semantic relationships between words, contributing to the understanding of linguistic structures. This work provides a foundation for further research in the field of language models and emphasizes the importance of the *Word2Vec* tool in the process of building and developing language resources.

Through this paper, it is clear that the application of *Word2Vec* enables progress in understanding linguistic patterns, thereby facilitating better utilization of language in various applications. Such research opens doors to further innovations in the field of natural language processing, encouraging development in how we communicate and use language in the modern digital environment.

Keywords **— Word2vec, Wikipedia, Language model, Natural language processing, Artificial neural networks, Distance, Word-analogy, Skip-gram, Continuous Bag Of Words**

Dodatak A

Pregled ostalih rezultata

Ovo poglavlje sadrži ostale rezultate dobivene *distance* i *word-analogy* metrikama. Usporedno se prikazuju rezultati modela nastalih *Skip-gram* i *CBOW* tehnikama. Rezultati prikazuju prvih 6 riječi i njihove kosinusne sličnosti dobivene upisivanjem riječi iz zaglavlja tablice.

A.1 Udaljenost među riječima - *Distance*

Tablica A.1 Rezultati za riječ "broj"

Riječ: **broj** Pozicija u vokabularu: 99			
Skip-gram		Continuous Bag of Words	
Riječ	Sličnost	Riječ	Sličnost
ukupan	0.671110	postotak	0.697313
postotak	0.547080	udio	0.660185
brojevi	0.535100	udjel	0.584961
broju	0.529158	dio	0.576590
brojem	0.526195	priljev	0.554816
porastao	0.525202	iznos	0.530836

Tablica A.2 Rezultati za riječ "a"

Riječ: **a** Pozicija u vokabularu: 10			
Skip-gram		Continuous Bag of Words	
Riječ	Sličnost	Riječ	Sličnost
te	0.664508	dok	0.792715
dok	0.658792	ali	0.764021
i	0.620468	te	0.750982
ali	0.548376	jer	0.733497
pa	0.539102	iako	0.710956
gdje	0.534970	no	0.702547

Tablica A.3 Rezultati za riječ "bajt"

Riječ: **bajt** Pozicija u vokabularu: 48392			
Skip-gram		Continuous Bag of Words	
Riječ	Sličnost	Riječ	Sličnost
bitova	0.683446	bita	0.374332
bita	0.528976	bitova	0.352930
kodiranje	0.494684	najmanjih	0.352648
ascii	0.475448	kulon	0.344806
decimalni	0.474972	ascii	0.338424
memoriji	0.462022 skup	0.337670	

Tablica A.4 Rezultati za riječ "matematika"

Riječ: **matematika** Pozicija u vokabularu: 17387			
Skip-gram		Continuous Bag of Words	
Riječ	Sličnost	Riječ	Sličnost
logika	0.711206	logika	0.476589
matematička	0.699094	matematička	0.464419
fizika	0.678136	filozofija	0.460458
filozofija	0.670700	psihologija	0.459838
sociologija	0.665217	metafizika	0.453976
matematike	0.661529	teorijska	0.451300

A.2 Analogije odnosa među riječima - *Word-analogy*

Tablica A.5 Rezultati za riječi "jedan", "dva" i "tri"

Riječ: **jedan** - Pozicija u vokabularu: 67 Riječ: **dva** - Pozicija u vokabularu: 74 Riječ: **tri** - Pozicija u vokabularu: 88			
Skip-gram		Continuous Bag of Words	
Riječ	Sličnost	Riječ	Sličnost
četiri	0.756160	četiri	0.777269
oba	0.593015	oba	0.632030
dvije	0.572233	dvije	0.502465
sva	0.497166	zadnjeg	0.426641
različita	0.492928	ovoga	0.410245
sljedeća	0.447942	istog	0.406021

Tablica A.6 Rezultati za riječi "engleska", "london" i "njemačka"

Riječ: **engleska** - Pozicija u vokabularu: 2653 Riječ: **london** - Pozicija u vokabularu: 1273 Riječ: **njemačka** - Pozicija u vokabularu: 599			
Skip-gram		Continuous Bag of Words	
Riječ	Sličnost	Riječ	Sličnost
berlin	0.678000	berlin	0.584055
weimar	0.595524	stuttgart	0.443629
stuttgart	0.575291	bonn	0.422789
hamburg	0.575097	köln	0.422329
stockholm	0.561315	bruxelles	0.421232
leipzig	0.553237	pariz	0.420084

Tablica A.7 Rezultati za riječi "francuska", "pariz" i "hrvatska"

Riječ: **francuska** - Pozicija u vokabularu: 733 Riječ: **pariz** - Pozicija u vokabularu: 2546 Riječ: **hrvatska** - Pozicija u vokabularu: 114			
Skip-gram		Continuous Bag of Words	
Riječ	Sličnost	Riječ	Sličnost
zagreb	0.590633	dubrovnik	0.414962
beč	0.522758	zagreb	0.406759
cobiss	0.497060	beč	0.393728
hrvatski	0.452062	beograd	0.379807
truhelka	0.436070	ljubljanu	0.369974
batušić	0.433526	pulu	0.368838