TakeLab

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

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 774

Primjena modela dubokog učenja na analizu sentimenta izraza hrvatskoga jezika

Siniša Biđin

Zagreb, lipanj 2014.

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

Zagreb, 10. ožujka 2014.

DIPLOMSKI ZADATAK br. 774

Pristupnik: **Siniša Biđin**
Studij: Računarstvo
Profil: Računarska znanost

Zadatak: **Primjena modela dubokog učenja na analizu sentimenta izraza hrvatskoga jezika**

Opis zadatka:

Porastom raspoloživih količina korisnički generiranog sadržaja povećalo se zanimanje za strojnom analizom sentimenta. Uobičajeni postupci analize sentimenta temelje se na rječniku apriornog sentimenta. Problem predstavlja modeliranje kompozicionalnosti, odnosno sentimenta višerječnih izraza poput "poprilično dobar" ili "nimalo loš", ali i većih jezičnih jedinica, poput rečenica ili odlomaka. Najnovija istraživanja pokazuju da je kompozicionalnost sentimenta moguće uspješno modelirati metodama dubokog učenja, koje koriste višeslojne ili hijerarhijske strukture s ciljem modeliranja složenih odnosa između podataka.
U okviru diplomskoga rada potrebno je proučiti postupke dubokog učenja s naglaskom na novije modele korištene u obradi prirodnog jezika te postupke za modeliranje sentimenta riječi i višerječnih izraza. Razraditi postupak analize sentimenta višerječnih izraza hrvatskoga jezika pomoću modela semantičke kompozicije zasnovanog na rekurzivnoj neuronskoj mreži prema radu Sochera i dr. (2012). Razviti programsku implementaciju postupka, možebitno se oslanjajući na javno dostupne biblioteke za duboko učenje. Izgraditi i ručno označiti odgovarajući skup podataka za učenje i ispitivanje. Provesti eksperimentalno vrednovanje postupka, usporedbu s odgovarajućim referentnim metodama, uključivo onima temeljenima na apriornim leksikonima, te detaljnu analizu pogrešaka. Razmotriti primjenu modela na predikciju sentimenta rečenica i većih dijelova teksta. Radu priložiti izvorni i izvršni kod razvijenog sustava i označene skupove podataka te citirati korištenu literaturu.

Zadatak uručen pristupniku: 14. ožujka 2014.
Rok za predaju rada: 30. lipnja 2014.

Mentor:

Doc. dr.sc. Jan Šnajder

Djelovođa:

Doc. dr.sc. Tomislav Hrkać

Predsjednik odbora za diplomski rad profila:

Prof. dr.sc. Siniša Srbljić

Sadržaj

1. Uvod

Obrada prirodnog jezika područje je računarske znanosti, umjetne inteligencije i lingvistike koje se bavi razvojem metoda za ekstrakciju informacija iz prirodnog jezika, poput metoda za automatsko sažimanje teksta, prevođenje teksta s jednog jezika na drugi, određivanje vrsta riječi u tekstu ili otkrivanje tekstualne reprezentacije govora.

Jedna primjena metoda obrade prirodnog jezika je i analiza sentimenta, postupak kojem je cilj koristeći tekst odrediti polaritet stava, emocionalnog stanja ili osjećaja koji se kroz taj tekst iznose. Tipičan primjer analize sentimenta bio bi postupak koji na temelju tekstualnih recenzija filmova pokušava odrediti jednostavnu reprezentaciju stava recenzenata: je li prosječno mišljenje o filmu pozitivno ili negativno?

Uobičajeni postupci analize sentimenta temelje se na rječniku apriornog sentimenta, gdje je svakoj riječi unaprijed dodijeljen određen sentiment, promatrajući je izvan konteksta. Problem nastaje prilikom analize sentimenta višerječnih izraza: ako je riječ "dobar" obilježena pozitivnim sentimentom, je li pozitivniji izraz "pomalo dobar" ili izraz "užasno dobar"? Kakvog je apriornog sentimenta riječ "užasno"? Kakvog je sentimenta izraz "nimalo dobar"?

Takvi su modeli vrlo ograničeni jer nisu u stanju naučiti ispravne reprezentacije kompozicija većeg broja riječi, što znači da ne mogu visokom točnošću odrediti značenje ili određeno semantičko svojstvo te kompozicije, poput sentimenta. Višerječni se izrazi sastoje od riječi koje mogu mijenjati značenje čitavog izraza ovisno o riječima unutar svojeg konteksta, što je svojstvo riječi koje bi bolji model mogao iskoristiti.

Ovim radom obrađen je uvod u rješenje tog problema. Cilj je uspješno klasificirati udjele različitih razina sentimenta u jednostavnim dvorječnim izrazima na hrvatskom jeziku, usredotočujući se prvenstveno na parove priloga i pridjeva.

S tim na umu, prvo su učene vektorske reprezentacije riječi. Definirane su dvije različite metode učenja takvih reprezentacija, prema radovima Collobert

et al. (2011) i Mikolov et al. (2013a), te implementirana jedna od njih. Vektorske su reprezentacije tada učene koristeći obje metode, na temelju dva različita pročišćena korpusa rečenica na hrvatskome jeziku. Tako dobivena četiri skupa vektorskih reprezentacija riječi uspoređena su mjerenjem međusobne sličnosti riječi unutar svakog od njih.

Takve vektorske reprezentacije ujedno su početni parametri modela za klasifikaciju sentimenta višerječnih izraza temeljenog na rekurzivnim neuronskim mrežama prema radu (Socher et al., 2012b). U radu je opisan takav model i njegova implementacija, pritom ograničena na dvorječne izraze. Zatim je izvršena evaluaciju modela nad tri različita skupa sentimentom označenih dvorječnih izraza: nad umjetnim skupom izraza označenih ručno izvan konteksta, nad skupom prevedenih izraza dohvaćenih iz filmskih recenzija na engleskome jeziku, te nad skupom izraza preuzetih iz recenzija restorana i hrane. Performanse konačnog modela izmjerene su i navedene u ovisnosti o korištenoj metodi učenja vektorskih reprezentacija riječi, korištenom korpusu rečenica na hrvatskome jeziku i korištenom skupu sentimentom označenih dvorječnih izraza.

Svi korišteni korpusi, skupovi podataka, implementacije i alati priloženi su radu u potpunosti, a važniji odabrani dijelovi navedeni su još i u dodatcima A i B.

Poglavlje 2 navodi motivaciju iza korištenja upravo algoritama dubokog učenja. Poglavljem 3 opisane su vektorske reprezentacije riječi i metode kojima se one mogu naučiti te navedeni rezultati učenja takvih reprezentacija nad različitim korpusima. Poglavljem 4 opisan je model za klasifikaciju sentimenta dvorječnih izraza, navedeni rezultati njegove evaluacije nad različitim skupovima za učenje te opisan način na koji se postupak može rekurzivno primijeniti na izraze dulje od dvaju riječi.

2. Duboko učenje

Duboko je učenje širok skup različitih algoritama strojnog učenja koji nastoje modelirati složene odnose između podataka koristeći arhitekture sastavljene od više nelinearnih transformacija (Bengio et al., 2012). Njihova je primjena nad mnogim problemima strojnog učenja rezultirala nadmoćnim performansama: pokazali su se najboljim ili gotovo najboljim rješenjem od područja prepoznavanja govora (Seide et al., 2011) do automatske anotacije i rangiranja glazbenih zapisa (Hamel et al., 2011); klasifikacije slikovnih zapisa znamenki (Rifai et al.; Ciresan et al., 2012) i objekata općenito (Krizhevsky et al., 2012); označavanja vrsta riječi, prepoznavanja imenovanih entiteta i sintaksnog parsanja rečenica[1] (Collobert et al., 2011); pronalaska različitih rečenica istog značenja (engl. *paraphrase detection*) (Richard Socher and Eric H. Huang and Jeffrey Pennington and Andrew Y. Ng and Christopher D. Manning, 2011) te analize sentimenta (Socher et al., 2012b, 2013). Nastavkom poglavlja opisana je motivaciju iza korištenja upravo takvih algoritama.

2.1. Motivacija

Većina trenutnih uspješnih postupaka u području strojnog učenja ovisi o ručno sastavljenim značajkama ulaznog skupa za učenje. Kako bi takvi sustavi postigli što bolje performanse, u skupu je za učenje potrebno istaknuti one značajke koje se smatraju relevantnima za donošenje odluka prilikom susretanja novih, nepoznatih primjera.

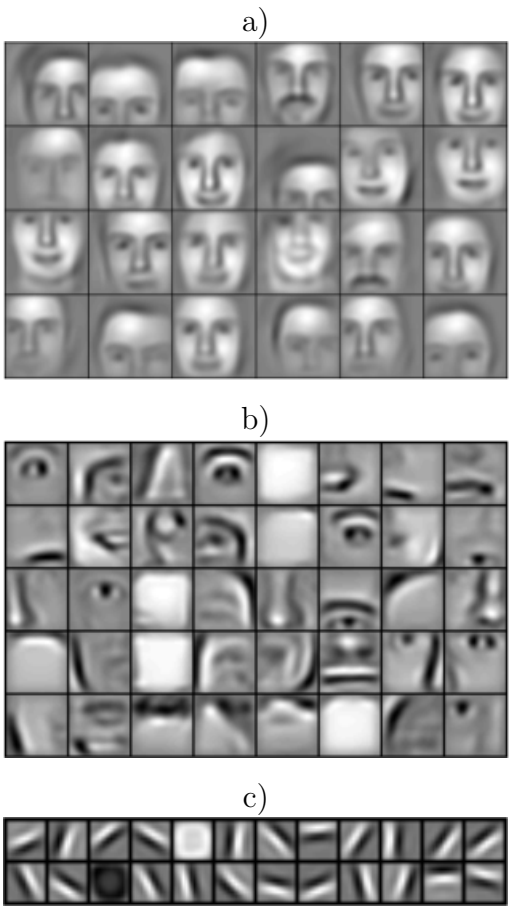
Ručna tvorba novih značajki može biti vrlo vremenski zahtjevan posao, a tako stvorene značajke nepotpune (u smislu da ne opisuju sva svojstva skupa za učenje važna za donošenje novih odluka) ili nepotrebno specifične (gdje ih opisuju više puta, kao i nevažna svojstva). Značajke je još potrebno iznova određivati za svaku novu primjenu: značajke potrebne za detekciju teksta u slici nisu nužno iste kao

[1] http://ml.nec-labs.com/senna/

i one za detekciju ljudskog lica.

Problem je i nedostatak označenih primjera za učenje: gotovo svi podatci koji su nam dostupni nisu označeni. Stoga je vrlo korisno imati na raspolaganju metode koje postižu dobre rezultate usprkos tom problemu, u što većoj mjeri nenadziranim učenjem. Postupci dubokog učenja omogućuju nam nenadziranu gradnju modela promatranih podataka koji nam zatim olakšavaju učenje klasifikatora nad tim podatcima (Socher et al., 2012a).

Učenje značajki ili reprezentacije (engl. *feature learning, representation learning*) postupak je koji pokušava automatski naučiti dobre značajke nekog skupa podataka, tako da se njihova ručna gradnja može zaobići. Upravo metode dubokog učenja pružaju nam odličan način automatskog učenja apstraktnijih i u konačnici korisnijih reprezentacija podataka (Bengio et al., 2012).



Slika 2.1: Slojevi modela za nenadzirano učenje reprezentacija objekata, u ovom slučaju lica ljudi (Lee et al., 2009). Viši slojevi (a, b) uče više razine reprezentacije objekata (čitava lica i njihove dijelove), dok niži (c) uče reprezentacije jednostavnih geometrijskih oblika.

3. Vektorske reprezentacije riječi

Rezultati obrade prirodnog jezika metodama dubokog učenja uvelike ovise o neuronskim jezičnim modelima koji riječi predstavljaju višedimenzionalnim vektorima realnih vrijednosti (Levy i Goldberg, 2014). Točne vrijednosti takvih vektorskih reprezentacija riječi uče se na temelju velikih količina tekstova koristeći različite varijante neuronskih mreža (Collobert et al., 2011; Mikolov et al., 2013a) s ciljem poprimanja takvih vrijednosti koje što bolje opisuju sličnosti (Turney, 2006) između riječi leksikona.

Riječi koje se pojavljuju u sličnim kontekstima nakon obrade takvim metodama poprimit će vektorske reprezentacije koje su si međusobno bliže od reprezentacija onih riječi koje se pojavljuju u drugačijim kontekstima. Time se grupiraju reprezentacije riječi koje međusobno dijele semantička svojstva ("lopov", "razbojnik", "provalnik"), a upravo tako grupirane vektorske reprezentacije riječi pokazale su se kao odlične značajke za različite vrste primjena u obradi prirodnog jezika (Collobert et al., 2011; Socher et al., 2011, 2012b), uključujući i analizu sentimenta.

Pošto učenje i korištenje takvih reprezentacija direktno poboljšava performanse modela za klasifikaciju sentimenta (Erhan et al., 2010), nastavkom poglavlja učene su vektorske reprezentacije riječi koristeći dvije različite metode nad dva različita korpusa.

3.1. Diskretne i kontinuirane reprezentacije

Jedna od glavnih značajki jezičnih modela temeljenih na neuronskim mrežama je njihova reprezentacija riječi kao više-dimenzionalnih vektora realnih vrijednosti. Čitav se leksikon takvim modelima preslikava u jedan prostor relativno niskog broja dimenzija. Glavna prednost takvog modela je što takvom distribuiranom reprezentacijom riječi uspijeva postići velik stupanj generalizacije, za razliku od klasičnih pristupa koji riječi modeliraju diskretnim vektorima vrlo visoke dimen-

zionalnosti (engl. *one-hot vectors*).

$$\begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{bmatrix} \quad \text{diskretan vektor}$$

$$\begin{bmatrix} 0.0145 & 0.031 & 2.141 & \cdots & -0.845 & 0.345 & \cdots & -0.23 \end{bmatrix} \quad \text{kontinuirani vektor}$$

Idealno, kontinuiranim su vektorima riječi predstavljene na takav način da sličnije riječi imaju sličnije vektorske reprezentacije. Takvo je svojstvo korisno u mnogim primjenama (Baroni et al., 2014), a u slučaju analize sentimenta važno je prvenstveno jer poboljšava performanse modela koji takve vektorske reprezentacije koriste kao svoje početne parametre (Erhan et al., 2010).

Diskretne je vektore lako odrediti: direktno se stvaraju na temelju korištenog leksikona. Kontinuirane je pak potrebno naučiti posebnim postupkom. Pošto su određeni s ciljem kasnijeg korištenja kao početnih parametara modela za klasifikaciju sentimenta, njihovo učenje naziva se još i predučenjem (engl. *pretraining*).

Postoji mnogo različitih modela za učenje kontinuiranih reprezentacija riječi, no u ovome radu usredotočuje se na reprezentacije riječi naučene neuronskim mrežama, pošto je pokazano kako njihova uporaba rezultira znatno većim sličnostima između rezultirajućih vektora riječi (Mikolov et al., 2013c), što je svojstvo koje se željelo iskoristiti naknadnim postupkom klasifikacije sentimenta.

Ostatkom poglavlja opisana su dva različita pristupa generaciji vektorskih reprezentacija riječi: usporedbom ispravnog i neispravnog konteksta riječi (Collobert et al., 2011) i modelom kontinuiranih vreća riječi (engl. *continuous bag-of-words model, CBOW*) (Mikolov et al., 2013a), te su opisani rezultati dobiveni takvim pristupima nad više korpusa. Kako odabrani pristupi utječu na konačne performanse klasifikacije sentimenta pokazano je u poglavlju 4.4.

3.2. Predučenje usporedbom konteksta

Prema ovome modelu (Collobert et al., 2011), vektori riječi uče se na temelju usporedbe riječi u svome ispravnom kontekstu sa riječi u neispravnom kontekstu. Riječ u svome uobičajenom kontekstu predstavlja pozitivan primjer, dok taj isti kontekst s drugom, nasumičnom riječi predstavlja negativan.

ispravan kontekst: *"bio je **odličan** kao premijer"*

neispravan kontekst: *"bio je **diplomirati** kao premijer"*

Cilj je naučiti takav model koji će pozitivnom primjeru dodijeliti veću vrijednost od negativnog. Rezultirajući parametri modela, nakon učenja, bit će upravo

traženi vektori pojedinih riječi.

3.2.1. Model

Vrijednost nekog izraza (sačinjenog od riječi unutar konteksta) računat će se neuronskom mrežom. Neka je svaka riječ predstavljena svojim vektorom $w \in \mathbb{R}^n$, u početku inicijaliziranim na nasumične vrijednosti. Vektori svih riječi leksikona V zajedno čine matricu vektora riječi $L \in \mathbb{R}^{n \times |V|}$.

$$L = \begin{bmatrix} w_{1,1} & w_{2,1} & \dots & w_{|V|,1} \\ w_{1,2} & w_{2,2} & \dots & w_{|V|,2} \\ \dots & \dots & \dots & \dots \\ w_{1,n} & w_{2,n} & \dots & w_{|V|,n} \end{bmatrix}$$

Takvi vektori riječi su ujedno i značajke riječi koje se želi naučiti, odnosno parametri modela koje će se daljnjim koracima optimizirati. Vektor pojedine riječi iz matrice L uzima se množeći je s *one-hot*-vektorom e, gdje e predstavlja poziciju riječi w u leksikonu V.

$$w = Le$$

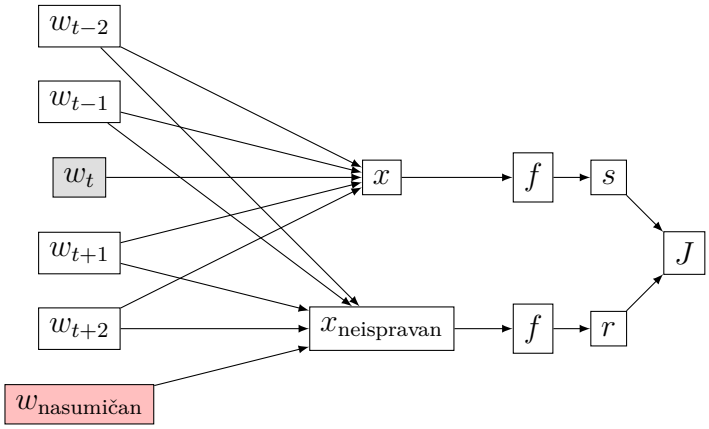
Jedna riječ zajedno sa svojim kontekstom predstavlja izraz duljine k koji je opisan vektorom sastavljenim od vektora pojedinačnih riječi izraza. Na primjer, vektor izraza "bio je **odličan** kao premijer" sastoji se od $k = 5$ vektora riječi w_1 ("bio"), w_2 ("je"), w_3 ("odličan"), w_4 ("kao") te w_5 ("premijer"), spojenih u $x \in \mathbb{R}^{kn}$:

$$x = \begin{bmatrix} w_{1,1} & w_{1,2} & \dots & w_{1,n} & w_{2,1} & \dots & \dots & \dots & w_{k,n} \end{bmatrix}^{\mathrm{T}}.$$

Vrijednost određenog izraza x određen je troslojnom neuronskom mrežom. Definicijom dodatnih parametara $W \in \mathbb{R}^{m \times kn}$, $U \in \mathbb{R}^m$, $b \in \mathbb{R}^m$ (početno postavljenih na nasumične vrijednosti) te nelinearne funkcije f (npr. sigmoidalne), krajnju vrijednost $s \in \mathbb{R}$ izraza x tada opisujemo kao

$$\begin{aligned} z &= Wx + b \\ a &= f(z) \\ s &= U^{\mathrm{T}} a. \end{aligned}$$

Učenje modela vrši se optimizirajući parametre na taj način da maksimiziraju vrijednosti izraza sa ispravnim kontekstom s, a minimiziraju vrijednosti izraza sa neispravnim kontekstom r. Odnosno, traže se parametri L, W, b i U takvi da za



Slika 3.1: Skica modela usporedbe konteksta.

svaki par izraza sa ispravnim i neispravnim kontekstom s i r funkcija J postiže što manju (bolju) vrijednost.

$$\begin{aligned} s &= U^{\mathrm{T}} f(Wx + b) \\ r &= U^{\mathrm{T}} f(W x_{\text{neispravan}} + b) \\ J(L, W, b, U) &= \max(0, 1 - s + r) \end{aligned}$$

3.2.2. Učenje

Optimalni parametri pronalaze se jednim od optimizacijskih algoritama. Priložena implementacija ovog postupka, zbog vrlo velike količine podataka koje ćemo obrađivati, kao optimizacijski algoritam koristi stohastički gradijentni spust.

Kako bi se, ovisno o vrijednosti jednog ulaznog para izraza, korektno izmijenili parametre modela, potrebno je odrediti derivacije J, odnosno s i r u ovisnosti o svim korištenim parametrima. Pretpostavljajući da je vrijednost $J > 0$, slijede derivacije izraza s (iz kojih slijede oni za r).

$$\frac{\partial s}{\partial U} = \frac{\partial}{\partial U} U^{\mathrm{T}} a = a = f(Wx + b)$$

Derivacija po matrici težina W je nešto složenija. Određena je prvo derivacija

po jednoj težini W_{ij}. Ona se koristi samo za računanje jednog a_i, stoga:

$$\begin{aligned}
\frac{\partial s}{\partial W_{ij}} &= \frac{\partial}{\partial W_{ij}} U^{\mathrm{T}} a \\
&= \frac{\partial}{\partial W_{ij}} U_i a_i \\
&= U_i \frac{\partial a_i}{\partial z_i} \frac{\partial z_i}{\partial W_{ij}} \quad \left(\text{slijedi iz } \frac{\partial y}{\partial x} = \frac{\partial y}{\partial u}\frac{\partial u}{\partial x}\right) \\
&= U_i \frac{\partial f(z_i)}{\partial z_i} \frac{\partial z_i}{\partial W_{ij}} \\
&= U_i f'(z_i) \frac{\partial z_i}{\partial W_{ij}} \\
&= U_i f'(z_i) \frac{\partial \left(W_i x + b_i\right)}{\partial W_{ij}} \\
&= U_i f'(z_i) \frac{\partial}{\partial W_{ij}} \sum_{m=1}^{kn} W_{im} x_m \\
&= U_i f'(z_i) \frac{\partial W_{ij} x_j}{\partial W_{ij}} \\
&= U_i f'(z_i) x_j
\end{aligned}$$

Kažemo još i $\delta_i = U_i f'(z_i)$. Derivacija s po svim težinama tada iznosi:

$$\frac{\partial s}{\partial W} = \delta x^{\mathrm{T}}.$$

Izraz δ_i nalazi se i u derivaciji po parametru b_i, što omogućuje da prilikom implementacije postupka nije potrebno računati iste vrijednosti više puta.

$$\begin{aligned}
\frac{\partial s}{\partial b_i} &= U_i \frac{\partial}{\partial b_i} a_i \\
&= U_i f'(z_i) \frac{\partial W_{i\cdot} x + b_i}{\partial b_i} \\
&= U_i f'(z_i) = \delta_i \\
\frac{\partial s}{\partial b} &= \delta.
\end{aligned}$$

Konačno, derivacija po parametru x iznosi:

$$\begin{aligned}
\frac{\partial s}{\partial x_j} &= \sum_{i=1}^{m} \frac{\partial s}{\partial a_i} \frac{\partial a_i}{\partial x_j} \\
&= \sum_{i=1}^{m} \frac{\partial U^{\mathrm{T}} a}{\partial a_i} \frac{\partial a_i}{\partial x_j} \\
&= \sum_{i=1}^{m} U_i \frac{\partial f(W_{i\cdot} x + b)}{\partial x_j} \\
&= \sum_{i=1}^{m} U_i f'(W_{i\cdot} x + b) \frac{\partial W_{i\cdot} x}{\partial x_j} \\
&= \sum_{i=1}^{m} \delta_i W_{ij} \\
&= \delta^{\mathrm{T}} W_{\cdot j}
\end{aligned}$$

Preostaje evaluirati J nad svakim parom ispravnog i neispravnog konteksta te izmijeniti parametre modela u ovisnosti o izračunatim gradijentima.

3.2.3. Implementacija

Vektorske reprezentacije uče se priloženom vlastitom implementacijom (v. dodatak A). Uče se reprezentacije riječi $w \in \mathbb{R}^8$, pošto se pokazalo da su vektori znatno veće dimenzionalnosti utječu negativno na krajnje performanse sustava za klasifikaciju sentimenta (Socher et al., 2012b). Pritom se koristi kontekst duljine pet riječi (dvije ispred, dvije nakon). Prilikom optimizacije parametara stohastičkim gradijentim spustom kao početna stopa učenja α koristi se 0.1, te se linearno umanjuje do nule kako postupak obrađuje sve primjere. Rezultati postupka navodeni su u poglavlju 3.5.

Potrebna programska podrška

Prije korištenja priložene implementacije, potrebno je instalirati dodatne alate i biblioteke.

- Najnoviji interpreter programskog jezika *Python3*.[1]
- Skup paketa za matematiku, znanost i inženjerstvo *SciPy*.[2]
- Biblioteku za efikasnu evaluaciju matematičkih izraza *Theano*.[3]

[1] https://www.python.org/downloads
[2] http://www.scipy.org/install.html
[3] http://www.deeplearning.net/software/theano

Primjer izvršavanja

Priloženi program kao ulaz zahtijeva put do datoteke s korpusom rečenica (jedna rečenica po liniji), datoteke s leksikonom (jedna riječ po liniji) i, neobavezno, najveći broj iteracija algoritma nakon kojih program prestaje s radom.

```
$ ./wordvecs-collobert-weston.py corpus.txt lexicon.txt 100000000
```

Nakon učenja, rezultirajući su vektori reprezentacija riječi pohranjeni u lokalnu datoteku `./vectors-collobert-weston.npy`.

Trajanje rada ovog programa ovisi uvelike o veličini korpusa, no u pravilu je program, za imalo znatnije korpuse, vrlo vremenski zahtjevan. Uspoređujući popularne metode generacije distribuiranih vektora riječi, ova je metoda daleko najsporija (Mikolov et al., 2013b). Kao primjer, učenje vektora reprezentacija riječi na temelju engleske Wikipedije, veličine 44GB, trajalo je dva mjeseca.[4]

3.3. Predučenje kontinuiranim vrećama riječi

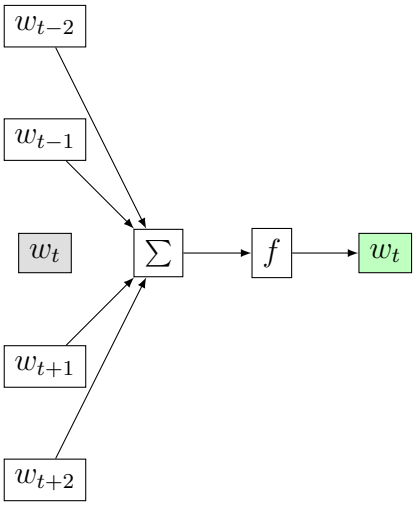
Radi velike vremenske zahtjevnosti prve opisane metode (v. odjeljak 3.2.3) te usporedbe konačnih rezultata klasifikacije sentimenta u ovisnosti o metodi učenja početnih vektora reprezentacija riječi, osim metode usporedbe konteksta koristi se i ona kontinuiranih vreća riječi (engl. *continuous bag-of-words, CBOW*) te je u nastavku vrlo ukratko opisana.

Kontinuirane vreće riječi (Mikolov et al., 2013a) model je koji pokušava odrediti vektor neke riječi u ovisnosti o vektorima riječi koje čine njen kontekst. Kao kontekst se uzimaju i prethodeće i slijedeće riječi, no njihov redoslijed nema utjecaj na konačan rezultat (stoga i ime: vreća riječi). Ovaj se model pokazao kao najbolji trenutan model za zadatak određivanja sličnih riječi (Baroni et al., 2014), mnogo bolji od prvog opisanog modela usporedbe konteksta.

Kontekst je sačinjen od niza vektora riječi koje se jednostavnom operacijom (npr. sumom) pretvara u jedan vektor. Zatim se nad tim vektorom primjenjuje klasifikator koji rezultira procjenom vektora riječi s odabranim kontekstom (slika 3.2).

Na temelju razlike procjene vektora i njegove stvarne vrijednosti računa se greška koja se zatim, kao i u slučaju metode usporedbe konteksta, koristi kako bi

[4] http://ronan.collobert.com/senna/



Slika 3.2: Model kontinuiranih vreća riječi (Mikolov et al., 2013a).

se izmijenili parametri klasifikatora, odnosno sami početni vektori reprezentacija riječi.

Umjesto korištenja vlastite implementacije, za učenje ovakvih vektora koristi se postojeći alat `word2vec`[5] autora metode (Mikolov et al., 2013a). Učenje ovih vektora mnogo je brži proces od učenja metodom usporedbe konteksta (Mikolov et al., 2013b) te ga stoga koristimo nad dostupnim korpusima u potpunosti. Rezultate navodimo u poglavlju 3.5.

Primjer izvršavanja

Prije korištenja alata, potrebno je instalirati sve potrebne alate i biblioteke kao i za metodu usporedbe konteksta navedene u odjeljku 3.2.3. Kao i tada, uče se vektorske reprezentacije riječi $w \in \mathbb{R}^8$ unutar konteksta duljine pet riječi (dvije ispred i dvije nakon), no ovaj put koristeći algoritam kontinuiranih vreća riječi.

Priloženi program mora primiti put do datoteke s korpusom te željenu veličinu vektora reprezentacija riječi, a nakon izvršavanja automatski poziva alat `word2vec` te vrši potrebne konverzije između oblika podataka.

```
$ ./wordvecs-from-corpus.py fhrwac.txt 8
```

Rezultirajući vektori reprezentacija riječi nakon završetka procesa pohranjeni su u lokalnu datoteku `./vectors.npy`.

[5] https://code.google.com/p/word2vec

3.4. Korišteni korpusi rečenica

Kako bi se što bolje naučile vektorske reprezentacije riječi te zatim postigli bolji rezultati klasifikacije sentimenta izraza, potreban je dobar korpus rečenica na hrvatskom jeziku. Rezultirajuće reprezentacije mogu u velikoj mjeri ovisiti o korištenom korpusu: korpusom novinskih članaka naučit će se dobre reprezentacije formalno pisanog teksta, no ne nužno i neformalnog. Ukoliko se reprezentacije kasnije namjeravaju koristiti u postupcima nad neformalno pisanim tekstom, želja nam je korpusom obuhvatiti i veliku količinu takvih tekstova.

Nastavkom poglavlja opisuju se dva korištena korpusa: onaj web-stranice forum.hr i korpus fHrWaC. Zatim se opisuje metoda kojom se korpusi pripremaju za predučenje te navode rezultati predučenja reprezentacija riječi u ovisnosti o korištenom korpusu.

3.4.1. Korpus forum.hr

Korpus web-stranice forum.hr sastoji se od 298,522,012 riječi iz 6,273,118 tekstova pisanih u vrlo neformalnom stilu, često prepunim pravopisnih i gramatičkih pogrešaka. Rečenice ovog korpusa ponekad nisu korektno razdvojene, jer su tekstovi izvorno pisani ne koristeći interpunkciju. Mnoge su rečenice još pisane i ne koristeći dijakritike: riječi "čovjek", "ćovjek" i "covjek" stoga se smatraju potpuno različitim riječima, što će umjetno uvećati veličinu leksikona i negativno utjecati na kvalitetu konačnih reprezentacija riječi.

- *"Ellen je okej samo i ona nekad pretjera."*
- *"ja sam davno rekao ovo: da su vlast u hrvatskoj preuzeli komunisti"*
- *"PISMA JE SUPER i od svih do sad najbolja i meni!"*

Tekstovi korpusa odabrani su temeljem njihove pripadnosti dijelovima webforuma unutar kojih se očekivala veća gustoća riječi koje sadrže sentiment (poput dijelova vezanih uz politiku, glazbu i sport), s nadom da će takav korpus biti od veće koristi za postupke analize sentimenta. Korpus je radu priložen u potpunosti.

3.4.2. Korpus fHrWaC

Korpus fHrWaC (Šnajder et al., 2013) filtrirana je verzija korpusa hrWaC (Ljubešić i Erjavec, 2011) koja sadrži 50,940,598 rečenica i 1,232,632,208 riječi dohvaćenih s raznih web-izvora na hrvatskom jeziku. Za razliku od korpusa forum.hr,

korpus hrWaC sadrži mnogo manje gramatičkih i pravopisnih grešaka i velikim je dijelom pisan u formalnom stilu.

- *"Klijent je suprugu obavijestio da ide na kraći službeni put."*
- *"Poslovnik stupa na snagu danom donošenja."*
- *"Pedesete godine su vrijeme ljestvica i vitla te timskog rada velikog broja sudionika istraživanja."*

Jasno razdvojene rečenice trebale bi pomoći performansama metodi usporedbe konteksta, pošto neće doći do pogrešnog preklapanja konteksta jedne riječi s rečenicom u kojoj se one ne nalazi. Takvih je grešaka više prilikom obrađivanja korpusa forum.hr, pošto njegovi tekstovi često uopće ne sadrže interpunkciju.

3.5. Rezultati predučenja

Koristeći metode usporedbe konteksta (v. odjeljak 3.2) i kontinuiranih vreća riječi (v. odjeljak 3.3), vektorske se reprezentacije riječi uče nad oba korpusa forum.hr (v. odjeljak 3.4.1) i fHrWaC (v. odjeljak 3.4.2). Prethodno se svaki korpus pojednostavljuje idućim koracima:

1. Uklanjaju se svi znakovi osim slova hrvatske i engleske abecede.
2. Sva se slova umanjuju.
3. Vrši se lematizacija teksta: svaku riječ zamjenjuje se njenom lemom.

Lematizacija se vrši na temelju poluautomatski izgrađenog morfološkog leksikona, postupkom opisanim u (Šnajder et al., 2008).

Nad naučenim vektorima vrši se upit: koje su riječi w_i najsličnije nekoj riječi z? Kao mjeru sličnosti između dvaju riječi koristimo kosinus kuta između njihovih vektora (engl. *cosine similarity*):

$$\text{sličnost}(w, z) = \cos\theta_{w,z} = \frac{w \cdot z}{||w||\ ||z||} = \frac{\sum_{i=1}^{n} w_i z_i}{\sqrt{\sum_{i=1}^{n} w_i^2}\sqrt{\sum_{i=1}^{n} z_i^2}}$$

Rezultati upita za četiri odabrane riječi vidljivi su u tablicama 3.1, 3.2, 3.3 i 3.4. Za svaku riječ navode se četiri tablice: po jedna za svaku kombinaciju korištenog korpusa i metode učenja vektorskih reprezentacija riječi. Svaka tablica sadrži prvih 11 najsličnijih riječi koje se ujedno pojavljuju među 10^5 najčešćih riječi korištenog korpusa.

Tablicama je vidljivo da model kontinuiranih vreća riječi postiže bolje rezultate nad oba korpusa od modela usporedbe konteksta (npr˙ pogotovo tablica 3.4), što je konzistentno s usporedbama dvaju metoda koje pokazuju da su kontinuirane vreće riječi mnogo bolji model za zadatak sličnosti među riječima (Baroni et al., 2014).

Još je vidljiv i velik utjecaj korištenog korpusa na pronađene riječi: postupak nad neformalnije pisanim tekstovima korpusa forum.hr rezultira mnogim riječima koje se ne mogu naći u novinskim člancima, poput "brukaš" ili "lopine". S druge strane, korištenje korpusa fHrWaC rezultira bogatijim rasponom sličnih riječi, što je objašnjivo manjom veličinom i mnogo skromnijim leksikonom neformalnijeg od dvaju korpusa (korpus forum.hr sadrži 416371 jedinstvenih riječi, a fHrWaC gotovo dvostruko više: 752594).

Kako svaka od četiri inačice naučenih vektora reprezentacija riječi utječe na konačne performanse modela za klasifikaciju sentimenta pokazuje se među rezultatima navedenim u idućem poglavlju.

Usporedba konteksta, forum.hr.	
w	$\cos(\theta_{w,\text{lopov}})$
fašizam	0.999851966871
stići	0.999585185524
pleme	0.999424936843
provesti	0.999391698868
seljak	0.999349832501
istraga	0.999312696003
kriminalac	0.999312326054
familija	0.999211516005
šutjeti	0.999121898159
nećeš	0.999006579868
pljačka	0.998980015016

Kontinuirane vreće riječi, forum.hr.	
w	$\cos(\theta_{w,\text{lopov}})$
udobrovoljiti	0.986697051105
protestirati	0.976458774282
probisvjeti	0.973409591499
prevaren	0.970877439717
lopine	0.97022578622
lustrirati	0.969929168999
nagoditi	0.968235999039
hadezenjare	0.966689563846
hapsiti	0.964966894312
ulagivati	0.964278935858
dupelisci	0.961251398756

Usporedba konteksta, fHrWaC.	
w	$\cos(\theta_{w,\text{lopov}})$
održ	0.999790177584
ustvari	0.999573159923
kriti	0.999124583525
puštati	0.99902118875
straža	0.998905480836
plač	0.998899503849
seliti	0.998849638732
servirati	0.998794573602
obrazložiti	0.998775285514
pokvariti	0.998485865522
namjestiti	0.9984618786

Kontinuirane vreće riječi, fHrWaC.	
w	$\cos(\theta_{w,\text{lopov}})$
prevarantice	0.989873774261
žuljati	0.975462282171
ispljuskati	0.974456953709
strpati	0.973903079515
tovariti	0.970958606636
svodnik	0.970528690051
galamiti	0.970281325152
uspaničen	0.96887460789
okrasti	0.966868089764
vikati	0.964150890946
prići	0.963532667622

Tablica 3.1: Riječi najsličnije riječi "lopov", temeljem korpusa forum.hr i fHrWaC te dvije različite metode učenja vektora. Veći $\cos(\theta_{w,z})$ označava veću sličnost između vektora riječi.

Usporedba konteksta, forum.hr.	
w	$\cos(\theta_{w,\text{politika}})$
građanin	0.998171374917
odlučivati	0.997409847409
investicija	0.997380077275
ain	0.99687781209
toro	0.996366153658
seljak	0.995872786409
lopov	0.99531514364
fašizam	0.995281398643
vlada	0.994584216782
zanimljiv	0.994550571408
provesti	0.994107542335

Kontinuirane vreće riječi, forum.hr.	
w	$\cos(\theta_{w,\text{politika}})$
nametnut	0.992852476768
demokratskije	0.99000244433
demorkacije	0.989099549545
neutralnost	0.986156486986
parlamentarizam	0.985859681848
politak	0.984355078604
pasivnost	0.983240878489
nadnacionalan	0.978652214501
jednostran	0.978443671194
involviran	0.977519294566
predstavljati	0.977239102308

Usporedba konteksta, fHrWaC.	
w	$\cos(\theta_{w,\text{politika}})$
sektor	0.996149121962
objekt	0.996123654962
automobil	0.996072207558
zapošljavanje	0.995046324482
lak	0.994845364212
afganistanu	0.994746159449
mjera	0.994700225885
uplata	0.994393107354
osiguran	0.994294419391
recesija	0.994242734036
stajalište	0.99403169726

Kontinuirane vreće riječi, fHrWaC.	
w	$\cos(\theta_{w,\text{politika}})$
spremnost	0.985789032777
eurointegracijski	0.981192368374
klijentelistički	0.979326417608
kooperativnost	0.975464625919
politak	0.974092808798
zagovarati	0.974063835431
neutralnost	0.973617312028
ustrajanje	0.973236923516
podupirati	0.971441544062
ustrajavanje	0.969821343506
ustavnopravan	0.968888890655

Tablica 3.2: Riječi najsličnije riječi "politika", temeljem korpusa forum.hr i fHrWaC te dvije različite metode učenja vektora. Veći $\cos(\theta_{w,z})$ označava veću sličnost između vektora riječi.

Usporedba konteksta, forum.hr.

w	$\cos(\theta_{w,\text{odgovoriti}})$
objasniti	0.998526596533
izbor	0.997006327224
dokazati	0.996749693298
žele	0.996606586238
saznati	0.996015096036
pamet	0.995974574268
pametan	0.995736694552
sudjelovati	0.995549501305
branitelj	0.99501599945
epruveta	0.994898624489
činiti	0.994755481336

Kontinuirane vreće riječi, forum.hr.

w	$\cos(\theta_{w,\text{odgovoriti}})$
umeš	0.995787698611
hellaa	0.986549194051
replicirati	0.984826043071
trolam	0.98444984877
pojasniti	0.980812899699
prdiš	0.980233452518
brukaš	0.973167018021
supermoderator	0.971006064174
trolati	0.970815201616
trollati	0.9705000076
moderator	0.970214119245

Usporedba konteksta, fHrWaC.

w	$\cos(\theta_{w,\text{odgovoriti}})$
kojim	0.996859968548
uspjeti	0.993403107553
uspjeo	0.993248732855
neće	0.992689308888
zasigurno	0.99222239057
kojom	0.991973006245
odlučiti	0.991812927912
još	0.991603563905
smijati	0.991280727443
naglasiti	0.990600041146
izjaviti	0.990431786136

Kontinuirane vreće riječi, fHrWaC.

w	$\cos(\theta_{w,\text{odgovoriti}})$
protupitanje	0.981935314803
decidiran	0.976739183234
kosorica	0.974179254742
izjavljivati	0.973832450621
potvrdan	0.972377047182
kako	0.971330870094
kosorici	0.966387032238
prešutiti	0.964606667909
prigovoriti	0.96371258732
dometnuti	0.961599149418
smjeniti	0.960524801348

Tablica 3.3: Riječi najsličnije riječi "odgovoriti", temeljem korpusa forum.hr i fHrWaC te dvije različite metode učenja vektora. Veći $\cos(\theta_{w,z})$ označava veću sličnost između vektora riječi.

Usporedba konteksta, forum.hr.

w	$\cos(\theta_{w,\text{komunistički}})$
ova	0.999337318776
slovenski	0.999142574483
kila	0.999082149418
secular	0.99905212634
turistički	0.999051513665
opremljen	0.999035463486
znanstven	0.998965918867
involvement	0.998923192641
izraelski	0.998819365296
companion	0.998773993799
dugotrajan	0.998518951341

Kontinuirane vreće riječi, forum.hr.

w	$\cos(\theta_{w,\text{komunistički}})$
miloševićeve	0.994549058888
šurovanje	0.992177219107
zavnoha	0.989038240453
antifašistički	0.986460124603
kliki	0.985768121864
miloševićevoj	0.984896197668
ultranacionalistički	0.98319500911
eksponent	0.982955077918
nacifašizam	0.98272849956
miloševićevo	0.981964866127
sudioništvo	0.980227220686

Usporedba konteksta, fHrWaC.

w	$\cos(\theta_{w,\text{komunistički}})$
tenisačica	0.999668745804
bacanje	0.999431404407
iskusan	0.998769960529
leden	0.998674781333
nogometaša	0.998607915999
gust	0.998599130986
lički	0.998552887556
rimski	0.998392044984
kontra	0.998314280768
ružičast	0.998256613354
ribar	0.998044689594

Kontinuirane vreće riječi, fHrWaC.

w	$\cos(\theta_{w,\text{komunistički}})$
kontrarevolucija	0.992008763306
velikosrpski	0.99001371994
kvislinški	0.989424216085
osloboditeljski	0.986847442064
diktatura	0.986168607764
džihad	0.98518379568
vlašću	0.983418874415
dinastički	0.98216701445
fašistički	0.981599341343
presizanje	0.980593947862
monarhistički	0.980225061787

Tablica 3.4: Riječi najsličnije riječi "komunistički", temeljem korpusa forum.hr i fHrWaC te dvije različite metode učenja vektora. Veći $\cos(\theta_{w,z})$ označava veću sličnost između vektora riječi.

4. Klasifikacija sentimenta izraza

Najčešći pristup gradnji vektorskih reprezentacija višerječnih izraza jest primjena linearne kombinacije pojedinačnih vektorskih reprezentacija riječi poput sume ili aritmetičke sredine, kao u modelu kontinuiranih vreća riječi (v. odjeljak 3.3). Takav je pristup dobar u slučajevima kada je izraz moguće opisati doslovce kao skup svojih pojedinačnih riječi, no nema dobre rezultate kada se izrazi sastoje od riječi koje igraju ulogu operatora, mijenjajući značenje čitavog izraza ovisno o značenju riječi s kojom su upareni. Sentimenti izraza poput "užasno pametan" ne mogu se konzistentno određivati direktno putem linearne kombinacije riječi "užasno" i "pametan": one su pojedinačno negativnog, odnosno pozitivnog sentimenta, no uparene zajedno zajedno rezultiraju izrazom znatno pozitivnog sentimenta.

Potreban nam je model koji je sposoban naučiti razne načine na koje riječi mogu mijenjati značenje izraza u kojem se nalaze ovisno o riječima koje utječu na njih. Takav bi model naučio samo jednu reprezentaciju riječi "užasno", no klasificirao sentimente izraza "užasno pametan" i "užasno lijen" kao potpuno suprotne.

Ostatkom poglavlja opisuje se upravo takav jedan model, prilaže njegova implementacija, te se primjenjuje nad različitim skupovima parova riječi s ciljem ispravne klasifikacije njihovog sentimenta.

4.1. Nelinearni model

Bolji je pristup umjesto linearne kombinacije dvaju vektorskih reprezentacija riječi a i b koristiti nelinearnu. Težinska matricu W množi se vektorom čitavog izraza $\begin{bmatrix} a & b \end{bmatrix}^{\mathrm{T}}$, sačinjenog od spajanja vektorskih reprezentacija pojedinih riječi a i b. Na rezultat se zatim primjenjuje nelinearna funkcija g (npr. sigmoidalna), što rezultira vektorskom reprezentacijom čitavog izraza p.

$$p = g\left(W \begin{bmatrix} a \\ b \end{bmatrix}\right)$$

Upravo takav model pokazao se sposobnim naučiti mnogo veći raspon funkcija od obične linearne kombinacije (Socher et al., 2011), no još uvijek ne daje vrlo dobre rezultate. Jedan je problem zajednička težinska matrica W: linearnom kombinacijom svih mogućih parova riječi s jednom jedinom zajedničkom težinskom matricom teško je izraziti sve željene vektorske reprezentacije izraza sačinjenih od tih parova riječi. Koristiti samo takvu zajedničku težinsku matricu znači pretpostaviti da ona sadrži sve informacije o načinima na koji bilo koja riječ a utječe na bilo koju drugu b, i obrnuto.

Idealno, riječi bi utjecale jedne na drugu samo svojim parametrima, a ne globalnim parametrima zajedničkima za sve riječi: takav bi pristup pojedinačnim riječima omogućio mnogo veću sposobnost izražavanja različitih utjecaja nad različitim riječima. Upravo takav linearan model (Baroni i Zamparelli, 2010) pokazao se uspješnim prilikom rada nad vrlo jednostavnim izrazima, no u ovome radu koristi se model koji se bez izmjene može proširiti od parova bilo kakvih riječi do čitavih rečenica i dokumenata.

Ukoliko je neuronska mreža u potpunosti linearna (nad podatcima ne primjenjuje nelinearne funkcije), tada se ne dobiva ikakva korist od uvećavanja broja njenih slojeva. Više ulančanih slojeva takve neuronske mreže može se stopiti u samo jedan: linearna neuronska mreža, neovisno o broju slojeva, uvijek čini linearnu transformaciju ulaznih podataka, odnosno uči linearnu funkciju. Dubokim neuronskim mrežama teži se suprotnome: želi se naučiti kompleksna nelinearna funkcija koja mnogo bolje opisuje neki skup podataka. Takva prednost dolazi i s manom: slojevi nelinearnosti čine rezultirajuću funkciju nekonveksnom (Socher et al., 2012a). To znači da metode optimizacije parametara modela više ne garantiraju pronalazak globalnog optimuma: iste se metode (poput gradijentnog spusta) u takvom slučaju nastavljaju koristiti, no ovaj put bez garancija pronalaska stvarnog optimuma.

Bez predučenja, početne su parametri neuronske mreže prisiljeni biti postavljeni na nasumične vrijednosti. No, ukoliko umjesto nasumičnih koristimo prednaučene parametre, optimizacijski postupak vrlo nelinearne funkcije tada većom vjerojatnošću pronalazi bolji lokalni optimum. Predučenje se za duboke neuronske mreže tako pokazalo vrlo bitnim za postizanje dobrih rezultata (Erhan et al., 2010).

4.2. MV-RNN

Model MV-RNN (engl. *Matrix-Vector Recursive Neural Network*) (Socher et al., 2012b) koristi oba gore opisana pristupa: svakoj riječi pridodaje vektorsku reprezentaciju koja je opisuje te matricu koja opisuje njen utjecaj na sve druge riječi leksikona, a nad svakom linearnom kombinacijom dvaju takvih riječi primjenjuje nelinearnu funkciju. Takav postupak zatim rezultira novim vektorom i novom matricom grupnog izraza (istih dimenzija kao i početnih riječi) koji se mogu istom operacijom kombinirati s vektorskim reprezentacijama drugim riječima ili izraza.

Nastavkom poglavlja detaljno se opisuje model MV-RNN (v. odjeljak 4.2), uči ga se (v. odjeljak 4.2.2) nad tri različita skupa parova riječi (v. odjeljak 4.3) te prikazuje dobivene rezultate (v. odjeljak 4.4).

4.2.1. Model

Svakoj riječi leksikona, neovisno o njenoj vrsti, pridodaje se njen vlastiti kontinuirani vektor i matrica (u daljnjem tekstu njena MV-reprezentaciju, jednadžba 4.1). Dok njen vektor opisuje samu riječ, njena matrica opisuje način na koji ta riječ utječe na druge riječi. Mnoge riječi same po sebi ne izražavaju sentiment, no imaju velik utjecaj na sentiment drugih riječi (npr. "vrlo"): vektori takvih riječi u tom slučaju slobodni su optimizacijom parametara težiti potpuno neutralnom stanju, dok će matrice poprimati vrijednosti koje znatno mijenjaju sentiment cijelog izraza u određenom smjeru.

$$\begin{aligned} w &= (x, X) \\ x &\in \mathbb{R}^n \\ X &\in \mathbb{R}^{n \times n} \end{aligned} \tag{4.1}$$

Kao početne vrijednosti vektora $x \in L$ koriste se upravo prethodno naučene vektorske reprezentacije riječi (v. odjeljak 3.5) dimenzija $n = 8$. Matrice $X \in L_M$ se pak inicijaliziraju na jediničnu matricu kojoj se dodaju malene količine šuma, čime se čini pretpostavka da većina riječi u ulozi operatora uopće ne utječe na druge riječi: ne izmjenjuje im sentiment.

$$\begin{aligned} X &= I + \epsilon \\ \epsilon &\in \langle -0.01, 0.01 \rangle \end{aligned}$$

Isto kao i u slučaju predučenja vektora riječi, matrice riječi su istovremeno i značajke primjera za učenje i parametri modela. Naknadnim optimizacijskim postupkom, krenuvši od nasumičnih vrijednosti, one poprimaju sposobnost očekivanog utjecaja na druge riječi (prema označenim primjerima skupa za učenje).

Pošto model MV-RNN pretpostavlja rekurzivno ponavljanje istog postupka nad sve većim i većim izrazima (od sentimenta para riječi do sentimenta rečenice ili dokumenta), cijeli izraz mora imati isti oblik svojstava kao i pojedinačne riječi: svoju MV-reprezentaciju. Na taj se način vektor p i matricu P nekog izraza može uspoređivati s onima pojedinačnih riječi: čitavi izrazi mogu imati svoj sentiment, ali i način utjecaja na druge riječi, odnosno izraze. MV-reprezentaciju izraza sačinjenog od dvije riječi $\alpha = (a, A)$ i $\beta = (b, B)$ stoga se računa idućim kombinacijskim funkcijama (Socher et al., 2012b):

$$p_{\alpha,\beta} = f_{A,B}(a, b) = f(Ba, Ab) = g\left(W \begin{bmatrix} Ba \\ Ab \end{bmatrix}\right) \tag{4.2}$$

$$p_{\alpha,\beta} \in \mathbb{R}^n$$

Funkcija g pritom je neka nelinearna funkcija (u ovom radu koristi se sigmoidalna) koja će nam omogućiti učenje mnogo većeg raspona funkcija no što bi nam dozvolila obična linearna kombinacija. Težinska matrica $W \in \mathbb{R}^{n \times 2n}$ globalna je matrica zajednička za sve izraze koja pretvara $[Ba\ Ab]^{\mathrm{T}} \in \mathbb{R}^{2n}$ natrag u vektor dimenzija $\mathbb{R}^n$, spreman za rekurzivnu primjenu iste funkcije nad novom MV-reprezentacijom izraza. Matrica izraza računa se na sličan način:

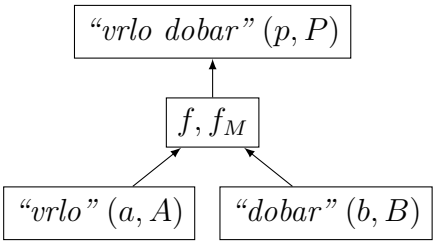
$$P_{\alpha,\beta} = f_M(A, B) = W_M \begin{bmatrix} A \\ B \end{bmatrix} \tag{4.3}$$

$$P_{\alpha,\beta} \in \mathbb{R}^{n \times n}$$

$$W_M \in \mathbb{R}^{n \times 2n}$$

Primjena rekurzivnih operacija nad MV-reprezentacijama čitavih izraza obrađena je u (Socher et al., 2012b) s dobrim rezultatima, a posebno je opisana u odjeljku 4.2.4. Pošto je fokus rada na na izraze sačinjene od dvije riječi, matrica P, odnosno funkcija f_m, trenutno nisu potrebne.

Nakon što je izračunata vektorska reprezentacija jednog izraza, računa se procjena njegove distribucije sentimenta po odabranom broju klasa. Radi lakše usporedbe s rezultatima Socher et al. (2012b), odabrano je $K = 10$ mogućih



Slika 4.1: MV-reprezentacija izraza ovisi o MV-reprezentacijama njegovih dijelova.

klasa sentimenta: od prve najnegativnije do desete najpozitivnije. Distribucija $d(p)$ sentimenta vektorske reprezentacije izraza p po svim klasama tada iznosi:

$$\begin{aligned} d(p) &= \text{softmax}\left(W_c\ p\right) \\ W_c &\in \mathbb{R}^{K \times n} \\ d(p) &\in \mathbb{R}^K \end{aligned} \tag{4.4}$$

Funkcija *softmax* rezultira probabilističkom interpretacijom pripadnosti izraza klasama sentimenta. Svakoj klasi dodijeljena je neka vjerojatnost $d(p)_i$ takva da vrijedi $\sum_{i=1}^{K} d(p)_i = 1$.

$$\text{softmax}_i(x) = \frac{e^{x_i}}{\sum_{i=j}^{n} e^{x_j}}$$

Pošto se svaka vrijednost vektora x_i koristi kao eksponent, funkcija *softmax* preuveličava udio onih vrijednosti koje su znatno bliže maksimalnoj u vektoru (odnosno, za takve vrijednosti teži 1), odnosno umanjuje udio onih koje znatno odstupaju od maksimuma (odnosno, teži 0).

4.2.2. Učenje

Takva rezultirajuća distribucija zatim se uspoređuje sa stvarnom, računajući razliku, odnosno grešku između njih. Radi potrebe učenja modela greška između dvaju distribucija računa se srednjom vrijednošću unakrsne entropije između svih elemenata dvaju vektora, a konačni rezultati prikazuju se koristeći KL-divergenciju.

$$E(x, y) = -\frac{1}{K} \sum_{i=1}^{K} \left(x_i \ln y_i + (1 - x_i) \ln(1 - y_i)\right)$$

Konačna funkcija koju se minimizira optimizacijskim postupkom tako je srednja vrijednost grešaka za svih N primjera skupa za učenje:

$$J = \frac{1}{N} \sum_{i=1}^{N} E\left(x^{(i)}, y^{(i)}\right) \tag{4.5}$$

KL-divergencija

Kullback-Leiblerova divergencija ili KL-divergencija mjera je razlike između dvije probabilističke distribucije P i Q, gdje P određuje stvarnu distribuciju, a Q njenu procjenu na temelju modela. KL-divergenciju distribucije Q od distribucije P označujemo s $D_{KL}(P||Q)$ i ona iznosi

$$D_{KL}(P||Q) = \sum_{i=1}^{n} \ln\left(\frac{P_i}{Q_i}\right) P_i, \quad D_{KL}(P||Q) \geq 0.$$

Distribucije su sličnije što je KL-divergencija bliža nuli. KL-divergencija nije metrika, jer je nesimetrična i ne zadovoljava svojstvo nejednakosti trokuta. No, pošto ipak zadovoljava neka svojstva metrike, poput jednakosti samo u slučaju kada su obje distribucije jednake, korisna je kao mjera greške između distribucija (Gray, 1990), za što je u ovome radu i koristimo.

4.2.3. Implementacija

Učenje modela provodi se stohastičkim gradijentnim spustom za svaku kombinaciju izvora početnih vektora riječi. Vrši se 300 iteracija kroz sve primjere skupa za učenje, pritom umanjujući početnu stopu učenja $\alpha = 0.1$ linearno prema nuli.

Primjer izvršavanja

Prije učenja modela, potrebno je instalirati sve potrebne alate i biblioteke kao i za metodu usporedbe konteksta navedene u odjeljku 3.2.3. Program se pokreće nad željenim leksikonom, prednaučenim vektorima riječi, datotekom sa skupom za učenje i datotekom s lemama čitavog korpusa.

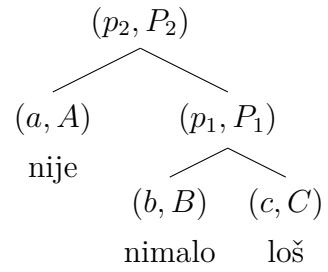
```
$ ./train.py words.txt vectors.npy examples.txt lemmas.txt
took 1303 training examples out of 1564 total
testset  kl before training is 1.004 +/- 0.440
trainset kl  after training is 0.016 +/- 0.016
testset  kl  after training is 0.022 +/- 0.022
```

Nakon završetka procesa, optimalni su parametri pohranjeni u datotekama u lokalnom direktoriju. Čitav program priložen je radu, a odabrani dijelovi navedeni su i u dodatku B.

4.2.4. Primjena na rečenice i veće dijelove teksta

Postupak klasifikacije sentimenta može se rekurzivno primjenjivati od manjih izraza prema sve većim. Ključ je u reprezentaciji višerječnih izraza: ona je iste dimenzionalnosti kao i reprezentacija samih riječi. Uzmimo za primjer izraz "nije nimalo loš" (slika 4.2).

Primjenjujući nad MV-reprezentacijama riječi "nimalo" (b, B) i "loš" (c, C) operacije tvorbe MV-reprezentacije višerječnog izraza (jednadžbe 4.2 i 4.3), dobiva se MV-reprezentacija (p_1, P_1) potrebna za računanje reprezentacije čitavog izraza (p_2, P_2). Postupak je identičan za bilo koji par riječi, odnosno višerječnih izraza.



Slika 4.2: Primjer binarnog stabla parsanja za izraz "nije nimalo loš".

Model se uči slično kao i u slučaju s dvorječnim izrazima, samo što se sada umjesto stvarne distribucije sentimenta dvorječnog izraza radi sa stvarnom distribucijom sentimenta svih čvorova u stablu parsanja. Za svako stablo se prvo računaju MV-reprezentacije svih čvorova (p_i, P_i), krenuvši od lišća prema korijenu. Nad svakim čvorom procjenjuje se distribucija sentimenta primjenom funkcije *softmax* te računaju njene derivacije po svim parametrima od korijena prema lišću, što se može učiniti na efikasan način metodom propagacije unatrag opisanom u (Rumelhart et al., 1986) i (Goller i Kuchler, 1996).

$$J = -\sum_{t} \sum_{a \in t} E\left(x^{(a)}, y^{(a)}\right)$$

Pošto se svaki primjer za učenje ovaj put sastoji od čitavog stabla, greška se mjeri zbrajajući greške između stvarne i procijenjene distribucije sentimenta za svaki pojedinačni čvor stabla. Za svako stablo t i čvor $a \in t$ vrši se zbroj greške E definirane za slučaj dvorječnog izraza (v. odjeljak 4.2.2), što upravo predstavlja funkciju J koju želimo minimizirati optimizacijskim postupkom. Ta funkcija nije konveksna, no u praksi joj se nalazi dobar lokalni minimum (Socher et al., 2012b).

4.3. Skupovi za učenje

Model klasifikacije sentimenta evaluira se nad tri skupa za učenje različitih veličina, kompleksnosti i razina šuma: nad umjetno stvorenim skupom parova prilog-pridjev, nad dijelom skupa parova riječi iz recenzija filmova prevedenih s engleskog na hrvatski jezik, te nad parovima riječi dohvaćenih iz recenzija restorana na hrvatskom jeziku.

Svaki skup za učenje sastoji se od niza parova dvaju riječi i njihove distribucije vjerojatnosti pripadanja određenoj klasi sentimenta. Pripadnost je podijeljena na 10 različitih klasa koje predstavljaju razine sentimenta, od najnegativnijeg (1) do najpozitivnijeg (10).

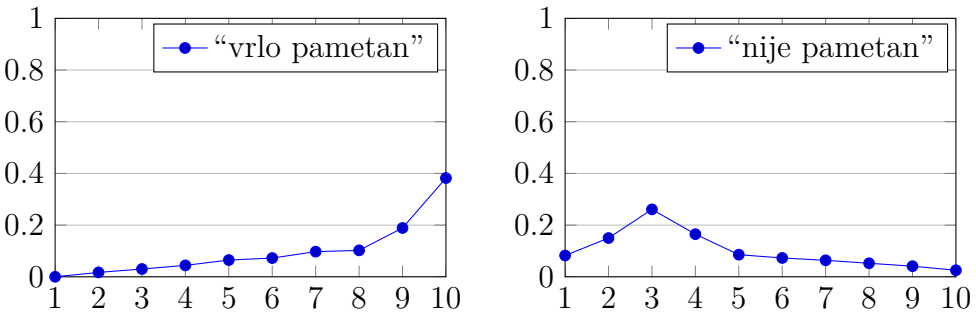
$$\begin{array}{rl} \textit{“jako dobar”} & \begin{bmatrix} 0.01 & 0.01 & 0.02 & 0.04 & 0.07 & 0.09 & 0.11 & 0.15 & 0.19 & 0.31 \end{bmatrix} \\ \textit{“baš loš”} & \begin{bmatrix} 0.23 & 0.32 & 0.16 & 0.10 & 0.07 & 0.04 & 0.03 & 0.02 & 0.02 & 0.01 \end{bmatrix} \end{array}$$

Skup za učenje nikada ne sadrži sentiment pojedinih riječi. Modelu se takva informacija nikada ne predaje eksplicitno. Modelu su poznate samo vjerojatnosti sentimenta čitavih parova riječi, temeljem kojih on implicitno uči sentiment pojedinih riječi tog para (te način na koji se taj sentiment mijenja kada je riječ u kombinaciji s drugom). Svi korišteni skupovi za učenje detaljnije se opisuju u nastavku.

4.3.1. Simuliran skup parova riječi izvan konteksta

Ovaj skup sastavljen je od 1560 parova riječi, sastavljenih od svih kombinacija 26 različitih priloga i negacija te 60 različitih pridjeva pozitivnog, odnosno negativnog sentimenta. Prilozi se pritom sastoje od onih koji najčešće uvećavaju sentiment izraza (poput “jako” ili “nevjerojatno”) te onih koji ga umanjuju (poput “možda” ili “donekle”).

Različite uparene riječi označene su distribucijama sentimenta ručno, koristeći priložen program. Pritom je sentiment izraza odabran ne promatrajući ikakav kontekst tog izraza. Odabranoj klasi sentimenta dodijeljena je najveća vjerojatnost, dok je ostalim pridodana simulirana nasumična vrijednost koja teži nuli što je klasa udaljenija od one primarno odabrane. Takav postupak rezultira primjerima za učenje koji imaju samo jedan očit najvjerojatniji sentiment: skup za učenje ne sadrži primjere čiji je sentiment dvosmislen. Slika 4.3 sadrži neke primjere izraza takvog skupa za učenje.



Slika 4.3: Neki primjeri parova iz simuliranog skupa za učenje. Točke prikazuju distribuciju vjerojatnosti pripadanja klasama sentimenta.

Pošto je ovakav skup vrlo jednostavan, repetitivan i “čist”, te pošto se svaka pojedina riječ u njemu pojavljuje u barem 26 različitih parova riječi, za očekivati je kako će model nad njim imati odlične rezultate (v. tablicu 4.1).

4.3.2. Skup prevedenih parova riječi iz recenzija filmova

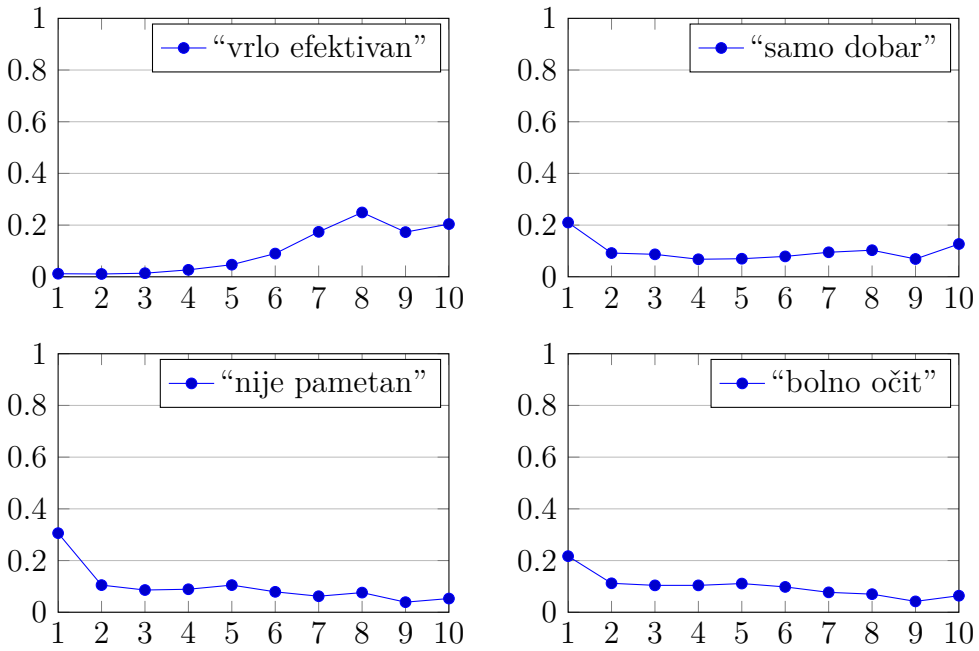
Web-stranica *imdb.com* (engl. *International Movie Database*) sadrži, među ostalim, mnoge korisničke recenzije filmova. Svaka korisnička recenzija napisana je na engleskom jeziku i uparena s ocjenom filma od 1 do 10. Preuzet je javno dostupan skup parova riječi dohvaćenih iz takvih recenzija[1] te su zatim ručno na hrvatski prevedene samo one koje se unutar skupa svih recenzija pojavljuju barem 300 puta.

Uz svaki takav par povezane su količine njegovog pojavljivanja u recenzijama svakih mogućih ocjena. Takve količine direktno se pretvara o očekivane distribucije vjerojatnosti pripadnosti klasama sentimenta.

$$\begin{array}{rl} \textit{“very watchable”} & \begin{bmatrix} 18 & 17 & 21 & 45 & 91 & 243 & 419 & 326 & 133 & 130 \end{bmatrix} \\ \textit{“vrlo gledljiv”} & \begin{bmatrix} 0.01 & 0.01 & 0.02 & 0.03 & 0.06 & 0.17 & 0.29 & 0.23 & 0.09 & 0.09 \end{bmatrix} \end{array}$$

Rezultirajući skup sastoji se od 1026 različitih parova između 208 jedinstvenih riječi te nije direktna preslika podskupa verzije na engleskome jeziku, pošto mnogi parovi različiti na engleskom jeziku postaju identični nakon prijevoda na hrvatski. Takvi višestruki identični primjeri spajaju se u jedan sumirajući si količine pojavljivanja u recenzijama različitih ocjena. Slika 4.4 sadrži neke primjere parova ovog skupa za učenje.

[1] http://compprag.christopherpotts.net/iqap-experiments.html



Slika 4.4: Neki primjeri skupa za učenje temeljenog na prevedenim parovima preuzetim iz recenzija filmova. Točke prikazuju distribuciju vjerojatnosti pripadanja klasama sentimenta.

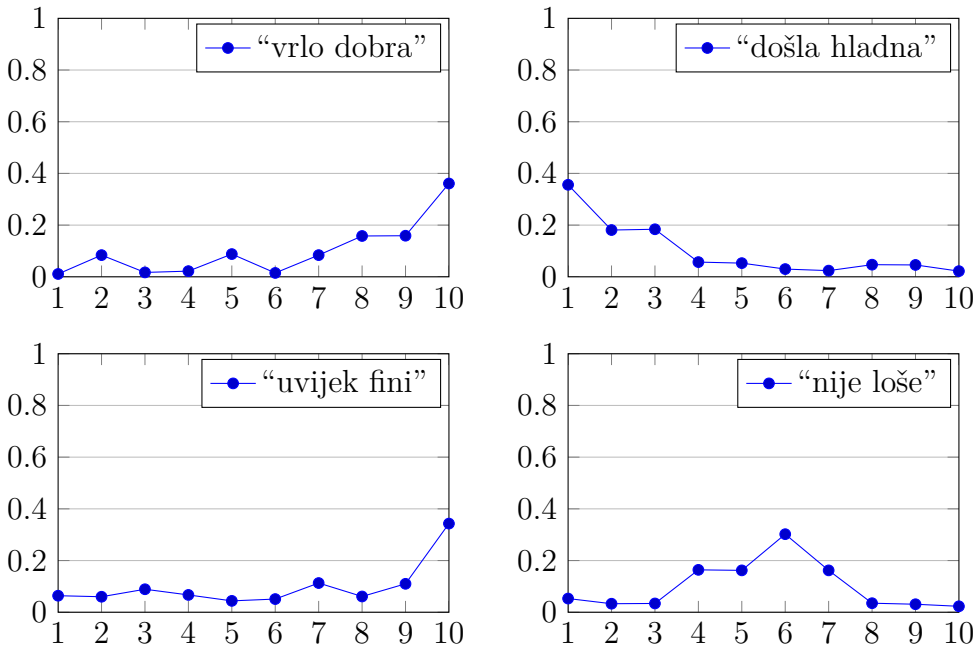
Evaluacija modela MV-RNN provedena je nad ovim skupom (na engleskome jeziku) u radu (Socher et al., 2012b), čije performanse zbog sličnosti skupa onome prevedenom na hrvatski jezik očekujemo replicirati (v. tablicu 4.1).

4.3.3. Skup parova riječi iz recenzija restorana

Koristi se javno dostupan skup unaprijed jezično obrađenih recenzija restorana preuzetih s web-stranice *pauza.hr*[2] (Glavaš et al., 2013). Recenzije su sintaktički obilježene, stoga se iz njih priloženim programom preuzimaju parovi uzastopnih priloga, pridjeva i glagola, zajedno s ocjenom recenzije u kojoj se nalaze. Grupiranjem identičnih parova dobiva se skup od 1146 jedinstvenih parova i količina njihovih pojavljivanja u recenzijama svih mogućih ocjena, čime se ujedno dobivaju i njihove distribucije vjerojatnosti pripadnosti klasama sentimenta. Slika 4.5 sadrži neke primjere parova ovog skupa za učenje.

Za razliku od prethodno opisanih recenzija filmova, recenzija restorana ima mnogo manje. Stoga je i konačan skup parova riječi mnogo manji. Mnogi parovi se u recenzijama pojavljuju rijetko, a isto vrijedi i za riječi od kojih se sastoje

[2] http://takelab.fer.hr/data/cropinion/



Slika 4.5: Neki primjeri skupa za učenje temeljenog na parovima preuzetim iz recenzija restorana. Točke prikazuju distribuciju vjerojatnosti pripadanja klasama sentimenta.

(do najmanje tri puta). Stoga se prilikom evaluacije očekuju lošije performanse modela nad ovim skupom (v. tablicu 4.1).

4.4. Evaluacija

Parametri modela MV-RNN uče za svaku kombinaciju korištenog korpusa (korpusi forum.hr i fHrWaC), prednaučenih vektorskih reprezentacija riječi (modelima usporedbe konteksta i kontinuiranih vreća riječi) i skupa parova riječi (skupovi simuliranih bez konteksta, prevedenih iz recenzija filmova te direktno preuzetih iz recenzija restorana).

Za svaku od tih 12 kombinacija u tablici 4.1 navode se performanse konačnog modela MV-RNN nad skupom za testiranje. Skup se za testiranje u svim slučajevima sastoji u potpunosti od izraza koji se ne nalaze istodobno i u skupu za učenje: svaki izraz skupa za testiranje naučenom modelu je potpuno nov. No, riječi koje čine taj izraz najčešće su dio različitih izraza i u skupu za učenje.

Za svaku od 12 kombinacija odabrane su procjene distribucije vjerojatnosti sentimenta za 4 različitih izraza unutar skupa za testiranje i prikazane zajedno sa stvarnim distribucijama tih primjera, kao i s KL-divergencijom kao mjerom

Predučenje	Simulirani	Recenzije filmova	Recenzije restorana
CBOW/fHrWaC	$\mathbf{0.024 \pm 0.022}$	$\mathbf{0.067 \pm 0.111}$	$\mathbf{0.299 \pm 0.217}$
CBOW/forum.hr	0.027 ± 0.021	0.075 ± 0.132	0.332 ± 0.234
Collobert/fHrWaC	0.034 ± 0.033	0.082 ± 0.144	0.329 ± 0.254
Collobert/forum.hr	0.066 ± 0.136	0.089 ± 0.16	0.381 ± 0.283

Tablica 4.1: Prosječne performanse modela za klasifikaciju sentimenta izraza (izražene KL-divergencijom, v. odjeljak 4.2.2) za svaku kombinaciju metode predučenja vektorskih reprezentacija riječi, korištenog korpusa rečenica i skupa parova riječi. Istaknute su najbolje performanse za svaki od skupova parova riječi.

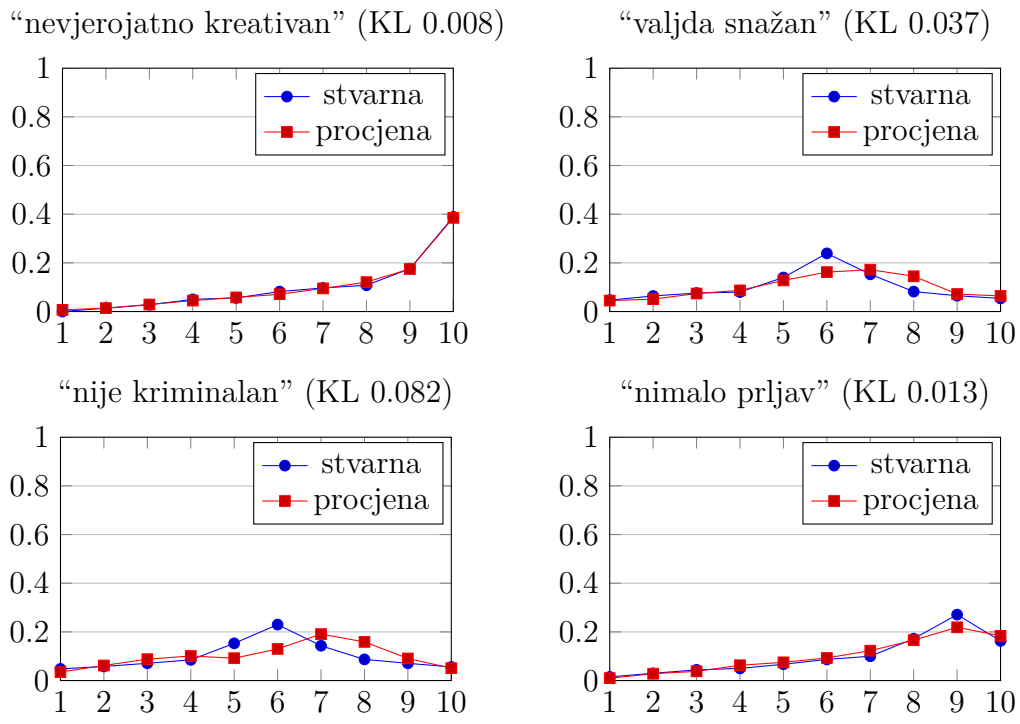
greške, slikama u ostatku poglavlja.

Rezultati za skup simuliranih parova očekivano su odlični: svaka se riječ skupa pojavljuje u mnoštvu slučajeva uparena sa svakom drugom riječi i distribucije vjerojatnosti sentimenta vrlo su ujednačene i jednostavne (nedostaju dvosmisleni izrazi, izrazi istovremeno vjerojatno pozitivnog i negativnog sentimenta).

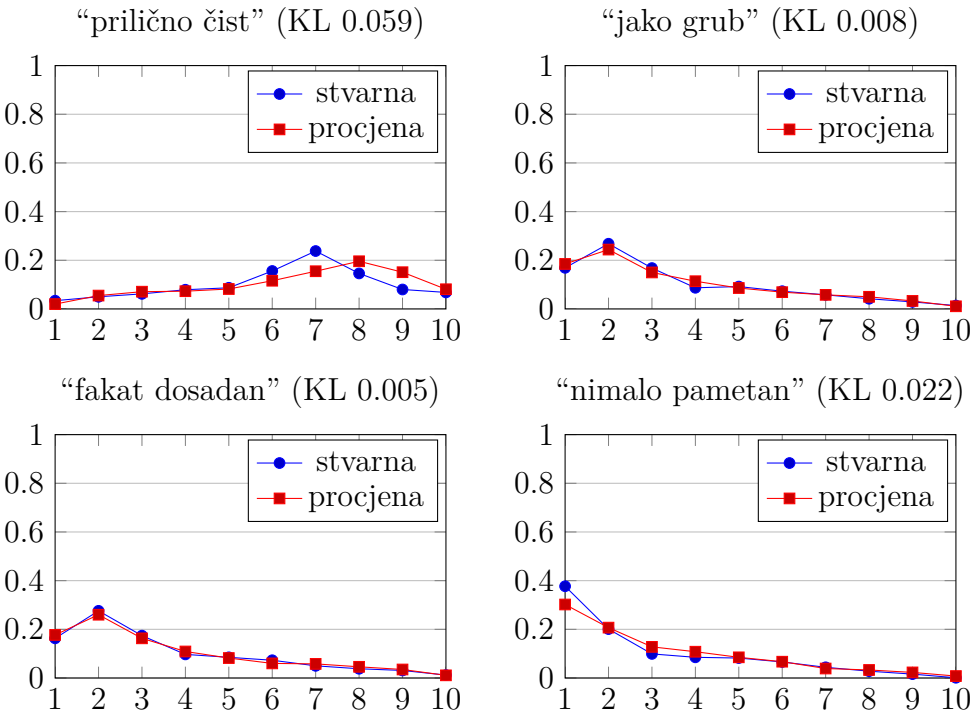
Uspoređujući rezultate s onima iz (Socher et al., 2012b) za slučaj skupa za učenje temeljenog na recenzijama filmova, gdje je koristeći predučenje metodom usporedbe konteksta prijavljena prosječna KL-divergencija nad skupom za testiranje od 0.091, dobivaju se približno iste iznosi prosječne greške.

Evaluacija nad skupom temeljenom na recenzijama restorana rezultirala je mnogo većim greškama. Pošto je skup sastavljen na temelju mnogo manje količine tekstova, sastoji se od izraza s vrlo niskim prosječnim ponavljanjima riječi i vrlo dvosmislenim te "oštrim" distribucijama sentimenta, gdje procjena klase sentimenta najveće vjerojatnosti samo za jednu klasu udaljena od stvarne rezultira visokom KL-divergencijom između dvaju distribucija.

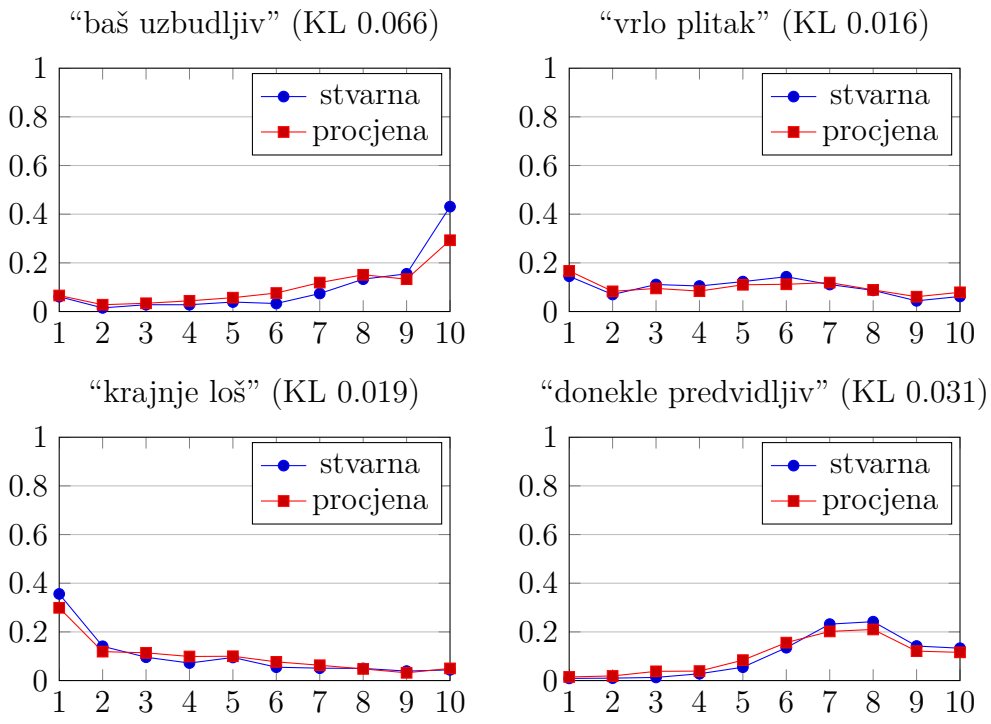
Na koncu, rezultati pokazuju da je, uz dovoljno dobar skup za učenje, model sposoban naučiti različita operatorska značenja priloga i negacija te sentiment parova prilog-pridjev, odnosno negacija-pridjev, bez prethodne informacije o sentimentu samog pridjeva. Model ispravno prepoznaje da prilozi poput "nevjerojatno" ili "užasno" snažno pojačavaju sentiment, dok drugi poput "valjda" ili "donekle" očekivano rezultiraju blažim pojačanjem, neovisno o tome je li pridjev sam po sebi pozitivnog ili negativnog sentimenta. Također, model korektno mijenja predznak sentimenta pridjeva ukoliko je uparen s negacijom poput "nimalo" ili "nije", pritom ne zanemarujući da "nimalo" vrši snažniju negaciju.



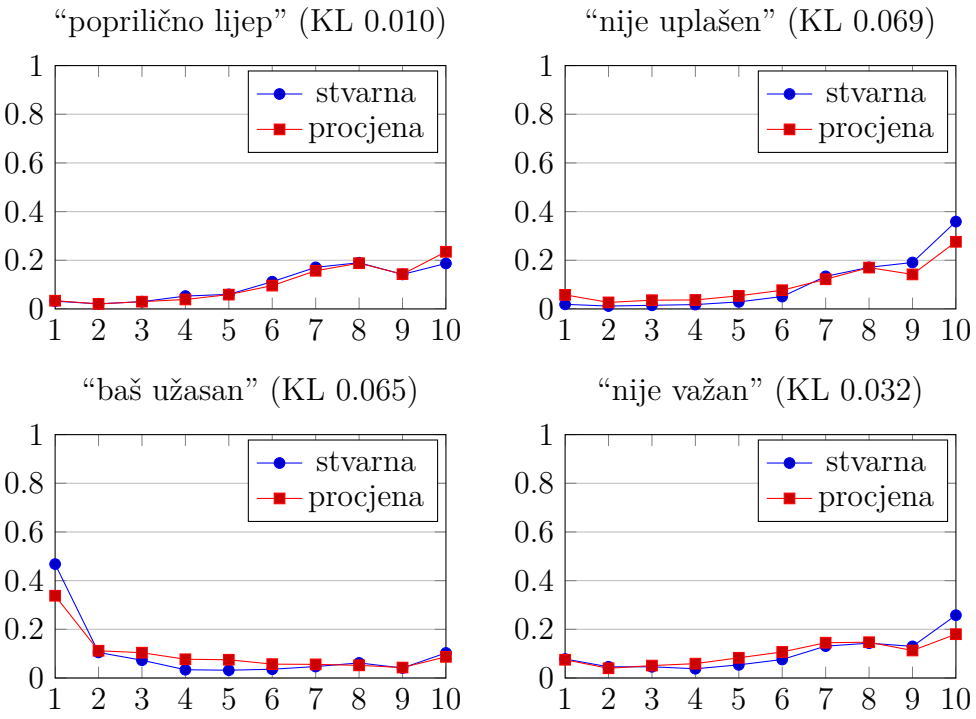
Slika 4.6: Evaluacija dijela testnog skupa simuliranih parova; predučenje modelom CBOW nad fHrWaC. Prosječna KL-divergencija testnog skupa iznosi 0.024 ± 0.022.



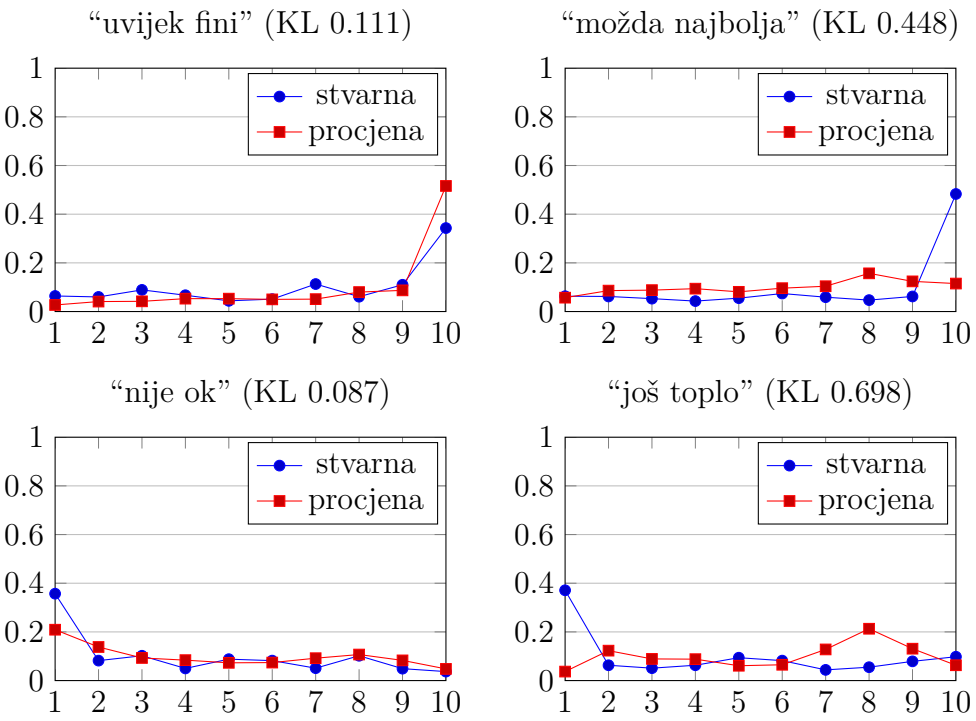
Slika 4.7: Evaluacija dijela testnog skupa simuliranih parova; predučenje modelom CBOW nad forum.hr. Prosječna KL-divergencija testnog skupa iznosi 0.027 ± 0.021.



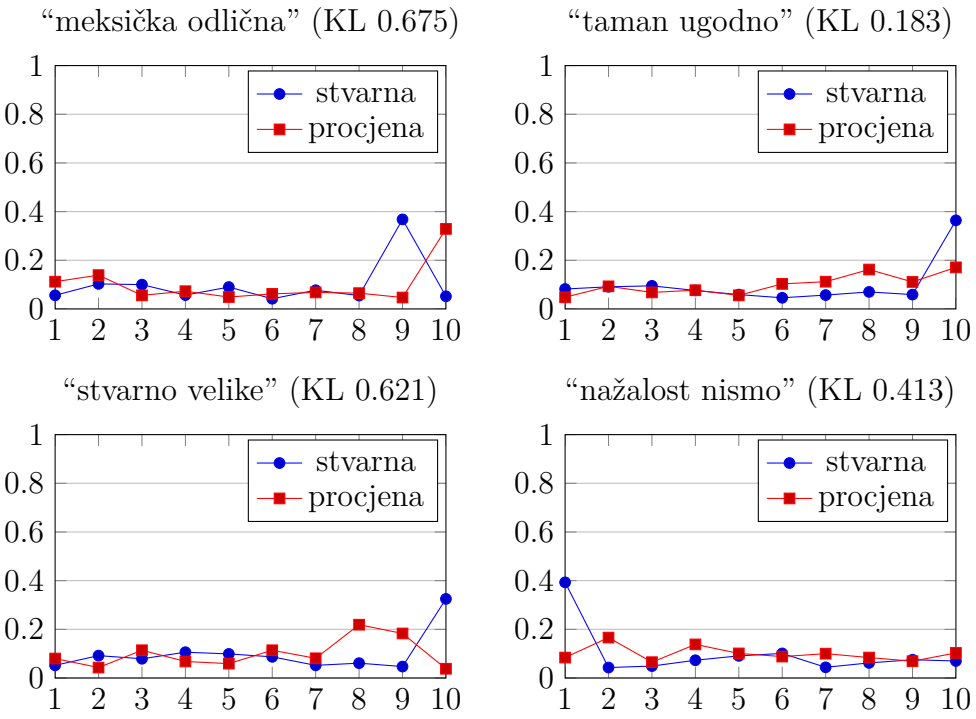
Slika 4.8: Evaluacija dijela testnog skupa recenzija filmova; predučenje modelom CBOW nad fHrWaC. Prosječna KL-divergencija testnog skupa iznosi 0.067 ± 0.111.



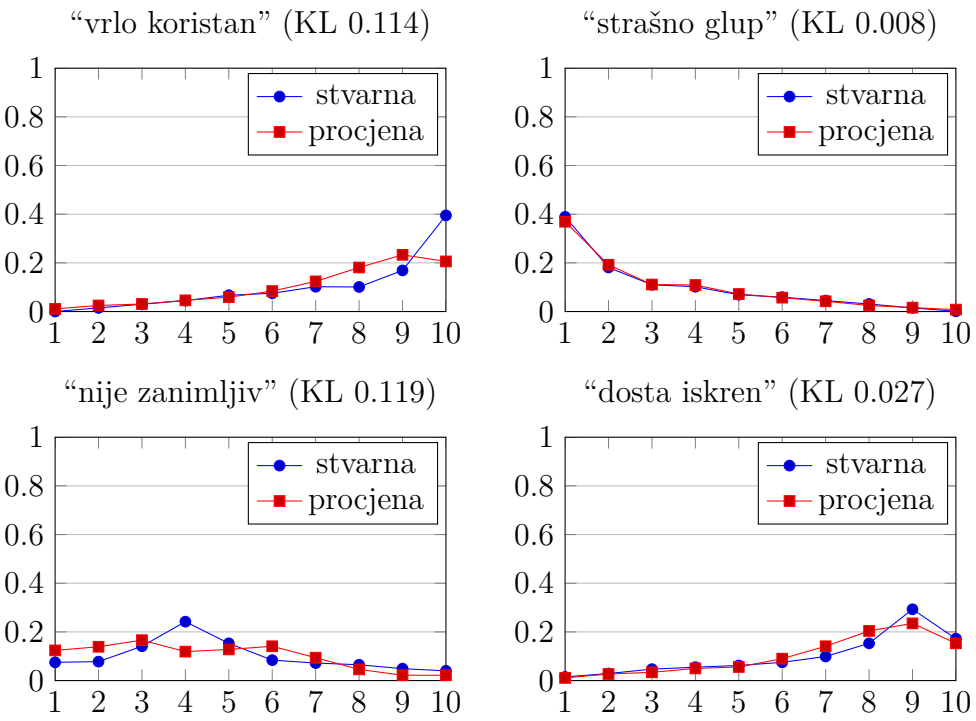
Slika 4.9: Evaluacija dijela testnog skupa recenzija filmova; predučenje modelom CBOW nad forum.hr. Prosječna KL-divergencija testnog skupa iznosi 0.075 ± 0.132.



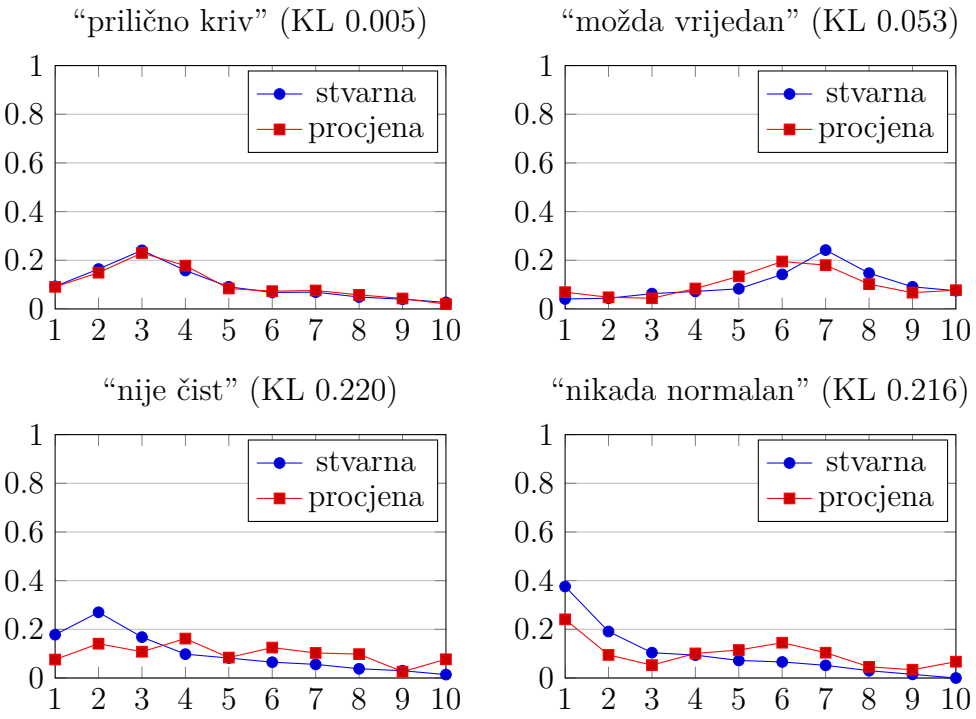
Slika 4.10: Evaluacija dijela testnog skupa recenzija restorana; predučenje modelom CBOW nad fHrWaC. Prosječna KL-divergencija testnog skupa iznosi 0.299 ± 0.217.



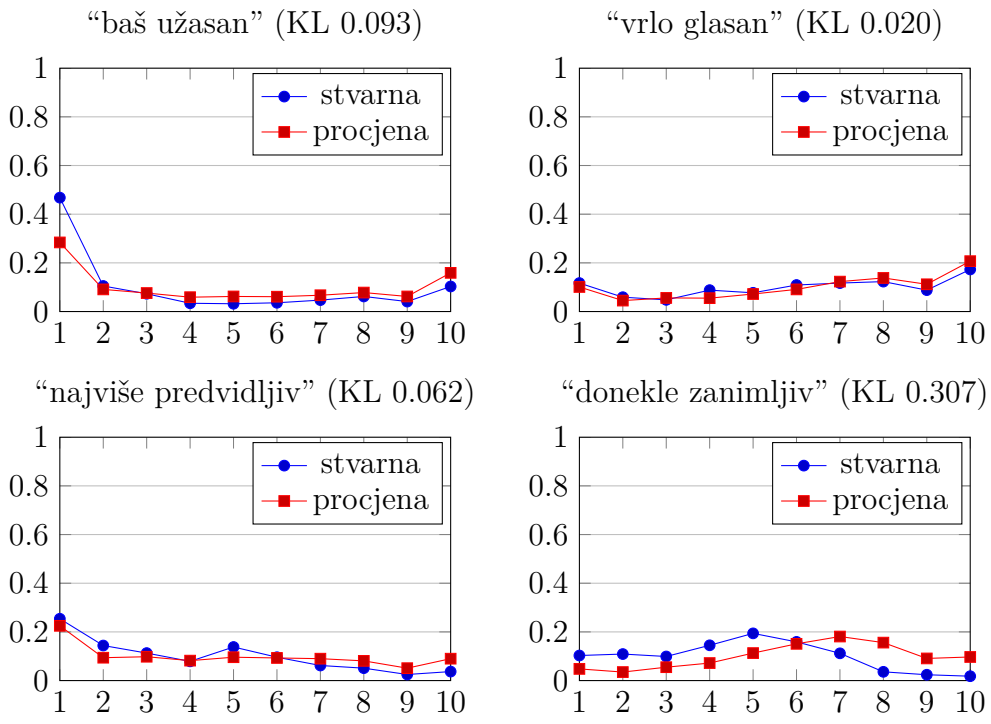
Slika 4.11: Evaluacija dijela testnog skupa recenzija restorana; predučenje modelom CBOW nad forum.hr. Prosječna KL-divergencija testnog skupa iznosi 0.332 ± 0.234.



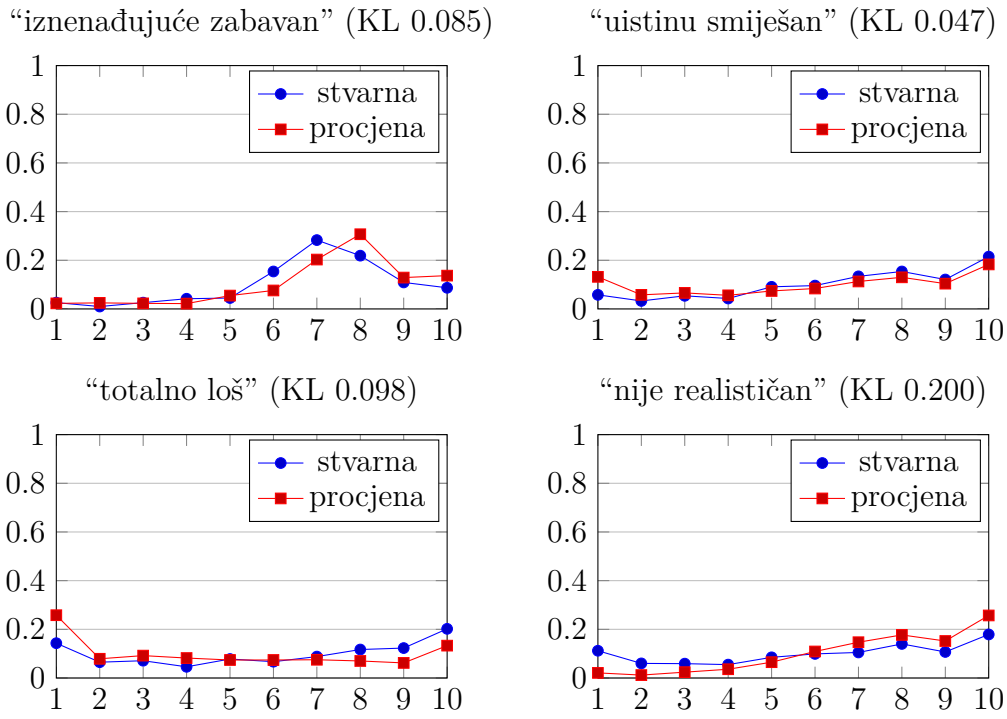
Slika 4.12: Evaluacija dijela testnog skupa simuliranih parova; predučenje usporedbom konteksta nad fHrWaC. Prosječna KL-divergencija testnog skupa je 0.034 ± 0.033.



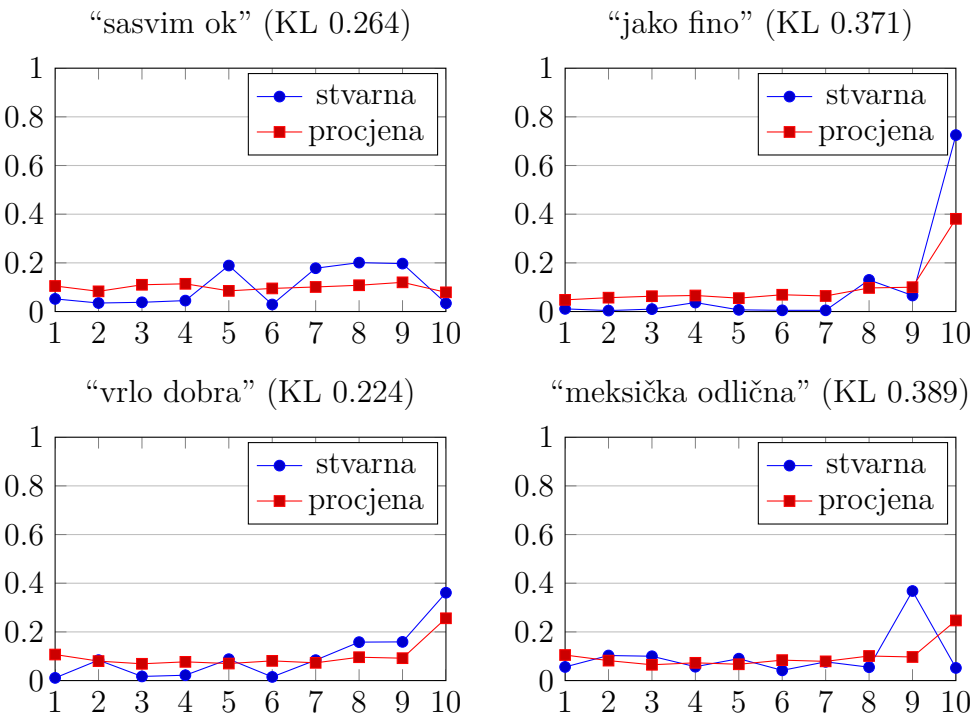
Slika 4.13: Evaluacija dijela testnog skupa simuliranih parova; predučenje usporedbom konteksta nad forum.hr. Prosječna KL-divergencija testnog skupa je 0.066 ± 0.136.



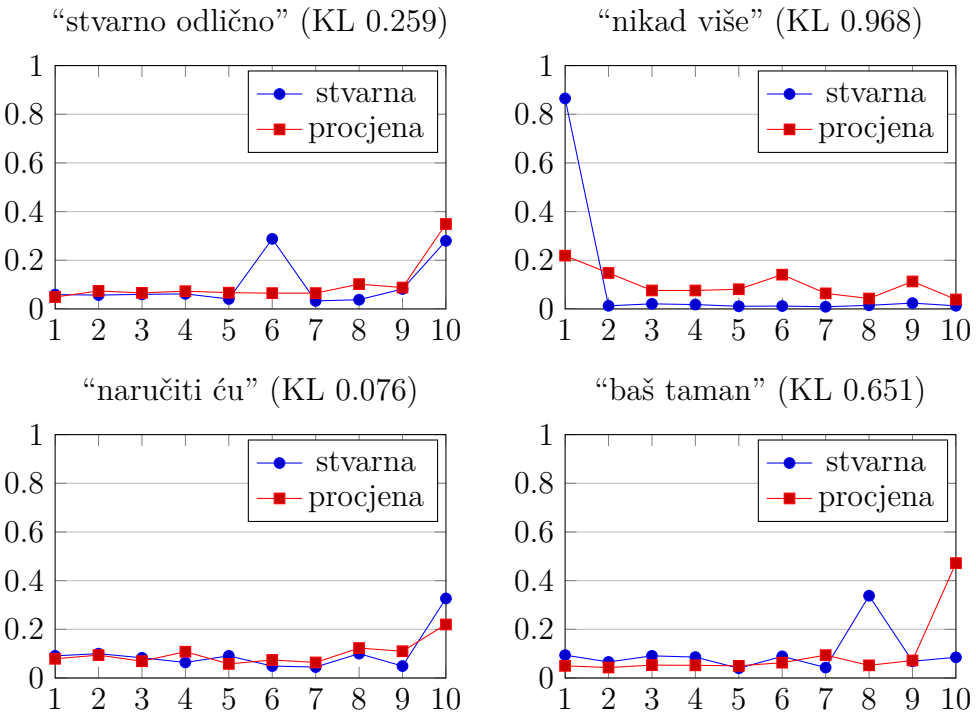
Slika 4.14: Evaluacija dijela testnog skupa recenzija filmova; predučenje usporedbom konteksta nad fHrWaC. Prosječna KL-divergencija testnog skupa je 0.082 ± 0.144.



Slika 4.15: Evaluacija dijela testnog skupa recenzija filmova; predučenje usporedbom konteksta nad forum.hr. Prosječna KL-divergencija testnog skupa je 0.089 ± 0.16.



Slika 4.16: Evaluacija dijela testnog skupa recenzija restorana; predučenje usporedbom konteksta nad fHrWaC. Prosječna KL-divergencija testnog skupa je 0.329 ± 0.254.



Slika 4.17: Evaluacija dijela testnog skupa recenzija restorana; predučenje usporedbom konteksta nad forum.hr. Prosječna KL-divergencija testnog skupa je 0.372 ± 0.317.

5. Zaključak

Analiza sentimenta je postupak kojim se na temelju teksta pokušava odrediti polaritet stava, emocionalnog stanja ili osjećaja koji se kroz taj tekst iznose. Postupci analize sentimenta zasnovani na rječniku apriornog sentimenta nailaze na problem loše klasifikacije sentimenta višerječnih izraza. Rješenju se pristupilo definicijom i implementacijom modela za klasifikaciju sentimenta višerječnih izraza temeljenog na rekurzivnim neuronskim mrežama prema radu (Socher et al., 2012b).

Provedena je evaluacija modela nad tri skupa za učenje različitih razina kompleksnosti. Primijenjen nad dovoljno dobrim skupom za učenje, model uspješno savladava problem neispravnih klasifikacija sentimenta višerječnih izraza: bez ikakvog apriornog znanja o pojedinim riječima, model je u stanju odrediti distribuciju vjerojatnosti pripadnosti klasama sentimenta za dvorječne izraze. Uz dovoljno sentimentom označenih dvorječnih izraza kao primjera za učenje, sentiment pojedinih riječi i načini na koje određene riječi utječu na druge model uči implicitno kroz parametre svojstvene svakoj riječi leksikona.

Performanse modela pospješene su postupkom predučenja: početne vektorske reprezentacije riječi određene su kombinacijom modela prema (Collobert et al., 2011) i (Mikolov et al., 2013a) te korpusa rečenica na hrvatskom jeziku. Rezultirajuće reprezentacije riječi imaju vidljiv pozitivan utjecaj na performanse modela.

Kao budući rad, bilo bi zanimljivo primijeniti model nad izrazima većim od dvorječnih, idealno čitavim rečenicama ili odlomcima na hrvatskome jeziku. Opisani model MV-RNN postiže dobre rezultate u slučaju klasifikacije sentimenta čitavih rečenica na engleskom jeziku (Socher et al., 2012b) te je stoga potencijalno rješenje, no noviji i slični modeli koriste manje parametara, brže uče te istovremeno postižu bolje rezultate (Socher et al., 2013).

Literatura

Marco Baroni i Roberto Zamparelli. Nouns Are Vectors, Adjectives Are Matrices: Representing Adjective-noun Constructions in Semantic Space. U *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, stranice 1183–1193, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=1870658.1870773`.

Marco Baroni, Georgiana Dinu, i Germán Kruszewski. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 1, 2014.

Yoshua Bengio, Aaron C. Courville, i Pascal Vincent. Unsupervised Feature Learning and Deep Learning: A Review and New Perspectives. *CoRR*, abs/1206.5538, 2012.

Dan C. Ciresan, Ueli Meier, i Jürgen Schmidhuber. Multi-column Deep Neural Networks for Image Classification. *CoRR*, abs/1202.2745, 2012.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, i P. Kuksa. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, i Samy Bengio. Why Does Unsupervised Pre-training Help Deep Learning? *J. Mach. Learn. Res.*, 11:625–660, Ožujak 2010. ISSN 1532-4435.

Goran Glavaš, Damir Korenčić, i Jan Šnajder. Aspect-Oriented Opinion Mining from User Reviews in Croatian. U *51st Annual Meeting of the Association for Computational Linguistics*, stranica in press, 2013.

C. Goller i A. Kuchler. Learning task-dependent distributed representations by backpropagation through structure. U *Neural Networks, 1996., IEEE International Conference on*, svezak 1, stranice 347–352 vol.1, Jun 1996. doi: 10.1109/ICNN.1996.548916.

Robert M. Gray. *Entropy and Information Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1990. ISBN 0-387-97371-0.

Philippe Hamel, Simon Lemieux, Yoshua Bengio, i Douglas Eck. Temporal Pooling and Multiscale Learning for Automatic Annotation and Ranking of Music Audio. U *ISMIR*, stranice 729–734, 2011.

Alex Krizhevsky, Ilya Sutskever, i Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. U F. Pereira, C.J.C. Burges, L. Bottou, i K.Q. Weinberger, urednici, *Advances in Neural Information Processing Systems 25*, stranice 1097–1105. Curran Associates, Inc., 2012.

Honglak Lee, Roger Grosse, Rajesh Ranganath, i Andrew Y. Ng. Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations. U *Proceedings of the 26th International Conference on Machine Learning*, stranice 609–616, 2009.

Omer Levy i Yoav Goldberg. Linguistic Regularities in Sparse and Explicit Word Representations. U *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, stranice 171–180, Ann Arbor, Michigan, June 2014. Association for Computational Linguistics.

Nikola Ljubešić i Tomaž Erjavec. hrWaC and slWac: Compiling Web Corpora for Croatian and Slovene. U Ivan Habernal i Václav Matousek, urednici, *Text, Speech and Dialogue - 14th International Conference, TSD 2011, Pilsen, Czech Republic, September 1-5, 2011. Proceedings*, Lecture Notes in Computer Science, stranice 395–402. Springer, 2011.

Tomas Mikolov, Kai Chen, Greg Corrado, i Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *CoRR*, abs/1301.3781, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, i Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. U C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, i K.Q. Weinberger, urednici,

Advances in Neural Information Processing Systems 26, stranice 3111–3119. 2013b.

Tomas Mikolov, Wen tau Yih, i Geoffrey Zweig. Linguistic Regularities in Continuous Space Word Representations. U *HLT-NAACL*, stranice 746–751. The Association for Computational Linguistics, 2013c.

Richard Socher and Eric H. Huang and Jeffrey Pennington and Andrew Y. Ng and Christopher D. Manning. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. U *Advances in Neural Information Processing Systems 24*. 2011.

Salah Rifai, Yann N. Dauphin, Pascal Vincent, Yoshua Bengio, i Xavier Muller. The Manifold Tangent Classifier.

D. E. Rumelhart, G. E. Hinton, i R. J. Williams. Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1. poglavlje Learning Internal Representations by Error Propagation, stranice 318–362. MIT Press, Cambridge, MA, USA, 1986. ISBN 0-262-68053-X.

Frank Seide, Gang Li, i Dong Yu. Conversational Speech Transcription Using Context-Dependent Deep Neural Networks. U *Interspeech 2011*. International Speech Communication Association, August 2011.

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

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*, stranice 784–789, 2013.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, i Christopher D. Manning. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. U *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2011.

Richard Socher, Yoshua Bengio, i Christopher D. Manning. Deep Learning for NLP (Without Magic). U *Tutorial Abstracts of ACL 2012*, ACL '12, stranice 5–5, Stroudsburg, PA, USA, 2012a. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=2390500.2390505`.

Richard Socher, Brody Huval, Christopher D. Manning, i Andrew Y. Ng. Semantic Compositionality Through Recursive Matrix-Vector Spaces. U *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2012b.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, i Christopher Potts Potts. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. U *EMNLP*, 2013.

Peter D. Turney. Similarity of Semantic Relations. *CoRR*, abs/cs/0608100, 2006.

Dodatak A
Predučenje usporedbom konteksta

wordvec/wordvecs-collobert-weston.py

```
#!/usr/bin/env python

# Learns word vectors using the Collober et al method. Slow for large datasets.
#
# An example of usage:
#
# $ cd wordvec
# $ THEANO_FLAGS=mode=FAST_RUN,device=gpu,floatX=float32,allow_gc=False
# $ ./wordvecs-collobert-weston.py ../datasets/fhrwac.txt words-fhrwac-300k.txt
#
# The resulting vectors are stored in NumPy form into the following file:
#
# ./vectors-collobert-veston.npy

from dictionary import load_dict
from gencontexts import generate_contexts
from numpy import random, sqrt, asarray, array, save
from params import uniformly_to
from performance import gpu
from random import shuffle
from sys import argv
from theano import tensor, shared, config, function, sandbox
from theano.tensor import inc_subtensor, reshape
from theano.printing import debugprint
from theano.tensor.nnet import sigmoid

corpus, words = argv[1:3]
dictionary, by_index, d_size = load_dict(words)
context_w, pivot = 5, 2
word_params_len = 8
hidden_params_len = 16
rand_lim = 0.005
rng = random.RandomState()
n_contexts = int(argv[3]) if len(argv) > 3 else None
```

```
35
36 # Defines a new shared matrix.
37 def rands(threshold, size, name):
38   rs = rng.uniform(low=-threshold, high=threshold, size=size)
39   ar = array(rs, dtype=config.floatX)
40   return shared(ar, name=name, borrow=True)
41
42 # Initialize parameters.
43 L = rands(rand_lim, (word_params_len, d_size), 'L')
44 W = rands(rand_lim, (hidden_params_len, word_params_len*context_w), 'W')
45 b = rands(rand_lim, (hidden_params_len,), 'b')
46 U = [1.0/hidden_params_len for i in range(hidden_params_len)]
47 U = array(U, dtype=config.floatX)
48 U = shared(value=U, name='U', borrow=True)
49
50 # Given the indices, define the word vectors.
51 xinds  = tensor.ivector('xinds')  # Indices of the context.
52 xcinds = tensor.ivector('xcinds') # Indices of the corrupt context.
53 x  = tensor.flatten(L[:,xinds])
54 xc = tensor.flatten(L[:,xcinds])
55
56 # Score the contexts.
57 a  = sigmoid(tensor.dot(W, x)  + b)
58 ac = sigmoid(tensor.dot(W, xc) + b)
59 s  = tensor.dot(U.T, a)
60 sc = tensor.dot(U.T, ac)
61
62 # Final cost function.
63 J = tensor.maximum(0, 1 - s + sc)
64
65 # Get all the gradients.
66 gW  = tensor.grad(J, W)
67 gx  = tensor.grad(J, x)
68 gxc = tensor.grad(J, xc)
69 gb  = tensor.grad(J, b)
70 gU  = tensor.grad(J, U)
71
72 alpha = tensor.fscalar('alpha')
73
74 # A single step, updating the weights based on two contexts.
75 train_step = function(
76   inputs  = [xinds, xcinds, alpha],
77   outputs = [gpu(J), gpu(s), gpu(sc)],
78   updates = [
79     (b, b - alpha*gb),
80     (W, W - alpha*gW),
81     (L,
82      inc_subtensor(
83        inc_subtensor(
84          L[:,xinds],
85          (-alpha)*reshape(gx, (word_params_len, context_w)))[:,xcinds],
86        (-alpha)*reshape(gxc, (word_params_len, context_w))))])
87
88 # Set up training parameters.
```

```
training = list(generate_contexts(corpus, context_w, dictionary, n_contexts))
start_alpha, epochs, alpha_func = 1.0, 3, uniformly_to(0.00001)
total_steps = len(training)*epochs
steps = 0.0

for e in range(1, epochs+1):
  shuffle(training)
  successes = 0

  print('epoch', e, '...')
  for indices in training:
    corrupt = indices[:]
    corrupt[pivot] = rng.randint(0, d_size)
    alpha = alpha_func['func'](start_alpha, steps/total_steps)
    cur_j, cur_s, cur_sc = train_step(indices, corrupt, alpha)
    steps += 1

    if array(cur_s) > array(cur_sc):
      successes += 1

  print('epoch', e, 'had', successes, 'out of', len(training), 'successes')

save('vectors-collobert-weston.npy', L.get_value())
```

wordvec/gencontexts.py

```
# Generates a given number of correct-corrupt context pairs, given a
# corpus path and number of contexts to generate (or None for all).

def generate_contexts(path, context_w, dictionary, up_to_n=None):

  if up_to_n is not None:
    i = 0

  for post in open(path, 'r'):
    words = post.strip().split()
    if len(words) < context_w:
      continue

    for context_x in range(0, len(words) - context_w + 1):
      try:
        yield [dictionary[w] for w in words[context_x : context_x + context_w]]
      except KeyError:
        continue

      if up_to_n is not None:
        i += 1
        if i == up_to_n:
          return
```

Dodatak B
Učenje modela za klasifikaciju sentimenta

sentiment/train.py

```
#!/usr/bin/env python

# Trains the sentiment classification model.
#
# An example of usage:
#
# $ cd sentiment
# $ THEANO_FLAGS=mode=FAST_RUN,device=gpu,floatX=float32,allow_gc=False
# $ ./train.py words-forum-hr.txt vectors-forum-hr.npy examples.txt
#
# Stores all the resulting trained model parameters into different
# NumPy files in the local directory.

from dictionary import load_dict
from error import mean_kl_divergence
from examples import load_examples
from function import create_train_and_eval_functions
from itertools import chain
from nocro import nocro
from numpy import load, zeros, eye, random, asarray, array, vstack, cast, save
from params import generate_hyperparam_combos
from performance import gpu
from random import shuffle
from stem import stem
from sys import argv
from theano import shared, tensor, config, function, Out, sandbox
from theano.tensor.nnet import sigmoid, softmax

# fetch lexicon.
words_path = argv[1]
dictionary, by_index, d_size = load_dict(words_path)
print('dictionary has', d_size, 'entries')

# fetch examples.
```

```
35 examples_path = argv[3]
36 examples = list(load_examples(examples_path, dictionary))
37
38 # split examples into training and testing groups.
39 shuffle(examples)
40 training_ratio = 5.0
41 split_point = int(len(examples)*training_ratio/(training_ratio+1.0))
42 training, testing = examples[:split_point], examples[split_point:]
43 print('took', split_point, 'training examples out of', len(examples), 'total')
44
45 # write down the test set.
46 with open('last-testing-set.txt', 'w') as f:
47   for (x1w, x2w), _, _, y in testing:
48     f.write('%s %s' % (x1w, x2w))
49     for e in y: f.write(' %f' % e)
50     f.write('\n')
51
52 # further split the training group into N buckets, for N-fold cross-validation.
53 n_fold = 10
54 bucket_size = int(len(training)/n_fold)
55 shuffle(training)
56 buckets = [training[i*bucket_size:(i+1)*bucket_size] for i in range(n_fold-1)]
57 buckets.append(training[(n_fold-1)*bucket_size:])
58
59 # run cross-validation for each combination of hyperparamater values,
60 # remembering the ones that perform best.
61 best_kl = None
62 best_combo = None
63
64 combos = list(generate_hyperparam_combos())
65 if len(combos) > 1:
66   for combo in combos:
67     start_alpha, epochs, alpha_func = combo
68     train_step, just_eval, _ = create_train_and_eval_functions(d_size)
69     mean_kl = 0.0
70
71     total_steps = (len(training)-bucket_size)*epochs
72
73     for i in range(n_fold):
74
75       steps = 0.0
76
77       # split the buckets into a validation and training set.
78       iter_validation = buckets[i]
79       iter_training = buckets[:i]
80       if i != n_fold - 1:
81         iter_training += buckets[i+1:]
82       iter_training = list(chain(*iter_training))
83
84       # train using the current hyperparam combo.
85       for e in range(1, epochs+1):
86         shuffle(iter_training)
87         for _, x1, x2, y in iter_training:
```

```
88            train_step(x1, x2, y, alpha_func['func'](start_alpha, steps/
                  total_steps))
89            steps += 1
90
91        mean_kl += mean_kl_divergence(iter_validation, just_eval).mean
92
93      # if mean kl is best, retain the combo.
94      mean_kl = mean_kl/n_fold
95      print('alpha %.3f' % start_alpha, '(%s)' % alpha_func['name'],
96            '(%d epochs)' % epochs, '=> mean KL', mean_kl)
97
98      if best_kl is None or mean_kl < best_kl:
99        print('   (current best KL!)')
100       best_kl = mean_kl
101       best_combo = combo
102
103 if best_combo is None:
104   print('skipped over hyperparam optimization, only got one:')
105   best_combo = combos[0]
106   print('  ', best_combo[0], best_combo[1], best_combo[2]['name'])
107 else:
108   print('best hyperparams combo is', best_combo[0], best_combo[1], best_combo
          [2]['name'])
109
110 # train via best combo on entire training set.
111 start_alpha, epochs, alpha_func = best_combo
112 total_steps = len(training)*epochs
113 steps = 0.0
114 train_step, just_eval, internal_arrays = create_train_and_eval_functions(d_size)
115 print('testset  kl before training is', mean_kl_divergence(testing, just_eval))
116 for e in range(1, epochs+1):
117   shuffle(training)
118   for _, x1, x2, y in training:
119     train_step(x1, x2, y, alpha_func['func'](start_alpha, steps/total_steps))
120     steps += 1
121
122 print('trainset kl  after training is', mean_kl_divergence(training, just_eval))
123 print('testset  kl  after training is', mean_kl_divergence(testing, just_eval))
124
125 # save best trained data.
126 for array in internal_arrays:
127   save('last-trained-%s.npy' % array, array.get_value())
```

sentiment/function.py

```
from numpy import load, zeros, eye, random, asarray, array, vstack, cast, save
from os.path import join
from performance import gpu
from sys import argv
from theano import shared, tensor, config, function, Out, sandbox
from theano.tensor.nnet import sigmoid, softmax, binary_crossentropy as xentropy

# Defines a new shared matrix.
def shared_arr(arr, name, flip_to_column=False):
  arr = asarray(arr, dtype=config.floatX)
  if flip_to_column:
    if arr.shape[0] < arr.shape[1]:
      arr = arr.T
  return shared(arr, name=name, borrow=True)

# Creates the functions neccessary to train and evaluate the model on the GPU.
def create_train_and_eval_functions(d_size, arrays_dir=None):

  # If given a dir, loads all the parameters. Otherwise creates random ones.
  if arrays_dir is not None:
    def locate(array):
      return join(arrays_dir, 'last-trained-%s.npy' % array)

  # Get word vectors L.
  vects_path = argv[2] if arrays_dir is None else locate('L')
  L = shared_arr(load(vects_path), name='L', flip_to_column=True)
  word_params_len = L.get_value().shape[1]

  # Get or create word matrices Lm.
  if arrays_dir is None:
    Lm = zeros((d_size, word_params_len, word_params_len))
    Lm_member_size = (word_params_len, word_params_len)
    for i in range(d_size):
      Lm[i]  = eye(word_params_len)
      Lm[i] += random.normal(scale=0.01, size=Lm_member_size)
  else:
    Lm = load(locate('Lm'))

  Lm = shared(asarray(Lm, dtype=config.floatX), name='Lm', borrow=True)

  # Get or create W, Wlabel and the bias params.
  class_num = 10
  if arrays_dir is None:
    W      = random.normal(scale=0.1, size=(word_params_len, word_params_len*2))
    Wlabel = random.normal(scale=1.0, size=(class_num, word_params_len))
    bias   = random.normal(scale=0.1, size=word_params_len)
  else:
    W      = load(locate('W'))
    Wlabel = load(locate('Wlabel'))
    bias   = load(locate('bias'))

  W      = shared_arr(W, 'W')
  Wlabel = shared_arr(Wlabel, 'Wlabel')
```

```
54    bias   = shared_arr(bias, 'bias')
55
56    # Build the computation graph.
57    alpha  = tensor.fscalar('alpha')
58    ai, bi = tensor.iscalar('ai'), tensor.iscalar('bi') # indices of the word pair
59    a,  b  = L[ai,:], L[bi,:]                           # word vectors
60    A,  B  = Lm[ai], Lm[bi]                             # word matrices
61    z = tensor.concatenate([tensor.dot(B, a), tensor.dot(A, b)])
62    p = sigmoid(tensor.dot(W, z) + bias)
63
64    y = tensor.fvector('y')
65    s = softmax(tensor.dot(Wlabel, p))[0]
66    e = xentropy(s, y).mean()
67
68    gWlabel = tensor.grad(e, Wlabel)
69    gW = tensor.grad(e, W)
70    ga = tensor.grad(e, a)
71    gb = tensor.grad(e, b)
72    gA = tensor.grad(e, A)
73    gB = tensor.grad(e, B)
74    gbias = tensor.grad(e, bias)
75
76    # Score a phrase and update the params accordingly.
77    step = function(
78      inputs = [ai, bi, y, alpha],
79      outputs = gpu(e),
80      updates = [
81        (Wlabel, Wlabel - alpha*gWlabel),
82        (W, W - alpha*gW),
83        (bias, bias - alpha*gbias),
84        (L, tensor.inc_subtensor(
85          tensor.inc_subtensor(
86            L[ai,:],
87            -alpha*ga)[bi,:],
88          -alpha*gb)),
89        (Lm, tensor.inc_subtensor(
90          tensor.inc_subtensor(
91            Lm[ai],
92            -alpha*gA)[bi],
93          -alpha*gB))])
94
95    # Just score the phrase, without updating params.
96    just_s = function(inputs = [ai, bi], outputs = gpu(s))
97
98    return step, just_s, (L, Lm, W, Wlabel, bias)
```

sentiment/error.py

```
from numpy import empty_like, log, sum, array, std, mean

def kl_divergence(q_orig, p):
  return sum(log(p/q_orig)*p)

def mean_kl_divergence(examples, eval_func):
  kls = []
  for _, x1, x2, y in examples:
    pred = eval_func(x1, x2)
    kls.append(kl_divergence(y, array(pred)))
  return KL(mean(kls), std(kls))

class KL:
  def __init__(self, mean, stddev):
    self.mean = mean
    self.stddev = stddev
  def __repr__(self):
    return '%.3f +/- %.3f' % (self.mean, self.stddev)
```

Primjena modela dubokog učenja na analizu sentimenta izraza hrvatskoga jezika

Sažetak

Uobičajeni postupci analize sentimenta temelje se na rječniku apriornog sentimenta. Problem predstavlja modeliranje sentimenta višerječnih izraza poput "poprilično dobar" ili "nimalo loš", ali i većih jezičnih jedinica. Opisan je i implementiran postupak učenja reprezentacija riječi prema metodi Colloberta i dr. (2011) te postupak analize sentimenta višerječnih izraza hrvatskoga jezika modelom zasnovanom na rekurzivnoj neuronskoj mreži prema radu Sochera i dr. (2012). Navedeni su rezultati učenja reprezentacija riječi na temelju dva različita korpusa i rezultati evaluacije modela za analizu sentimenta nad tri različita skupa za učenje.

Ključne riječi: obrada prirodnog jezika, duboko učenje, učenje reprezentacija riječi, analiza sentimenta, hrvatski jezik.

Using Deep Learning for Sentiment Analysis of Croatian Expressions

Abstract

Methods of sentiment analysis are usually based upon a sentiment-labeled lexicon. Problems arise when trying to model the sentiment of multi-word phrases such as "pretty good" or "not bad", but also of entire sentences. A word-representation training method based on Collobert et al. (2011) is defined and implemented, as is a method for sentiment analysis of multi-word phrases written in Croatian using a model based on recursive neural networks according to Socher et al. (2012). We state the results of word-representation training on two different corpora and the results of evaluating the sentiment analysis model on three different training sets.

Keywords: natural language processing, deep learning, word representation pre-training, sentiment analysis, Croatian language.