Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tilen Zelinka

Vektorske vložitve za prepoznavanje slovenskih glagolskih idiomov

diplomsko delo

univerzitetni študijski program
prve stopnje
računalništvo in informatika

Mentor: prof. dr. Marko Robnik Šikonja

Ljubljana, 2019

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Večbesedne zveze so skupine besed, katerih celostni pomen ni vsota pomenov posameznih besed. Ker se pogosto pojavljajo v različnih vrstah sporazumevanja, jih morajo sistemi za obdelavo naravnega jezika prepoznati in ustrezno obravnavati, npr. za uspešno strojno prevajanje. Avtomatska prepoznava večbesednih zvez v stavkih je zato pomemben, a težek problem. Goste vektorske vložitve preslikajo posamezne besede v visokodimenzionalen vektorski prostor, kjer podobne besede ostanejo blizu skupaj. To lastnost vložitev želimo izkoristiti za prepoznavanje glagolskih idiomov, saj gre pri njih za velike spremembe pomenov besednih zvez glede na njihove sestavne dele. Proučite problem prepoznavanja glagolskih idiomov s pomočjo vektorskih vložitev in strojnega učenja. Skonstruirajte značilke na podlagi vložitev besed in besednih zvez ter naučite model strojnega učenja. Rešitev ovrednotite na primeru slovenskega jezika.

Zahvaljujem se mentorju, prof. dr. Marku Robniku Šikonji, za hitro odzivnost, potrpežljivost in strokovno pomoč pri izdelavi diplomske naloge. Hvala tudi mojim najbližjim, za vso podporo in spodbudo med celotnim študijem.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
CA	classification accuracy	klasifikacijska točnost
AUC	area under the ROC curve	ploščina pod krivuljo ROC
SVM	support vector machine	metoda podpornih vektorjev
RF	random forests	naključni gozdovi
LR	logistic regression	logistična regresija
NN	neural network	nevronska mreža
CNN	convolutional neural network	konvolucijska nevronska mreža
MWE	multi-word expressions	VBZ, večbesedne zveze
VMWE	verbal multi-word expressions	GVBZ, glagolske večbesedne zveze
VID	verbal idioms	glagolski idiomi

Povzetek

Naslov: Vektorske vložitve za prepoznavanje slovenskih glagolskih idiomov

Vektorske vložitve preslikajo besede v visokodimenzionalne vektorje realnih števil, pri čemer imajo besede s podobnimi pomeni podobne vektorje. Preučili smo problem avtomatske prepoznave slovenskih glagolskih idiomov z uporabo značilk, zgrajenih iz vektorskih vložitev skupin besed in vektorskih vložitev posameznih besed. V ta namen smo zgradili dve podatkovni množici, ki vsebujeta primere glagolskih idiomov in naključnih skupin besed, opisanih z zgrajenimi značilkami. Na teh množicah smo ocenili uspešnost klasifikacije glagolskih idiomov z metodo podpornih vektorjev, naključnih gozdov in logistične regresije. Vse tri metode so pri klasifikaciji dokaj uspešne, najbolje se je izkazala metoda naključnih gozdov. Zaradi časovne zahtevnosti in omejitev prepoznave na skupine besed, za katere so znane vektorske vložitve, pa bodo za praktično uporabo potrebne še dodatne izboljšave.

Ključne besede: obdelava naravnega jezika, vektorske vložitve, stalne besedne zveze, strojno učenje.

Abstract

Title: Word embeddings for detection of verbal idioms in Slovene

Word embeddings map words to a high dimensional vector space, where words with similar meanings have similar vectors. We analyzed the problem of automatic identification of verbal idioms in Slovene using features built from embeddings of single words and groups of words. For this purpose, we built two data sets that contain verbal idioms and random word groups described with corresponding features. Using these data sets we evaluated the classification of verbal idioms with support vector machines, random forests, and logistic regression. All three methods were successful, the best being random forests. Due to large computational time and limitation to only identify groups of words with precomputed word embeddings the approach requires further improvements to be practically useful.

Keywords: natural language processing, word embeddings, multiword expressions, machine learning.

Poglavje 1

Uvod

Zaznava večbesednih zvez predstavlja pomemben, vendar težak problem na področju obdelave naravnega jezika. Gre za besedne zveze, pri katerih celota nosi drugačen pomen od vsote pomena posameznih besed. Eno od področij, na katerem je pomembna dobra zaznava večbesednih zvez, je strojno prevajanje, kjer bi se ob dobesednem prevodu vsake besede v večbesedni zvezi popolnoma izgubil pravi pomen. V kolikor je večbesedna zveza prepoznana, se lahko ob prevodu obravnava kot celota ter tako ohrani pomen.

V zadnjih letih se je razširila uporaba vektorskih vložitev besed za reševanje raznovrstnih problemov na področju obdelave naravnega jezika. Besede se preslikajo v vektorje v visokodimenzionalnem prostoru, pri čemer imajo besede s podobnimi pomeni podobne vektorje. V nalogi smo poizkusili preveriti, ali lahko iz prostorskih odvisnosti za izbrano skupino besed napovemo, če predstavlja večbesedno zvezo. Osredotočili smo se na glagolske idiome, ki so vrsta večbesednih zvez, pri katerih prihaja do velikih razlik med pomenom celote in pomenom vsote sestavin. V ta namen smo zgradili dve podatkovni množici, ki vsebujeta značilke zgrajene iz vektorskih vložitev. Ena podatkovna množica je zgrajena iz vložitev korpusa ccKRES, druga pa iz vložitev korpusa ccGigafida. Množici vsebujeta primere glagolskih idiomov pridobljenih iz korpusa PARSEME in primere naključnih skupin besed. Na množicah smo ocenili uspešnost klasifikacije glagolskih idiomov izbranih klasifikacijskih

metod.

Delo je sestavljeno iz šestih poglavij. V 2. poglavju predstavimo vektorske vložitve besed in klasifikacijske modele uporabljene pri evalvaciji. V 3. poglavju opišemo večbesedne zveze in glagolske idiome ter predstavimo uporabljene jezikovne korpuse. V 4. poglavju opišemo postopek učenja vektorskih vložitev, izbiro značilk in sestavo zgrajenih podatkovnih množic. V 5. poglavju evalviramo pomembnost posameznih atributov, uspešnost klasifikacijskih metod pri prepoznavi glagolskih idiomov v zgrajenih množicah in za to potreben čas. V 6. poglavju povzamemo narejeno, predstavimo zaključke in ideje za izboljšave.

Poglavje 2

Metode strojnega učenja

V tem poglavju najprej predstavimo vektorske vložitve, ki smo jih uporabili za učenje vektorjev skupin besed in posameznih besed. Sledi opis klasifikacijskih metod, ki smo jih uporabili za učenje prepoznave glagolskih idiomov (VID) in sicer: metode podpornih vektorjev, naključnih gozdov in logistične regresije.

2.1 Vektorske vložitve

Vektorske vložitve besed (angl. word embeddings) posameznim besedam ali delom povedi priredijo vektorje realnih števil. Za razliko od redke predstavitve besednih vektorjev, pri katerih je število dimenzij enako številu besed v slovarju in je vsaka beseda določena z vektorjem, pri katerem je ena dimenzija enaka 1 ostale pa 0, so takšni vektorji ponavadi krajši, posamezna dimenzija pa nosi več informacije. Učenje gostih vektorjev je računsko bolj zahtevno, zaradi česar se je metoda razširila šele pred kratkim. Goste vektorske vložitve besed so bile prvič predstavljene leta 2003 [2]. Do velikega napredka je prišlo leta 2013, ko je bila predstavljena metoda Word2vec [18], ki je omogočila hitrejše pridobivanje in obdelavo vektorskih vložitev. To je povzročilo, da so vektorske vložitve danes ena od glavnih tehnik za predstavitev besed v računske namene.

Moderne vektorske vložitve so implementirane z uporabo nevronskih mrež (NN), vendar se je razvilo več pristopov z različnimi arhitekturami [8]. Osnovna ideja izvira iz jezikovnih modelov, kjer poskušamo napovedati naslednjo besedo v nizu besed, na podlagi n predhodnih besed. Word2vec je predstavil dva pristopa, ki sta še vedno v uporabi: zvezna vreča besed (CBOW), ki za napoved trenutne besede uporablja predhodne in sledeče besede, ter preskočni n-gram (skip-gram), ki problem obrne in poskuša napovedati sosednje besede glede na podano besedo.

Kljub različnim arhitekturam NN za vektorske vložitve, te ponavadi vsebujejo tri tipe plasti [21]:

- vložitvena plast, ki generira vektorske vložitve tako, da pomnoži pozicijski vektor z matriko vektorskih vložitev,
- vmesne plasti, ki jih je pogosto več in na različne nelinearne načine transformirajo vhod,
- izhodna logistična (softmax) plast, ki izračuna verjetnostno porazdelitev napovedanih besed.

Ena od glavnih prednosti vektorskih vložitev je, da lahko vektorje naučimo na neoznačenih besedilih. To pomeni, da so nam na voljo bistveno večji korpusi, kot pri tehnikah, ki zahtevajo oznake.

Pomembna lastnost vektorskih vložitev je, da so dobljeni vektorji besed prostorsko povezani s pomenom besed. Na primer, vektorji za besede glavnih mest držav so blizu skupaj v prostoru. Nad takšnimi vektorji lahko izvajamo matematične operacije kot so seštevanje, odštevanje, računanje kosinusne razdalje ipd. Znan primer je predstavljen v enačbi (2.1), ki velja, kadar besede predstavljajo naučene vektorje.

$$kralj - moški + ženska \approx kraljica \tag{2.1}$$

V tej diplomski nalogi smo za učenje vektorskih vložitev uporabili prosto dostopno knjižnico fastText [6], ki je namenjena učenju klasifikacije povedi [9] in besednih vektorjev [3]. Knjižnica za Word2vec ponuja izbiro med

modeloma CBOW in skip-gram ter vsebuje dodatne funkcionalnosti, kot je prepoznava besed, ki niso bile v učni množici. Tega za namene naloge nismo uporabili. Knjižnica fastText je na voljo za programska jezika Python in C++.

2.2 Klasifikacijski modeli

V nadaljevanju so predstavljeni klasifikacijski modeli, ki so bili uporabljeni za učenje prepoznave VID na podatkovni množici, ki je bila zgrajena v okviru naloge.

2.2.1 Metoda podpornih vektorjev

Metoda podpornih vektorjev (SVM) se uporablja pri nadzorovanem strojnem učenju za klasifikacijo in regresijo. Prestavljena je bila leta 1995 [4]. Za razliko od večine drugih algoritmov strojnega učenja, ki zgradijo model na čim bolj ustreznih podmnožicah atributov, SVM uporabi vse atribute, tudi manj pomembne [12].

Pri SVM vsak atribut predstavlja koordinato v prostoru, cilj je postaviti optimalno hiperravnino, ki deli pozitivne in negativne primere. Optimalna hiperravnina je najbolj oddaljena od najbližjih primerov obeh razredov. Kadar popolna delitev ni možna, pri klasifikaciji učnih primerov dovolimo napako, ki pa jo nato minimiziramo. Včasih v originalnem prostoru hiperravnina ne zadošča za dobro delitev razredov. V tem primeru nad učnimi primeri predhodno izvedemo transformacijo v kompleksnejši prostor ter nato rešujemo problem z iskanjem optimalne hiperravnine v novem prostoru.

V praksi se SVM pogosto izkaže kot ena uspešnejših metod klasifikacije, še posebej pri velikh množicah z velikim številom manj pomembnih atributov. Težava je, da je težko interpretirati naučeno ter razložiti posamezne odločitve [12].

2.2.2 Naključni gozdovi

Namen metode naključnih gozdov je izboljšava točnosti drevesnih modelov [12]. Prvotno je bila metoda razvita za odločitvena drevesa. To so drevesne strukture, pri katerih se glede na izbrane kriterije na vsakem nivoju izbere atribut, ki najbolje deli trenutno učno množico. Ta se razdeli, postopek pa se rekurzivno ponavlja na naslednjih nivojih drevesa, dokler ne doseže ustavitvenega pogoja. Listi drevesa ustrezajo razredom.

Pri metodi naključnih gozdov se zgradi množica odločitvenih dreves, ki so ponavadi preprostejša, atributi pa so v vsakem vozlišču izbrani iz naključne podmnožice atributov. Klasifikacija novih primerov se izvaja s pomočjo glasovanja, pri čemer ima vsako naključno zgrajeno drevo en glas. Razporeditev glasov po razredih predstavlja napovedano verjetnostno porazdelitev.

Metoda močno zmanjša varianco preprostih odločitvenih dreves in je primerljiva z najboljšimi napovednimi algoritmi. Podobno kot pri SVM je tudi tu težava razlaga odločitev modela [12].

2.2.3 Logistična regresija

Logistična regresija je metoda za klasifikacijo, ki v osnovni obliki deluje na problemih z dvema razredoma, pri čemer se eden obravnava kot pozitivni, drugi pa kot negativni. Določimo funkcijo, ki glede na vektorje atributov učnih primerov (vsak atribut predstavlja eno dimenzijo vektorja) minimizira napako napovedi v učni množici. S pomočjo naučene funkcije za vsak nov primer izračunamo verjetnost pripadnosti pozitivnemu razredu, ki je med 0 in 1. Za negativni razred se verjetnost izračuna tako, da se dobljena verjetnost pozitivnega razreda odšteje od 1 (npr. verjetnost 0.15 bi pomenila, da spada primer v pozitivni razred z verjetnosjo 15%, v negativni pa z verjetnostjo 85%) [11].

Glavna prednost logistične regresije je, da lahko izračunane koeficiente interpretiramo kot razlage pomembnosti atributov in kot prispevke h klasifikaciji.

Poglavje 3

Glagolske večbesedne zveze

Za večbesedne zveze (VBZ) obstaja več definicij. Jezikoslovni vidik jih opredeljuje kot zveze, katerih celostni pomen ni vsota pomenov posameznih sestavin [7]. Ker se pogosto pojavljajo v vseh vrstah sporazumevanja, potrebujemo njihovo dobro prepoznavo. VBZ na področju obdelave naravnega jezika predstavljajo težak problem. Pogosto jih obravnavamo tako, da jih poiščemo v besedilu iz njih odstranimo presledke in jih s tem združimo v eno besedo. Težava pri tem pristopu je, da lahko v nekaterih primerih pride do oblikoskladenjskih sprememb, kar včasih pokvari zaznavo. Nepopoln model bi na primer lahko kot VBZ pravilno označil zvezo *"šlo po načrtih"*, hkrati pa bi označil zvezo *"šel po načrt"*, ki pa je uporabljena dobesedno [22].

Pri glagolskih večbesednih zvezah (GVBZ) je eden od sestavnih delov glagol. S prepoznavo teh zvez se ukvarja COST akcija PARSEME [23]. V okviru akcije je bilo razvitih več pristopov za indentifikacijo, ki uporabljajo različne algoritme kot npr. nevronske mreže [10] in pogojna naključna polja [17].

V okviru akcije PARSEME 1.1 so za slovenski jezik GVZB deljene na štiri kategorije:

- inherentno povratni glagoli (npr. *tiskati se*, *dati se*), ko glagol obstaja le, ko se veže z veznikom *se/si* ali pa mu ta spremeni pomen,
- zveze z glagoli v pomensko oslabljeni rabi (npr. *imeti predavanje*, *biti v dvomih*),

- predložnomorfemski glagoli (npr. *apelirati na*, *biti za*), ko glagol obstaja le, ko se veže s predložnim morfemom ali pa mu ta spremeni pomen,
- glagolski idiomi (npr. *dati pečat*, *oči so večje od želodca*)

V nadaljevanju so bolj natančno predstavljeni glagolski idiomi, na katere se osredotočamo v tej diplomski nalogi. Predstavimo tudi COST akcijo PARSEME, pripadajoči korpus različice 1.1 ter korpusa ccKRES in ccGigafida, uporabljena za učenje vektorskih vložitev.

3.1 Glagolski idiomi

Glagolski idiomi (VID) so ena od vrst GVBZ, ki so bile definirane v projektu PARSEME 1.1 [20]. V smernicah projekta so VID opredeljeni kot zveza vsaj dveh leksikaliziranih komponent, od katerih je glagol skladenjsko nadrejen najmanj eni. Zveza mora izkazovati samostojen pomen, ki je večinoma neodvisen od posameznih sestavnih delov(*pljuniti v roke* - pričeti z delom). Pri slovenščini za tovrstne GVBZ ponavadi uporabljamo izraz glagolski frazemi. VID ne nastopajo vedno kot glagolsko jedro stavka. Določeni glagolski frazemi lahko nastopajo tudi v vlogi predmetnega določila (npr. *ne spodobi se*) ali v vlogi stavka (npr. *srce se trga (komu)*). Zato termina nista popolnoma enaka, se pa v večjem delu prekrivata [7].

VID so večbesedne zveze, pri katerih je pogosta velika razlika med pomenom posameznih besed in celotne zveze, kar naredi kategorijo zanimivo za prepoznavo z vektorskimi vložitvami.

3.2 PARSEME

COST akcija PARSEME Shared Task [23, 19] je mednarodni interdisciplinarni znanstveni projekt, ki se osredotoča na avtomatično identifikacijo GVBZ v besedilih. Del projekta se ukvarja z razvojem korpusov za več je-

zikov z oznakami za GVBZ, ki bi se nato uporabil za učenje avtomatične prepoznave v besedilih.

Pri tej diplomski nalogi je uporabljen korpus za slovenski jezik razvit v sklopu PARSEME 1.1 [20], ki je na voljo od avgusta 2018 in vključuje korpuse za 19 jezikov. Slovenski korpus je zgrajen iz dela učnega korpusa ssj500k 2.0 [13], ki vsebuje odstavke vzorčene iz korpusa FidaPLUS [1]. Vsebuje odstavke iz literarnih del, časopisov, revij in govora. PARSEME Shared Task 1.1 za slovenščino vsebuje 13.511 označenih stavkov, od tega jih 2.920 vsebuje GBZ različnih kategorij, VID pa je označenih 724, od tega 457 unikatnih. Prevladujoče strukture VID znotraj korpusa so zveze glagola *biti* (*biti všeč, biti jasno, biti narobe* ipd.) in glagola *imeti* (*imeti smisel za, ne imeti pojma* ipd.). Druga pogosta obilka je zveza glagola s samostalnikom (*dati pečat, delati gužvo* ipd.). VID se pojavljajo tudi v drugačnih oblikah, kot na primer v stavčnih strukturah (npr. *stara zgodba se ponavlja*) ali pregovorov (npr. *kot svinja z mehom*) vendar manj pogosto [7]. Slika 3.1 prikazuje primer stavka z oznako VID v korpusu PARSEME.

Korpus je za namene PARSEME Shared Task že razdeljen na učno in testno množico, vendar za cilje te naloge delitve nismo potrebovali, zato smo korpus združili v celoto.

3.3 ccKRES

Eden od korpusov uporabljen za učenje vektorskih vložitev je ccKRES [15, 16, 5]. Za razliko od korpusa KRES, ki vsebuje avtorska dela in iz katerega je vzorčen, je ccKRES prosto dostopen. Vsebuje približno 10 milijonov besed, kar je približno 9% korpusa KRES, hkrati pa ohranja njegovo zgradbo. To je pomembno, saj je KRES uravnotežen, kar pomeni, da ima pestro sestavo različnih vrst besedil in s tem predstavlja dokaj celovito podobo jezika. Besedila znotraj korpusa so izšla med leti 1990 in 2011 in vsebujejo 20% revij, 20% časopisov, 20% internetnih besedil, 17% leposlovja, 18% stvarnih

```
# source_sent_id = . . autogen--data.parsemetsv--12073
# text = No, pa nas je spravil v dobro voljo.
1 No no _ L _ --_ SpaceAfter=No *
2 ,,_ ,_ --_ _ *
3 pa pa _ Vp _ --_ _ *
4 nas jaz _ Zop-mt _ --_ _ *
5 je biti _ Gp-ste-n _ --_ _ *
6 spravil spraviti _ Ggdd-em _ --_ _ 1:VID
7 v v _ Dt _ --_ _ 1
8 dobro dober _ Ppnzet _ --_ _ 1
9 voljo volja _ Sozet _ --_ SpaceAfter=No 1
10 . . _ . _ --_ SpaceAfter=No *
```

Slika 3.1: Primer oznake VID za besedno zvezo ¨spraviti v dobro voljo¨ v korpusu PARSEME.

besedil in 5% drugih besedil. Besede so bile označene avtomatsko, poleg vsake je zapisana lema in oblikoskladenjska oznaka, ki označuje vrsto besede (samostalnik, glagol ipd.) in njene lastnosti (spol, sklon, število) [16].

3.4 ccGigafida

Drugi prosto dostopen korpus uporabljen za učenje vektorskih vložitev je ccGigafida [14, 16, 5]. Vsebuje približno 100 milijonov besed, kar predstavlja približno 9% korpusa Gigafida iz katerega je vzorčen. Korpus pri tem ohranja zgradbo korpusa Gigafida, ki pa v nasprotju s korpusom KRES ni uravnotežen. Vsebuje besedila med leti 1990 in 2011 in vsebujejo 21% revij, 56% časopisov, 16% internetnih besedil, 2% leposlovja, 4% stvarnih besedil in 1% drugih besedil. Besede so označene na enak način kot v korpusu ccKRES [16]. Prednost korpusa ccGigafida pred korpusom ccKRES je, da vsebuje bistveno večje število besed.

Poglavje 4

Prepoznavalnik

Za učenje prepoznave VID smo izluščili primere VID iz korpusa PARSEME. Zgradili smo vektorske vložitve za posamezne besede in skupine naključnih besed znotraj korpusov ccKRES in ccGigafida. Ideja je bila iz vektorjev za skupine besed in vektorjev posameznih besed pridobiti značilke, s katerimi bi lahko z metodami strojnega učenja čim bolje napovedali, ali je skupina besed VID. V ta namen smo zgradili podatkovno množico, ki vsebuje pozitivne primere iz PARSEME izluščenih VID in negativne primere naključnih besednih skupin ter pripadajoče značilke izračunane iz naučenih vektorjev. V nadaljevanju so bolj podrobno opisani postopki učenja vektorskih vložitev, izbira značilk in zgradba končnih podatkovnih množic. Podatkovni množici in izvorna koda diplomske naloge so na voljo na spletnem naslovu https://github.com/zelinka/VVPSGI/.

4.1 Učenje vektorskih vložitev

Za izračun značilk smo potrebovali vektorje posameznih besed in skupin besed. Ker je možnih skupin besed mnogo, smo namesto prvotnega besedila iz korpusov izluščili lematizirano besedilo, kar je pri posameznih besedah odstranilo oblikoskladenjske spremembe in zmanjšalo velikost slovarja. Besede smo v skupine povezali z znakom ¨_¨ ter tako omogočili, da so se pri

učenju obravnavale kot ena beseda. Primer povezovanja besed je prikazan na sliki 4.1. Zaradi velikega števila možnih skupin smo, da bi zagotovili čim večje število naučenih vektorjev za primere VID, iz korpusa PARSEME izluščili množico VID, ter jih v besedilu označili najprej. Osredotočili smo se na VID, ki se v korpusu pojavljajo neprekinjeno, kar pomeni, da med prvo in zadnjo besedo VID v besedilu ni besed, ki ne bi bile del VID. Tovrstnih VID je v korpusu PARSEME 252. V skupine 2, 3 ali 4 zaporednih besed smo nato povezali še vse ostale besede, pri čemer smo upoštevali, da se v skupine niso povezovale besede iz različnih stavkov.

```
Jacob je skomignil z rameni in krenil dalje .
Jacob biti skomigniti z rame in kreniti dalje .
Jacob_biti skomigniti_z_rame in_kreniti_dalje .
```

Slika 4.1: Primer stavka iz prvotnega besedila ter stavkov lem posameznih besed in lem skupin besed uporabljenih pri učenju vektorskih vložitev.

Za učenje vektorjev posameznih besed in skupin smo uporabili knjižnico fastText v programskem jeziku C++. Pri obeh smo nastavitve ohranili na privzetih vrednostih, kjer imajo naučeni vektorji 100 dimenzij in je uporabljen skip-gram model. Ta deluje bolje na besedah, ki se v besedilu redko ponavljajo. Teh je zaradi velikega slovarja skupin besed, več.

Iz korpusa ccKRES smo na tak način pridobili 57.345 vektorjev posameznih besed in 48.925 vektorjev skupin besed, od tega 139 primerov vektorjev VID pridobljenih iz korpusa PARSEME. Iz korpusa ccGigafida smo pridobili 207.319 vektorjev posameznih besed in 721.160 vektorjev skupin besed, od tega 209 primerov vektorjev VID pridobljenih iz PARSEME. Primeri izbranih VID predstavljajo majhen delež vseh naučenih vektorjev. Vzrok za to je, da je v množici PARSEME majhno število edinstvenih VID, od katerih se določen del ne pojavi v dovolj velikem številu, da bi pri učenju pridobili njihove vektorje. Majhen delež množic naučenih vektorjev verjetno sestavljajo tudi VID, ki niso vsebovani v korpusu PARSEME, od koder smo izluščili

pozitivne primere za gradnjo podatkovnih množic. Dobljene vektorje smo uporabili za računanje značilk, ki smo jih dodali v podatkovno množico.

4.2 Značilke

S pomočjo naučenih vektorjev skupin besed in vektorjev posameznih besed, smo za vsako izbrano skupino besed izračunali 8 značilk. Te so:

- evklidska norma naučenega vektorja skupine besed (Mag Group),
- evklidska norma vektorja, ki predstavlja povprečje naučenih vektorjev posameznih besed, ki sestavljajo skupino (Mag Avg),
- kosinusna razdalja med vektorjem skupine besed in vsoto vektorjev posameznih besed znotraj skupine (CosDist Sum),
- skalarni produkt med vektorjem skupine besed in vsoto vektorjev posameznih besed znotraj skupine (DotProd Sum),
- povprečna kosinusna razdalja med skupino besed in vsemi naučenimi skupinami besed, ki se razlikujejo v eni besedi (CosDist Sim),
- kosinusna razdalja med vektorjem skupine besed in najbližjim vektorjem v množici vseh naučenih vektorjev skupin besed (N1),
- kosinusna razdalja med vektorjem skupine besed in drugim najbližjim vektorjem v množici vseh naučenih vektorjev skupin besed (N2),
- kosinusna razdalja med vektorjem skupine besed in tretjim najbližjim vektorjem v množici vseh naučenih vektorjev skupin besed (N3).

4.3 Podatkovna množica

Iz pripadajočih naučenih vektorjev smo za vsakega od korpusov ccKRES in ccGigafida zgradili podatkovno množico. Za pozitivne primere VID smo vzeli

uspešno naučene vektorje vseh skupin besed, ki so enaki prvotno označenim VID izluščenim iz korpusa PARSEME. Negativne primere smo izbrali naključno med vsemi ostalimi vektorji skupin besed na tak način, da smo med številom pozitivnih in negativnih primerov ohranili razmerje 1:1, ob tem pa je bila povprečna dolžina skupin besed pri pozitivnih in negativnih primerih približno enaka. Poleg značilk posameznih skupin, izračunanih na deset decimalk natančno, podatkovni množici vsebujeta še atributa *Group*, v katerem je zapisano ime skupine besed, in atribut *Type*, ki z 1 označuje pozitivne primere in z 0 negativne.

V nadaljevanju sta bolj podrobno predstavljeni podatkovni množici zgrajeni iz korpusov ccKRES in ccGigafida.

4.3.1 Podatkovna množica VID ccKRES

Množica vsebuje 278 skupin besed, od tega 139 primerov VID. Povprečna dolžina skupin besed je pri negativnih in pozitivnih primerih enaka 2,59. Manjkajoče vrednosti se pojavljajo samo pri atributu *CosDist Sim* pri 30 skupinah besed, od tega 18 pri pozitivnih primerih in 12 pri negativnih. Razlog za manjkajoče vrednosti je, da med skupinami besed za katere smo uspešno pridobili vektorje ni takšnih, ki bi se od izbrane skupine besed razlikovale v samo eni besedi.

4.3.2 Podatkovna množica VID ccGigafida

Množica vsebuje 418 skupin besed, od tega 209 primerov VID. Povprečna dolžina skupin besed je pri pozitivnih primerih 2,69, pri negativnih pa 2,70. Manjkajoče vrednosti se ponovno pojavljajo samo pri atributu *CosDist Sim* pri 22 skupinah besed, od tega 21 pri pozitivnih primerih in 1 pri negativnih. Tabela 4.1 vsebuje primer dveh vnosov v podatkovni množici VID ccGigafida.

Group	Mag Group	Mag Avg	CosDist Sum	DotProd Sum	CosDist Sim	N1	N2	N3	Type
iti_v_nič	3.0408...	1.6102...	1.0481...	-0.7077...	0.4071..	0.1407...	0.1598...	0.1599...	1
se_dva_dan	3.0721...	1.6125...	1.1021...	-1.5179...	0.3363...	0.1504...	0.1743...	0.1761...	0

Tabela 4.1: Primer vnosa za dve skupini besed v podatkovni množici VID ccGigafida. Za boljšo preglednost so števila prikazana le na štiri decimalna mesta natančno.

Poglavje 5

Evalvacija

V tem poglavju je evalvirana uspešnost izbranih metod strojnega učenja pri indentifikaciji VID v zgrajenih podatkovnih množicah. Analiziramo pogoste napake, ki jih dela klasifikator. S pomočjo interpretacije koeficientov logistične regresije, ocene pomembnosti atributov z metodo naključnih gozdov in izrisa korelacijskih matrik obeh množic smo poskusili ugotoviti pomembnost posameznih atributov pri klasifikaciji. Analiziramo tudi čas potreben za učenje vektorskih vložitev in za izračun značilk. Metodo podpornih vektorjev (SVM), naključne gozdove (RF) in logistično regresijo (LR) smo klicali iz Python knjižnice scikit-learn. Pri SVM smo obliko jedra spremenili v linearno in število dreves pri RF smo nastavili na 100, ostale nastavitve smo ohranili na privzetih vrednostih knjižnice. Pri evalvaciji smo uporabili 10-kratno prečno preverjanje. Kot mere za ocenjevanje učenja smo uporabili klasifikacijsko točnost (CA), ploščino pod krivuljo ROC (AUC) ter senzitivnost in specifičnost.

5.1 Analiza atributov podatkovnih množic

Interpretacija koeficientov logistične regresije je predstavljena v tabeli 5.1. Razvidno je, da so pri obeh množicah najbolj pomembni *N1, N2, N3* in *CosDist Sim.* Z večjim slovarjem v podatkovni množici VID ccGigafida se

	Mag Group	Mag Avg	CosDist Sum	DotProd Sum	CosDist Sim	N1	N2	N3
Podatkovna množica VID ccKRES	-0.923	0.533	-0.225	0.023	1.834	1.743	3.972	-1.553
Podatkovna množica VID ccGigafida	-0.751	0.282	-0.403	-0.203	2.297	1.963	2.029	-1.389

Tabela 5.1: Koeficiente logistične regresije lahko interpretiramo kot pomembnost atributov pri klasifikaciji.

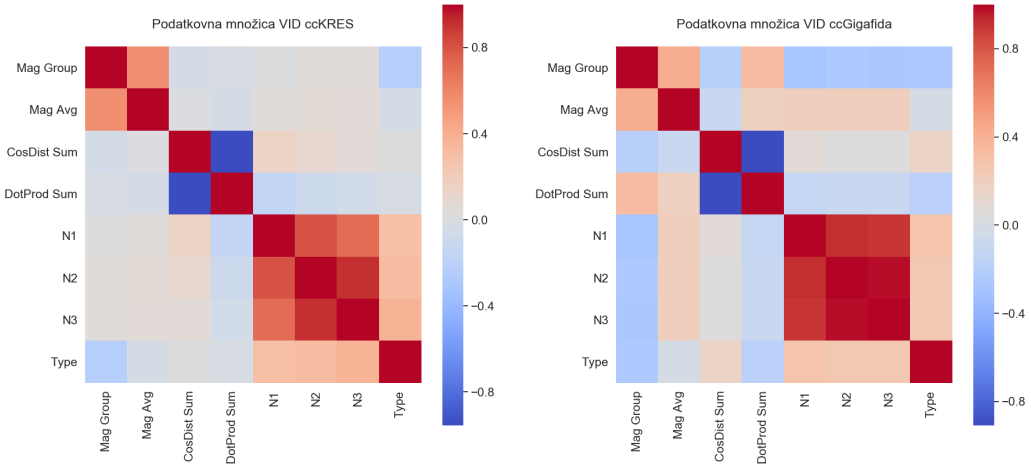
	Mag Group	Mag Avg	CosDist Sum	DotProd Sum	CosDist Sim	N1	N2	N3
Podatkovna množica VID ccKRES	0.157	0.11	0.099	0.099	0.135	0.142	0.157	0.102
Podatkovna rnnožica VID ccGigafida	0.141	0.115	0.091	0.098	0.146	0.136	0.125	0.148

Tabela 5.2: Ocene pomembnosti atributov z uporabo metode RF.

v primerjavi s podatkovno množico VID ccKRES pomembnost posameznih atributov rahlo spremeni. Posebej izstopa *N2*, ki je v množici VID ccGigafida ocenjen kot bistveno manj pomemben.

V tabeli 5.2 so prikazane ocene pomembnosti atributov pri klasifikaciji z metodo RF. Atributi so po pomembnosti razvrščeni podobno kot v tabeli 5.1. Glavna razlika je pri atributu *Mag Group*, ki ga RF v obeh množicah oceni kot enega najpomembnejših. Vidne so tudi manjše razlike v ocenah med najnižje in najvišje ocenjenimi atributi.

Za pomoč pri oceni posameznih atributov smo izrisali tudi korelacijski matriki obeh množic, ki sta prikazani na sliki 5.1. Podobno kot v tabelah 5.1 in 5.2 je tudi tu razvidno, da je pri večji množici VID ccGigafida manjša korelacija med razredom *Type* in atributi, ki predstavljajo razdalje do najbližjih sosednjih vektorjev, ob tem pa se nekoliko poveča korelacija med *Type* in ostalimi atributi.



Slika 5.1: Korelacijski matriki obeh podatkovnih množic.

5.2 Rezultati učenja

Kot je razvidno iz rezultatov, ki so prikazani v tabeli 5.3, se v obeh podatkovnih množicah kot najbolj uspešna metoda izkaže RF, ki je najuspešnejša glede na vse mere, razen senzitivnost v podatkovni množici VID ccKRES. Poleg tega je tudi edina metoda, ki ohranja podobno CA v obeh množicah, pri ostalih dveh je CA v množici VID ccGigafida bistveno manjša. Razlog za to je verjetno, da sta razreda v množici VID ccKRES večinoma deljena glede na atribute *N1, N2* in *N3*, medtem ko je v množici VID ccGigafida pomembnost atributov bolj enakomerno razporejena in je razreda težje ločiti linearno. Metoda RF je v obeh množicah najboljša tudi pri ocenah specifičnosti in AUC, v množici VID ccKRES pa ima najslabšo senzitivnost. Po senzitivnosti je SVM boljši od RF v množici VID ccKRES in primerljiv z RF v množici VID ccGigafida, vendar ima od vseh uporabljenih algoritmov najslabšo specifičnost in CA. Na nobeni od množic SVM ne daje bistveno drugačnih rezultatov od LR, kar je verjetno posledica majhnega števila pomembnih atributov.

Analizirali smo napake pri klasifikaciji z metodo RF v podatkovni množici VID ccGigafida. Pogoste so napake pri določenih VID, ki vsebujejo glagol

		CA	Senzitivnost	Specifičnost	AUC
Podatkovna množica VID ccKRES	SVM	0.702	**0.77**	0.634	0.768
	RF	**0.72**	0.69	**0.749**	**0.778**
	LR	0.705	0.741	0.67	0.764
Podatkovna množica VID ccGigafida	SVM	0.621	0.712	0.531	0.731
	RF	**0.715**	**0.731**	**0.688**	**0.791**
	LR	0.643	0.688	0.597	0.689

Tabela 5.3: Ocene izbranih algoritmov strojnega učenja na osnovnih podatkovnih množicah pridobljenih iz korpusov ccKRES in ccGigafida.

biti, kot so: *biti_prav*, *biti_narobe*, *biti_v_korak_z* ali *biti_všeč*. Manjši delež tovrstnih VID se pogosto klasificira pravilno, primeri takšnih so: *biti_zatisniti_oko*, *biti_moč* in *biti_žal*. Med pogostimi napakami klasifikacije negativnih primerov izstopajo skupine besed, ki vsebuje veznike, kot so: *glede_na_zdajšnji*, *ki_štrleti_iz*, *na_položaj_desen*, *na_hitro_povedati* ali *in_tam_živeti*. Število besed v skupni samo po sebi nima vpliva na uspešnost klasifikacije z metodo RF. Prostor vektorskih vložitev je težko predstavljiv, zato ne moremo natančno določiti vzrokov za razlike med vektorji posameznih skupin besed, ki vodijo do primerov napačne klasifikacije.

Da bi preverili, če je razlika učinkovitosti algoritmov v obeh množicah res posledica atributov *N1, N2* in *N3*, za katere smo ocenili, da so najpomembnejši pri klasifikaciji v množici VID ccKRES, smo učenje ponovili z množicama, kjer smo te tri atribute odstranili. Rezultati so prikazani v tabeli 5.4.

Zaradi izgube treh pomembnih atributov se pri obeh množicah znižajo ocene vseh algoritmov. Če primerjamo razlike med ocenami algoritmov znotraj posamezne množice, te ohranijo podobne lastnosti kot v tabeli 5.3. Za razliko od ocen v tabeli 5.3 so tu algoritmi uspešnejši na množici VID ccGigafida kot na VID ccKRES. Razlika je posledica večje pomembnosti preostalih atributov pri klasifikaciji z izbranimi algoritmi v množici VID ccGigafida.

		CA	Senzitivnost	Specifičnost	AUC
Podatkovna množica ccKRES	SVM	0.604	**0.719**	0.489	0.63
	RF	**0.637**	0.64	**0.634**	**0.7**
	LR	0.608	0.612	0.603	0.62
Podatkovna množica ccGigafida	SVM	0.617	**0.708**	0.526	0.657
	RF	**0.653**	0.637	**0.67**	**0.748**
	LR	0.614	0.655	0.573	0.655

Tabela 5.4: Ocene izbranih algoritmov strojnega učenja na podatkovnih množicah pridobljenih iz korpusov ccKRES in ccGigafida brez atributov *N1, N2* in *N3*.

Iz tega sklepamo, da z večanjem slovarja naučenih besednih vektorjev in množice VID izbrani pristop klasifikacije ne bi odpovedal.

5.3 Čas izvajanja postopka

Učenje vektorskih vložitev in gradnjo podatkovnih množic smo izvajali na sistemu s procesorjem Intel Core i7-2600 in 16 GB pomnilnika. Učenje vektorskih vložitev na korpusu ccKRES traja približno enako časa za posamezne besede in za skupine besed, za oboje približno 5 minut. Pri korpusu ccGigafida se je ta čas podaljšal na 52 minut pri učenju vektorskih vložitev za posamezne besede in 35 minut za skupine besed. Glavni vpliv na čas učenja vektorskih vložitev ima število vseh besed v besedilu. Teh je v korpusu ccKRES za posamezne besede približno 10 milijonov, za skupine besed pa približno 3 milijoni. V korpusu ccGigafida je približno 100 milijonov posameznih besed, skupin besed pa 37 milijonov. Velikost slovarja na čas učenja vektorskih vložitev nima tako velikega vpliva. Slovar v korpusu ccGigafida vsebuje 207.319 posameznih besed in 721.160 skupin besed.

Zaradi računanja značilk je časovno zahteven tudi proces gradnje podatkovne množice. Glavni vzrok za to je, da je potrebno za vsak izbran vektor

skupine besed izračunati kosinusno razdaljo do vseh ostalih vektorjev v slovarju, kar pomeni, da na hitrost izvajanja najbolj vplivata velikost slovarja skupin besed in dolžina naučenih vektorjev. Gradnja podatkovne množice VID ccKRES je trajala približno 20 minut, kar je povprečno okoli 4 sekunde za posamezno skupino, podatkovne množice VID ccGigafida pa okoli 250 minut, kar je približno 36 sekund za posamezno skupino. Pri računanju značilk se nismo osredotočali na optimizacijo, zato bi bilo čas potreben za ta korak možno zmanjšati.

Celotno metodo bi lahko preizkusili tudi na večji množici podatkov, kjer bi pozitivne primere VID pridobili iz vira, kot je frazeološki leksikon. Čas računanja bi se podaljšal, če bi uporabili večji slovar vektorskih vložitev naučen na večjih korpusih. Metodo bi lahko pospešili tako, da bi odstranili izračun značilk *N1, N2* in *N3*. To bi odstranilo potrebo po računanje kosinusne razdalje do vsakega od vektorjev skupin besed v slovarju, kar bi bistveno zmanjšalo čas potreben za izračun vseh značilk, vendar bi nekoliko zmanjšalo uspešnost klasifikacije. V splošnem sicer ocenjujemo, da je predlagana metoda uporabna za paketno procesiranje besedil, za sprotno prepoznavanje VID v realnem času pa je prepočasna.

Poglavje 6

Zaključek

V diplomski nalogi smo preizkusili metodo prepoznave VID na podlagi vektorskih vložitev. Zgradili smo vektorske vložitve za posamezne besede in skupine besed iz slovenskih korpusov ccKRES in ccGigafida. Z uporabo dobljenih vektorjev smo skonstruirali značilke za primere VID izluščene iz korpusa PARSEME in za nabor naključnih skupin besed, ki smo jih uporabili pri gradnji dveh podatkovnih množic. Na teh množicah smo preizkusili in ocenili uspešnost klasifikacije VID s tremi klasifikacijskimi metodami, ki so bile vse dokaj uspešne. Klasifikacijo smo preizkusili tudi na zmanjšani množici, kjer smo odstranili tri atribute, za katere smo ocenili, da so pri klasifikaciji najpomembnejši. Pri tem so ocene klasifikacijskih metod bistveno bolj padle pri manjši množici VID ccKRES.

Naši rezultati kažejo, da je razvita metoda dokaj uspešna. Metoda je časovno precej zahtevna, prepoznava pa deluje samo na skupinah besed za katere imamo naučene vektorje. Metodo bi se dalo izboljšati. Število naučenih vektorjev bi lahko povečali z gradnjo vektorskih vložitev na večjem korpusu besedil, kot je Gigafida (1.2 milijarde besed). Z večanjem slovarja bi se sicer povečal čas, potreben za računanje značilk posamezne skupine besed. Smiselno bi bilo preizkusiti uporabo znakovnih n-gramov, ki knjižnici fastText omogočijo delovanje tudi za nepoznane besede, ki so podobne že znanim. Z uporabo virov, kot je frazeološki leksikon, bi lahko zgradili večjo množico,

ki bi verjetno izboljšala uspešnost klasifikacije. Preveriti je potrebno tudi uspešnost klasifikacije VID, ki jih v besedilu lahko na več delov ločujejo druge besede (*ni narobe - ni sicer nič narobe*). Za to bi potrebovali vektorske vložitve skupin, ki so daljše od štirih besed. Možno izboljšavo ponuja tudi izdelava novih značilk. V kolikor bi se pristop uporabilo za označevanja VID v besedilu, bi bilo smiselno primerjati rezultate z ostalimi postopki, ki so bili razviti v okviru akcije PARSEME.

Glavna prednost razvitega pristopa je, da bi lahko z uporabo manjše podmnožice znanih VID, kot VID prepoznali tudi nove, popolnoma drugačne skupine besed, za katere imamo zgrajene vektorske vložitve. Kljub obetajočim rezultatom pa ima naša implementacija še nekaj pomanjkljivosti, ki bi jih bilo treba odpraviti pred poskusom delovanja v praksi.

Literatura

[1] Špela Arhar and Vojko Gorjanc. *Korpus FidaPLUS: nova generacija slovenskega referenčnega korpusa*. Slavistično društvo Slovenije, 2007.

[2] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[3] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

[4] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[5] Tomaž Erjavec and Nataša Logar Berginc. Referenčni korpusi slovenskega jezika (cc) Gigafida in (cc) Kres. In *Zbornik Osme konference Jezikovne tehnologije. Ljubljana: Jožef Stefan Institute*, pages 57–62, 2012.

[6] fasttext. Dosegljivo: `https://fasttext.cc`. [Dostopano: 22. 1. 2019].

[7] Polona Gantar, Špela Arhar Holdt, Jaka Čibej, Taja Kuzman, and Teja Kavčič. Glagolske večbesedne enote v učnem korpusu ssj500k 2.1. In *Zbornik konference Jezikovne tehnologije in digitalna humanistika 2018*, pages 85–92, 2018.

[8] Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

[9] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[10] Natalia Klyueva, Antoine Doucet, and Milan Straka. Neural networks for multi-word expression detection. In *Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017)*, pages 60–65, 2017.

[11] Igor Kononenko and Matjaz Kukar. *Machine learning and data mining*. Horwood Publishing, 2007.

[12] Marko Kononenko, Igor in Robnik-Šikonja. *Inteligentni Sistemi*. Založba FR in FRI, Ljubljana, 2010.

[13] Simon Krek, Kaja Dobrovoljc, Tomaž Erjavec, Sara Može, Nina Ledinek, Nanika Holz, Katja Zupan, Polona Gantar, and Taja Kuzman. Training corpus ssj500k 2.0, 2017. Slovenian language resource repository CLARIN.SI.

[14] Nataša Logar, Tomaž Erjavec, Simon Krek, Miha Grčar, and Peter Holozan. Written corpus ccGigafida 1.0, 2013. Slovenian language resource repository CLARIN.SI.

[15] Nataša Logar, Tomaž Erjavec, Simon Krek, Miha Grčar, and Peter Holozan. Written corpus ccKres 1.0, 2013. Slovenian language resource repository CLARIN.SI.

[16] Nataša Logar, Miha Grčar, Marko Brakus, Tomaž Erjavec, Špela Arhar Holdt, Simon Krek, and Iztok Kosem. *Korpusi slovenskega jezika Gigafida, KRES, ccGigafida in ccKRES: gradnja, vsebina, uporaba*. Trojina, zavod za uporabno slovenistiko, 2012.

[17] Alfredo Maldonado, Lifeng Han, Erwan Moreau, Ashjan Alsulaimani, Koel Chowdhury, Carl Vogel, and Qun Liu. Detection of verbal multi-word expressions via conditional random fields with syntactic dependency features and semantic re-ranking. In *Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017)*, pages 114–120. Association for Computational Linguistics, 2017.

[18] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[19] Parseme shared task. Dosegljivo: `https://typo.uni-konstanz.de/parseme/`. [Dostopano: 22. 1. 2019].

[20] Carlos Ramisch, Silvio Cordeiro, Agata Savary, Veronika Vincze, Verginica Mititelu, Archna Bhatia, Maja Buljan, Marie Candito, Polona Gantar, Voula Giouli, et al. Edition 1.1 of the PARSEME shared task on automatic identification of verbal multiword expressions. In *the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 222–240, 2018.

[21] Sebastian Ruder. On word embeddings - part 1. Dosegljivo: `http://ruder.io/word-embeddings-1/`. [Dostopano: 22. 1. 2019].

[22] Ivan A Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. Multiword expressions: A pain in the neck for NLP. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 1–15. Springer, 2002.

[23] Agata Savary, Manfred Sailer, Yannick Parmentier, Michael Rosner, Victoria Rosén, Adam Przepiórkowski, Cvetana Krstev, Veronika Vincze, Beata Wójtowicz, Gyri Smørdal Losnegaard, et al. Parseme–parsing and multiword expressions within a European multilingual network. In *7th Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics (LTC 2015)*, 2015.