Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Kristjan Voje

Avtomatska izdelava vezljivostnih vzorcev za slovenske glagole

diplomsko delo

univerzitetni študijski program
prve stopnje
računalništvo in informatika

Mentor: prof. dr. Marko Robnik Šikonja
Somentorica: doc. dr. Apolonija Gantar

Ljubljana, 2018

To delo je objavljeno pod licenco Creative Commons Priznanje avtorstva-Nekomercialno-Deljenje pod enakimi pogoji 4.0 Mednarodna.

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Glagoli se v stavkih vežejo z različnimi dopolnili, ki povedo, npr. kdo je dejanje storil, komu ga je namenil, s kom ali čim in kje je dejanje storil ipd. Na to lahko gledamo, kot da glagol ustreza predikatu, ki ima različne argumente. Vezljivost glagolov je pomembna za semantično členitev besedil in je ena od tehnologij, ki so potrebne za razumevanje naravnega jezika. Mnogi jeziki že imajo izdelane strojno berljive slovarje glagolske vezljivosti. Proučite problem avtomatske konstrukcije vzorcev glagolske vezljivosti in izdelajte prototip. Na podlagi obstoječih rešitev za prikaz glagolske vezljivosti, predvsem češke in hrvaške, izdelajte prikazovalnik glagolske vezljivosti, ki upošteva tudi udeleženske vloge. Za ločevanje med različnimi pomeni glagolov uporabite in preizkusite nekaj inačic Leskovega algoritma.

Zahvaljujem se svoji družini za vso podporo, ki sem jo prejel med študijem. Posebej bi se rad zahvalil svojima mentorjema prof. dr. Marku Robniku Šikonji in doc. dr. Apoloniji Gantar za potrpežljivost in strokovno vodenje pri izdelavi te diplomske naloge. Hvala!

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
NLTK	Natural Language Toolkit	Python knjižnica za delo z naravnim jezikom
SSKJ	Dictionary of Literary Slovene	Slovar slovenskega knjižnega jezika
WSD	Word Sense Disambiguation	razdvoumljanje besednega pomena
NLP	Natural Language Processing	obdelava naravnega jezika
PDT	Prague Dependency Treebank	Praška odvisnostna drevesnica
POS	Part of Speech	besedna vrsta
MSD	Morpho-Syntactic Description	oblikoskladenjske oznake oz. opis

Povzetek

Povzetek: Za računalniško obdelavo naravnega jezika so ključnega pomena veliki označeni učni korpusi. Ko obravnavamo manjše količine podatkov, lahko te obogatimo s podrobnejšo analizo strukture jezika. Lastnost naravnega jezika, ki jo bomo obravnavali v diplomskem delu, je vezljivost. Vezljivost se nanaša na pomen povedi. Nosilci vezljivosti so pogosto glagoli, lahko pa tudi pridevniki in samostalniki. Določenemu pomenu nosilca vezljivosti v teoriji pripada določen vezljivostni vzorec. Vezljivostni vzorci so računalniško dobro berljivi in vsebujejo dovolj informacij za razdvoumljanje pomena nosilca vezljivosti.

Naše delo temelji na korpusu ssj500k 2.1 [15]. Dobra polovica korpusa vsebuje povedi z ročno označenimi udeleženskimi vlogami, iz katerih smo razbrali vezljivostne vzorce. Pripravili smo program, ki uporabniku omogoča interaktiven pregled vezljivostnih vzorcev v korpusu.

Različni pomeni istega glagola tvorijo različne vezljivostne vzorce. Nad stavki v korpusu smo preizkusili nabor algoritmov za gručenje z namenom iskanja vezljivostnih vzorcev, značilnih za določeni pomen glagola. Implementirali smo tri različice Leskovega algoritma in dve različici algoritma k-voditeljev. Podatke za Leskov algoritem smo črpali iz leksikona SloWNet [10] in slovarja SSKJ [3].

Ključne besede: vezljivostni vzorci, vezljivost, glagol

Abstract

Abstract: Natural language processing greatly depends on a sufficient amount of training data. When handling with smaller datasets, we can enrich our data by analyzing the semantic structure of the language. In our thesis, we will be working with valency. Valency carries information about the meaning of a sentence. While valency is usually a feature of verbs, we can also observe it in adjectives and nouns. Valency forms valency patterns around carriers. In theory, each sense of the valency carrier should form a distinguishable valency pattern. Valency patterns have a small feature space and are fit for training machine learning algorithms. They contain enough information to distinguish the sense of the valency carrier.

Our work is based on corpus ssj500k 2.1 [15]. Over half of the corpus contains hand-annotated semantic roles from which we extracted valency patterns. We built a program for listing and analyzing the valency patterns.

In theory, different verb senses form different valency patterns. We tested a number of clustering algorithms on the corpus sentences. The goal was to cluster the valency frames, based on similar senses, and to find sense specific valency patterns. We implemented three versions of Lesk algorithm and two versions of k-means algorithm. We used data from SloWNet [10] and SSKJ [3] for the knowledge based Lesk algorithms.

Keywords: valency frame, valency, verb

Poglavje 1

Uvod

V zadnjih letih se je občutno povečalo število aplikacij, ki znajo prepoznavati naravni jezik, tako pisni kot govorjeni. Pametni telefoni prepoznavajo uporabnikove govorjene ukaze, spletni prevajalniki so zmožni prevajanja celotnih besedil in na spletu se pojavljajo roboti, ki se znajo pogovarjati z uporabniki klepetalnic.

Za računalniško obdelavo naravnega jezika je pomembno razdvoumljanje pomenov posameznih besed. Pomen besede je odvisen od konteksta ali besed, ki jo obdajajo. Preprost model za določanje pomena besede so lahko vse besede v povedi. V slovenščini se besede močno sklanjajo, kar privede do prevelike množice značilk. Za računalniško branje slovenskega jezika potrebujemo model, neodvisen od stavčne strukture in sklanjatev. Primerna je ponazoritev jezika v obliki vezljivostnih vzorcev.

Teorija vezljivosti pravi, da lahko pomen stavka razberemo iz glavne besede, ki je v večini primerov glagol, ter iz vezljivostnih vzorcev, ki jih ta tvori. Določeni pomen glagola zahteva okoli sebe določeno število t. i. udeleženskih vlog. Te udeleženske vloge tvorijo vezljivostne vzorce. Za posamezni glagol bomo poskušali poiskati povezavo med vezljivostnimi vzorci in pomeni glagola.

V diplomskem delu bomo predstavili orodje za pregledovanje vezljivostnih vzorcev v korpusu ssj500k 2.1 [15]. Vezljivostne vzorce je potrebno

združiti v pomenske skupine. S pomočjo pomenskih skupin lahko odkrijemo vezljivostne vzorce, značilne za določene pomene glavne besede.

Pomenske skupine smo poskusili določiti z algoritmi strojnega učenja. Implementirali smo tri različice Leskovega algoritma ter dve različici k-voditeljev. Podatke za slovarsko podprto strojno učenje smo črpali iz slovarja SSKJ [3] in leksikona SloWNet [10].

V naslednjem poglavju predstavimo jezikoslovno teorijo vezljivosti in udeleženskih vlog. V tretjem poglavju opišemo sorodne rešitve za hrvaški in češki jezik, po katerih smo se zgledovali v našem delu. Četrto poglavje je predstavitev naše rešitve. Predstavimo spletno aplikacijo za pregledovanje vezljivostnih vzorcev ter opišemo algoritme strojnega učenja, ki smo jih preizkusili nad vhodnim korpusom. Četrto poglavje zaključimo s pregledom dodatnih algoritmov za razdvoumljanje, ki bi jih lahko uporabili za naš problem. V petem poglavju evalviramo rezultate algoritmov strojnega učenja. Šesto poglavje predstavlja zaključek in povzetek diplomskega dela.

Poglavje 2

Vezljivost

Vezljivost ali mednarodno valenca je lastnost besede, da nase veže določeno število obveznih skladenjskih mest, imenovanih udeleženske vloge. Vezljivost lahko opazujemo v t. i. globoki zgradbi povdei, ki vsebuje podatke o jezikoslovnem pomenu. V nekateri tuji literaturi se vezljivostni vzorci nahajajo v t. i. tektogramatični ravni povedi [18].

Nosilci vezljivosti so najpogosteje glagoli, a poznamo tudi vezljivost pridevnikov, izglagolskih samostalnikov ter izpridevniških samostalnikov. V diplomskem delu smo analizirali vezljivostne vzorce glagolov ter majhno število vezljivostnih vzorcev pridevnikov. V nadaljevanju se bodo opisi vezljivosti nanašali na vezljivost glagolov. Vezljivost določa obvezna skladenjska mesta, ki jih zapolnjujejo določila v določeni slovnični obliki. Obvezna skladenjska mesta nam napove pomenska usmerjenost glagola. Udeleženska vloga je za določene pomene glagola obvezna in za druge pomene neobvezna. Z razvrstitvijo vseh možnih udeleženskih vlog v okviru posameznega pomena glagola lahko opazujemo t. i. pomenske skupine glagolov [38].

Za primer vzamemo vezljivostne vzorce za glagol *delati*, predlagane v knjigi *Vezljivost v slovenskem jeziku* [38]. Vezljivostne vzorce smo poenostavili tako, da vsebujejo le oznake, ki se nanašajo na pomensko raven povedi. Navedeni so primeri povedi ter vezljivostnih vzorcev, ki jih opazujemo v posameznem povedi. Vezljivostni vzorci so v našem primeru zgrajeni iz

udeleženskih vlog, naštetih v tabeli 2.1. Medsebojna povezanost ter obveznost ali neobveznost udeleženskih vlog je ponazorjena z operatorji v tabeli 2.2.

Pomenske skupine glagola *delati*:

1. 'delovanje – zavestno uporabljanje telesne in duševne energije'

Poved	**Vezljivostni vzorec**
Motorji delajo.	No + Glagol
Strup dela hitro.	Pv + Glagol /+ Pr/Ra/Vs // N
Srce je začelo delati.	V + Glagol /+ N/Č/M

2. 'razmerje delovanja'

Poved	**Vezljivostni vzorec**
S knjigo delajo kakor s cunjo.	Pv + Glagol + Pr/Ra/Vs /+ N
Dela s pravopisom.	V + Glagol + Pr/Vs

3. 'rezultativnost ali ciljnost dejanja'

Poved	**Vezljivostni vzorec**
Delajo gumi (iz kavčuka).	V + Glagol + R (+ Pr/IM)
Delajo (naročniku) za denar.	V + Glagol + (Pre) + C

4. 'delovanje, dejanvnost, lastnost ali odnos'

Poved	**Vezljivostni vzorec**
Delajo korake.	V + Glagol + L
Dela kot skladiščnik.	V + Glagol + L
Cvetje dela sobo lepše.	Pv + Glagol + Pr + L
Dela (mu) veselje.	Pv + Glagol + (Ra) + L

Okrajšava	Udeleženska vloga
No	nosilec dejanja/dogajanja/procesa/stanja
Pv	povzročitelj dejanja
V	izvor/vršilec dejanja
Pr	prizadeto z dejanjem
Ra	razmerje z dejanjem
Vs	vsebina dejanja
M	mesto dejanja
Č	čas dejanja
N	način dejanja
S	sredstvo
R	rezultat dejanja
IM	izhodiščno mesto dejanja
L	lastnost

Tabela 2.1: Okrajšave udeleženskih vlog.

Operator	Opis operatorja
//	skladenjsko obvezni modifikator
+	sosledje stavčnih členov
/	izbirna različica
()	neobvezno vezljivo določilo

Tabela 2.2: Operatorji.

2.1 Udeleženske vloge

Označevanje udeleženskih vlog (ang. Semantic Role Labeling – SRL) je eden od podproblemov obdelave naravnega jezika. Udeleženske vloge predstavljajo pomensko raven povedi in morajo biti morajo čim bolj neodvisne od stavčne morfološke in skladenjske zgradbe. Kljub specifičnosti za posamezni jezik obstaja velika težnja po medjezikovni kompatibilnosti udeleženskih vlog.

Slovenski raziskovalci [16] so z udeleženskimi vlogami ročno označili približno polovico korpusa ssj500k 2.1 [15]. Oznake in opisi udeleženskih vlog so izpeljani iz funkcijskega generativnega pristopa Praške odvisnostne drevesnice [22]. Podobne oznake uporabljata hrvaški CROVALLEX [1] in češki PDTVallex [13], ki sta opisana v 3. poglavju.

Udeleženske vloge imajo lahko vlogo delovalnika ali okoliščine. Za primer vzemimo glagol narediti: *kdo* naredi *komu kaj* (*kdaj*, *kje*, *kako*, *zakaj*). *Kdo*, *komu* in *kaj* so realizacija delovalniških udeležencev, *kdaj*, *kje*, *kako* in *zakaj* pa realizacija okoliščinskih udeležencev. Določeni pomen glagola lahko predvideva določene udeleženske vloge, ki pa niso nujno realizirane. Medtem ko zgornji primer lahko predvideva vse delovalnike, okoliščine niso nujno potrebne za izražanje pomena glagola. Delovalniki so lahko predvideni in nerealizirani. Primer za to je poved "Delam nalogo.", v kateri je vršilec dejanja navzoč ("jaz"), a ni realiziran.

V tabeli 2.3 naštejemo udeleženske vloge, predlagane v zgoraj opisanem projektu. Udeleženske vloge v korpusu ssj500k 2.1 so pripravljene kot ogrodje za označevanje semantičnih vlog v slovenskem in hrvaškem jeziku [11].

Udeleženska vloga	**Opis**
ACT	delujoči udeleženci, povzročitelji ali nosilci dejanja
PAT	prizadeti predmet dejanja
REC	prejemnik, posredni udeleženec dejanja; nedelovalniški udeleženec, ki mu je dejanje v škodo ali v prid
ORIG	izhodišče, izvor/vir/povod dejanja
RESLT	učinek, rezultat, cilj dejanja
TIME	konkretni trenutek ali interval dejanja; trajanje stanja, dejanja
DUR	koliko časa
FREQ	frekvenca dejanja
LOC	konkretna lokacija, kraj, mesto dejanja; kje
SOURCE	začetna točka v prostoru; od kod
GOAL	končna točka v prostoru; kam
AIM	namen dejanja; čemu, s kakšnim namenom
CAUSE	vzrok dejanja; zakaj
CONTR	nepričakovana posledičnost dejanja; kljub čemu
COND	pogoj za obstoj dejanja ali dogodka
REG	glede na, primerjava
ACMP	predmet, oseba ali dogodek, ki spremlja dejanje ali druge udeležence
RESTR	izjema, omejitev
MANN	načinovna lastnost dejanja, rezultat ob koncu dejanja
MEANS	sredstvo ali orodje za izvedbo dejanja
QUANT	količina, razlika
MWPRED	zveze z nedoločniki
MODAL	zveze glagola *biti* + modalnega prislova/pridevnika
PHRAS	pomensko neprozorne zveze

Tabela 2.3: Udeleženske vloge, predlagane v članku [16]. Udeleženske vloge so razporejene v skupine: delovalniki, okoliščine in glagolske zveze.

Poglavje 3

Obstoječe rešitve

Pri izgradnji vezljivostnih vzorcev za slovenske glagole smo se zgledovali po dveh podobnih projektih: PDT-Vallex in CROVALLEX. PDT-Vallex je češki leksikon vezljivosti, ki je nastal na osnovi Praške odvisnostne drevesnice. CROVALLEX je hrvaški leksikon vezljivosti, ki je nastal po vzoru češkega Vallexa. V spodnjih razdelkih podamo kratek opis obeh leksikonov.

3.1 PDT-Vallex

Na češkem se izvaja dolgoročni projekt sintaktične in semantične anotacije dela češkega nacionalnega korpusa, rezultat katere je nastanek Praške odvisnostne drevesnice (ang. The Prague Dependency Treebank ali PDT [13]).

V diplomskem delu bomo opisali PDT-Vallex 2.0, ki je nastal na osnovi Praške odvisnostne drevesnice 2.0. To je prva različica Praške odvisnostne drevesnice, ki vsebuje označene globoke stavčne strukture, iz katerih lahko razberemo vezljivostne vzorce.

Praška odvisnostna drevesnica od različice 2.0 naprej predvideva označevanje na treh nivojih ali ravneh:

Morfološka raven obsega zaporedje členov, ki predstavljajo posamezne besede in ločila. Členi vsebujejo podatek o lemi besede ter njene oblikoslovne oznake. Lema je kanonska in nespregana oblika neke besede.

Analitična raven obsega drevesno strukturo, katere vozlišča so členi, ki sestavljajo izvorno poved. Nobeno vozlišče ni dodano ali izvzeto. Kjer sintaktična pravila dovoljujejo, so med vozlišči povezave, ki opisujejo sintaktično odvisnost členov.

Tektogramatična raven predstavlja globoko, pomensko strukturo stavka. Kot analitična raven je tudi tektogramatična raven predstavljena v obliki drevesne strukture. Od analitične ravni se razlikuje v tem, da vsebuje le vozlišča, ki so pomembna za pomen povedi. Tektogramatična raven vsebuje le t. i. avtosemantične besede (to so polno- ali predmetnopomenske besede). Povezave med vozlišči predstavljajo pomensko odvisnost členov. Iz te strukture lahko razberemo vezljivostne vzorce.

Najpomembnejša lastnost vozlišč na tej ravni je udeleženska vloga (v PDT-Vallex imenovana *funktor*). Udeleženska vloga nam pove, kakšno pomensko vlogo ima vozlišče v razmerju do iztočnice. Iztočnica je lahko glagol, pridevnik ali samostalnik [12].

3.1.1 Struktura PDT-Vallexa

Ob prvotni izgradnji leksikona je PDT-Vallex vseboval 5 262 glagolov, 4 090 samostalnikov ter 831 pridevnikov. Leksikon je bil označen kot PDT-Vallex 2.0.

Leksikon je sestavljen iz iztočnic. Vsaki iztočnici pripada eden ali več vezljivostnih vzorcev. V teoriji ima vsak pomen iztočnice svoj vezljivostni vzorec. Vezljivostni vzorec sestavljajo mesta, ki jih zapolnjujejo udeleženske vloge. Posamezni vezljivostni vzorec napoveduje določene obvezne in neobvezne udeleženske vloge [12].

3.1.2 Valenčni okvir v PDT-Vallexu

Na sliki 3.1 vidimo primer vezljivostnih vzorcev v leksikonu PDT-Vallex. Vsaka vrstica predstavlja svojo lemo. Lema zastopa osnovno obliko iztočnice. V našem primeru vsaki lemi pripada po en vezljivostni vzorec. Vezljivostni

vzorec vsebuje mesta (*Slot$_1$, Slot$_2$ ... Slot$_n$*), ki jih zapolnjujejo udeleženske vloge. Udeleženski vlogi sta dodana podatek o obveznosti *(obl)* ali izbirnosti *(opt)* ter sklon [12].

Word / Sense	Valency Frame				
	$Slot_1$	$Slot_2$	$Slot_3$	...	$Slot_n$
dát	ACT_{obl}(Nom)	PAT_{obl}(Acc)	$ADDR_{obl}$(Dat)		
dopis	ACT_{obl}(Poss/Gen)	$ADDR_{obl}$(Dat)			
plný	PAT_{obl}(Gen)				

Slika 3.1: Vezljivostni vzorci v leksikonu PDT-Vallex 2.0 za besede *dát (dati)*, *dopis (pismo)* in *plný (poln)*.

3.2 CROVALLEX

CROVALLEX 200.8 je valenčni leksikon hrvaških glagolov [29]. Leksikon vsebuje 1739 glagolov, ki jim pripada 5118 vezljivostnih vzorcev. Vsebuje tudi 173 sintaktično-semantičnih razredov, ki so bili iz baze VerbNet [32] prilagojeni za hrvaški jezik. Leksikon ne vsebuje samostalnikov in pridevnikov. Glagoli so bili izbrani iz Hrvaškega frekvenčnega slovarja [24]. Za leksikon so bili izbrani glagoli s frekvenco 11 ali več.

Leksikon je na voljo kot datoteka XML, dostopen pa je tudi preko spletnega brskalnika [1].

3.2.1 Struktura vnosa v CROVALLEX

Na sliki 3.2 je prikazan vnos iztočnice *napisati*. Iztočnici je v primeru homonima ali homografa dodana rimska številka. Homonima sta dve lemi z istim črkovanjem ter naglaševanjem, homografa pa sta lemi z istim črkovanjem in različnim naglaševanjem.

Iztočnici pripada eden ali več vezljivostnih vzorcev. Vsak vzorec predstavlja po en pomen iztočnice. Najprej so našteti vzorci, ki predstavljajo glavne

Slika 3.2: Vnos iztočnice *napisati* v leksikonu CROVALLEX.

pomene iztočnice, sledijo pa vzorci, ki prestavljajo redkeje uporabljene pomene, na primer idiome ali stalne besedne zveze.

Razmerje med glagolom ter notranjimi udeleženci je v CROVALLEX-u ponazorjeno s t. i. globokimi strukturami. Teoretično ozadje leksikona je podobno kot pri leksikonu PDT-Vallex, zato so uporabljene podobne udeleženske vloge. Struktura vezljivostnega vzorca v CROVALLEX-u je podobna strukturi vzorca v PDT-Vallexu. Vzorec je sestavljen iz vsaj ene ali več udeleženskih vlog. Posamezni udeleženski vlogi je dodan seznam oblikoslovnih oblik, ki predstavljajo t. i. površinsko slovnično strukturo. Udeleženske vloge imajo v leksikonu oznako za obvezno ali neobvezno prisotnost v vezljivostnem vzorcu.

Udeleženska vloga lahko pri določenem pomenu implicitno določa oblikoslovno obliko. Če je možnih oblikoslovnih oblik več, je udeleženski vlogi dodan seznam teh oblik.

Določeni pomen glagola ima točno določeno sintaktično obliko vezave udeleženskih vlog. Sprememba ene od udeleženskih vlog nam da nov vezljivostni vzorec.

V hrvaškem jeziku obstajajo tri razmerja med vezljivostjo ter pomenom povedi:

- Sprememba v pomenu glagola ne povzroči spremembe vezljivosti: *Ribolovci mrežom plaše ribe. - Ribiči plašijo ribe v mrežo.* V tem primeru *plaše* pomeni 'loviti, preganjati'. *Surla je plašio djecu paklom. - Surla je s peklom strašil otroke.* V tem primeru *plašio* pomeni 'strašiti'.

- Sprememba v vezljivosti glagola ne spremeni pomena: povedi *Marko pliva. - Marko plava.* ter *Marko pliva rekord. - Marko plava rekord.* imata podoben pomen ('plavati') kljub različnima vezljivostnima vzorcema.

- Sprememba pomena povzroči spremembo vezljivosti: povedi *Zagreb pije vodu iz podzemlja. - Zagreb pije vodo iz podzemlja.* ter *Juraj pije nekontrolirano. - Juraj pije nekontrolirano.* imata različna pomena (prvi je 'pitje', drugi je 'alkoholiziranost') ter različna vezljivostna vzorca.

V hrvaškem jeziku pogosto manjka prvi notranji udeleženec AGT, saj je lahko del glagola (*npr. Radim./Ja radim. - Delam.*). To pravilo velja tudi za slovenščino in ostale oblikoslovno bogate jezike. CROVALLEX to upošteva in označuje prvi argument kot AGT_0/1, kar pomeni, da je prvi argument lahko ali samostojna beseda ali pa del glagola.

V hrvaščini so površinske oblikoslovne strukture povezane z določenimi globokimi strukturami. AGT je pogosto vezan na imenovalnik, PAT na tožilnik, REC na dajalnik, ORIG se pogosto veže z od + tožilnik, medtem ko se RESL veže z na + tožilnik ter u + tožilnik [29].

Poglavje 4

Opis rešitve

Izdelali smo spletno aplikacijo, ki služi kot pregledovalnik vezljivostnih vzorcev slovenskih glagolov. Podatke za aplikacijo smo črpali iz učnega korpusa ssj500k 2.1 [15] (razdelek 4.1). Aplikacija uporabniku omogoča interaktiven pregled vezljivostnih vzorcev glagolov in pridevnikov, razvrščenih glede na iztočnico ali glede na vsebovane udeleženske vloge. Vezljivostni vzorci vsebujejo izvorno poved ter MSD oznake posameznih besed v povedi. Uporabnik lahko pregleduje vezljivostne vzorce, združene glede na skupne udeleženske vloge ali glede na skupni pomen povedi.

V diplomskem delu smo preizkusili skupino algoritmov strojnega učenja za razdvoumljanje pomenov povedi (razdelek 4.3). Algoritmi niso dosegli natančnosti za praktično uporabo, zato smo uporabnikom spletne aplikacije omogočili ročno označevanje pomenov povedi. Ročno označeni pomeni povedi bodo lahko v prihodnosti služili kot učni korpus za razdvoumljanje pomenov povedi.

V prvem razdelku bomo opisali učni korpus, iz katerega smo črpali podatke za spletno aplikacijo. V drugem razdelku bomo opisali funkcionalnost in uporabniško izkušnjo spletne aplikacije. V tretjem razdelku bomo podali arhitekturo in tehnične specifikacije spletne aplikacije. V zadnjem razdelku bomo opisali metode strojnega učenja za razdvoumljanje pomenov besed.

4.1 Učni korpus

Orodje za pregledovanje vezljivostnih vzorcev črpa podatke iz učnega korpusa ssj500k 2.1 [15]. Korpus ssj500k 2.1 je bil zgrajen iz korpusov jos100k in jos 1M, ki sta nastala v okviru projekta JOS [2].

Korpus vsebuje 27 829 stavkov, sestavljenih iz skupno 586 248 pojavnic, od tega 500 293 besed. Glagole predstavlja skupaj 15 988 pojavnic. Delež glagolskih lem, ki se v korpusu pojavijo le enkrat, je 47,5 %. Najpogostejše glagolske leme v korpusu so **biti** (7203 pojavitev), **imeti** (333 pojavitev) in **morati** (178 pojavitev). Zaradi nesorazmerno velike frekvence smo pri analizi korpusa izpustili glagol biti.

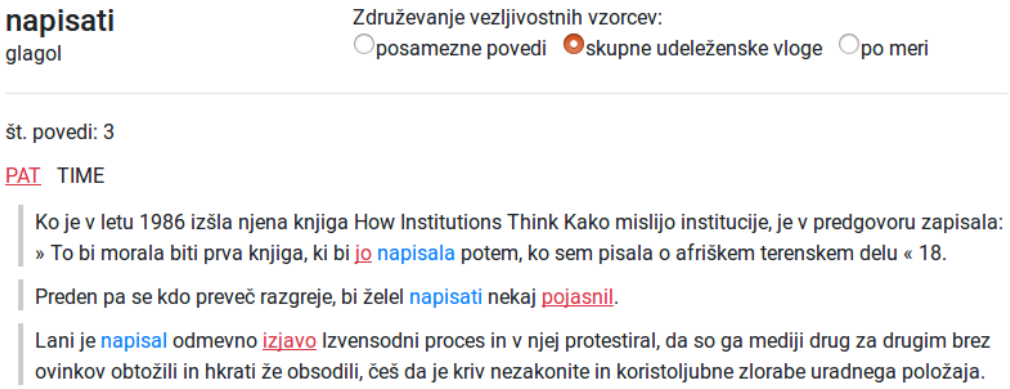
Pojavnice vsebujejo lemo, oblikoskladenjske oznake ter podatke o površinski in globinski zgradbi stavka. Približno četrtina korpusa vsebuje ročno označene udeleženske vloge [16].

Iz korpusa smo vzeli 5 030 povedi z označenimi udeleženskimi vlogami. Udeleženske vloge skupaj z iztočnicami tvorijo vezljivostne vzorce. V našem delu se osredotočamo na glagolske iztočnice, vključili pa smo tudi vezljivostne vzorce s pridevniškimi iztočnicami. Vezljivostne vzorce smo razvrstili v skupine z isto iztočnico. Rezultat je 2 252 skupin, od tega 1 724 skupin z glagolsko iztočnico in 528 skupin s pridevniško iztočnico.

4.2 Spletna aplikacija

Izdelali smo spletno aplikacijo, ki služi kot orodje za pregledovanje vezljivostnih vzorcev glagolov in nekaterih pridevnikov v korpusu ssj500k 2.1 [15]. Aplikacija na vhod prejme poljuben korpus, zgrajen po smernicah TEI [7]. Iz korpusa razbere vezljivostne vzorce ter jih interaktivno prikaže uporabniku (Slika 4.1). Uporabnik lahko pregleduje vezljivostne vzorce ter povedi, iz katerih so bili zgrajeni. Na voljo mu je pregled MSD oznak posameznih členov stavka. Implementirali smo osnovne funkcionalnosti za pregledovanje vezljivostnih vzorcev, ki jih vsebujeta orodji za pregledovanje CROVALLEX-a in PDT-Vallexa.

Uporabniku smo dodali možnost, da ročno definira pomenske skupine prikazanih glagolov in pridevnikov. Z uporabo aplikacije je mogoče zgraditi učno množico vezljivostnih vzorcev glagolov in pridevnikov, označeno s pomeni stavkov. Tovrstna učna množica bo uporabna za nadaljnje projekte obdelave naravnega jezika.



Slika 4.1: Vezljivostni vzorec glagola napisati.

4.2.1 Uporabniška izkušnja

Uporabnik lahko v meniju "Pregled" izbira iskanje po besedah ter iskanje po udeleženskih vlogah. Iskanje po udeleženskih vlogah uporabniku predstavi seznam vseh udeleženskih vlog. Poleg posamezne udeleženske vloge je navedeno število povedi, v katerih je ta udeleženska vloga prisotna. Privzeta nastavitev je iskanje po iztočnicah. Uporabniku se prikaže seznam glagolov in pridevnikov. Pridevniki so v seznamu od glagolov ločeni s končnico "_". Poleg vsake besede je število povedi, v katerih najdemo to besedo.

Ob kliku na besedo se v osrednjem oknu prikaže seznam okvirjev te besede. Posamezni okvir vsebuje seznam udeleženskih vlog ter eno ali več povedi. Prikazane udeleženske vloge so razporejene po naslednjem pravilu: prva je ACT, druga PAT, nato sledijo ostale udeleženske vloge, razporejene

po abecednem vrstnem redu.

Udeleženske vloge so interaktivno povezane s povedmi. Uporabnik lahko gre z miško nad udeleženske vlogo in ta se bo obarvala rdeče. Istočasno se bo rdeče obarvala beseda, ki ji je pripisana ta udeleženska vloga. Ob kliku na udeleženske vlogo bo par ostal rdeče obarvan tudi, ko uporabnik miško odmakne. Uporabnik lahko klikne na prazen prostor v okvirju in tako odstrani obarvanje udeleženske vloge in besede.

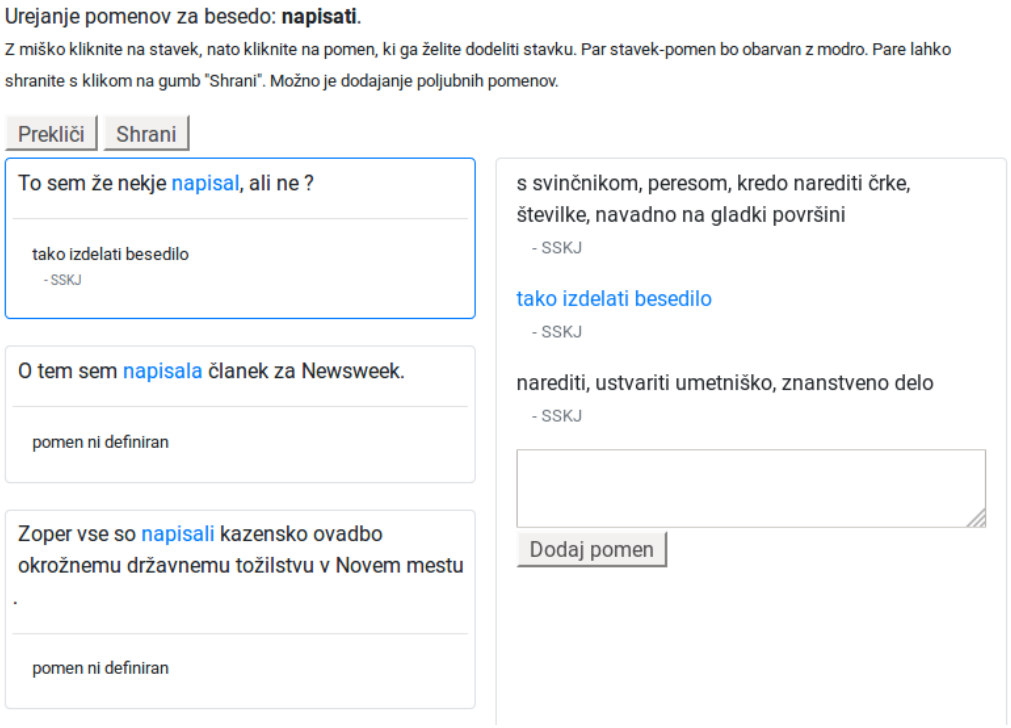
Uporabnik lahko s pomočjo možnosti "združevanje okvirjev" izbira način, na katerega se bodo okvirji združevali. Na voljo so trije načini združevanja okvirjev:

- **posamezne povedi:** naštete so vse povedi, ki jim pripada iztočnica. Vsaki povedi je dodan seznam udeleženskih vlog;
- **skupne udeleženske vloge:** povedi z enakim seznamom udeleženskih vlog se združijo v skupni okvir;
- **skupni pomen:** povedi se združijo po skupnih pomenih. Pomeni so vzeti iz SSKJ v spletnem portalu Fran [3]. Udeleženske vloge vseh združenih povedi se združijo v skupni seznam udeleženskih vlog. Ob izbiri načina "skupni pomen" se v okvirjih pokažejo informacije o pomenu povedi, ki so združene v posameznem okvirju. Privzeto povedi nimajo pomena ter spadajo v skupni okvir z oznako "pomen ni definiran". Uporabnik lahko s klikom na gumb "Uredi pomene" poljubno dodaja pomene posameznim povedim. Za urejanje pomenov je potrebna prijava, ki je mogoča s klikom na povezavo "Prijava". Pred prvo prijavo se mora uporabnik registrirati s klikom na povezavo "Registracija".

  Prijavljeni uporabnik lahko preko gumba "Uredi pomene" dostopa do pogleda za urejanje in dodajanje pomenov (Slika 4.2). Na voljo sta mu seznam povedi ter seznam pomenov. Privzeti seznam pomenov je narejen po zgledu pomenov iz SSKJ. Uporabniku je na dnu seznama na voljo okno, v katerega lahko napiše poljubni pomen in ga doda seznamu. Pomenu bo dodano uporabniško ime avtorja. Vsaki povedi

pripada po en pomen. Uporabnik lahko klikne na poved in okoli te se bo pojavil moder okvir. Poved je izbrana in uporabnik lahko klikne na enega od pomenov, ki se bo ob kliku obarval modro. Povezava med povedjo in pomenom je prikazana z barvnim okvirjem okoli povedi ter z obarvanim pomenom. Primer lahko vidimo na sliki 4.2.

Ob kliku na gumb "Shrani" se novi pomeni ter nove povezave med povedmi in pomeni shranijo v strežnik. Spremembe lahko vidijo vsi uporabniki spletne aplikacije.



Slika 4.2: Dodajanje pomenov povedim z glagolom napisati.

4.2.2 Arhitektura spletne aplikacije

Spletno aplikacijo sestavljajo tri komponente: podatkovna baza, zaledni del in uporabniški vmesnik.

Za podatkovno bazo smo uporabili odprtokodni projekt MongoDB. MongoDB je NoSQL baza, namenjena shrambi dokumentov, po strukturi podobnih in skladnih s formatom JSON. Prednost NoSQL baz pred klasičnimi SQL bazami je shranjevanje manj strukturiranih in nestrukturiranih objektov.

Spletna aplikacija uporabniku omogoča dodajanje pomenov glagolov in dodajanje teh pomenov povedim iz vhodnega korpusa. Podatki so shranjeni v naslednjih tabelah:

v2_senses hrani pomene, prepisane iz SSKJ, in uporabniško dodane pomene. Posamezni vnos vsebuje iztočnico, opis pomena, avtorja pomena ter datum vnosa. Vnosu je dodan unikatni identifikator;

v2_sense_map hrani povezave med pomeni in povedmi iz vhodnega korpusa. Posamezni vnos vsebuje identifikator pomena, identifikator povedi, iztočnico, avtorja povezave ter datum vnosa.

Zaledni del je implementiran v jeziku Python 3. Za obdelavo besedil smo uporabili knjižnico NLTK [6]. Strežnik in API smo implementirali z uporabo ogrodja Flask. Uporabniški vmesnik smo implementirali v jeziku JavaScript. Interaktivnost in uporabniku prijazen uporabniški vmesnik smo implementirali z uporabo knjižnice Vue.js.

4.3 Razdvoumljanje pomenov besed

Za avtomatiziran postopek odkrivanja vezljivostnih vzorcev glagolov moramo najprej ločiti povedi glede na pomen glagola. V spodnjem primeru lahko vidimo, da ima beseda *igrati* v SSKJ navedenih 8 različnih pomenov. Od tega imajo trije pomeni vsak po dva dodatna podpomena.

```
Primer pomenov besede igrati v SSKJ:
1. poustvarjati, navadno z umetniškim hotenjem
2. povzročati glasbo z glasbilom
3. biti dejaven v določenem skupinskem športu, organiziranem
```

```
po določenih pravilih
4. biti dejaven v določeni
  a) družabni igri
  b) igri za denar
5. ukvarjati se, navadno poklicno
  a) nepreh. z gledališko dejavnostjo
  b) z določenim skupinskim športom
6. ekspr. pretvarjati se, hliniti
7. ekspr., z dajalnikom delati nehotene majhne gibe,
premike za izražanje, kazanje
  a) močnega razburjenja
  b) prijetnega vznemirjenja, veselosti
8. ekspr., s prislovnim določilom biti opazen, viden
```

Razdvoumljanje pomenov besed (ang. Word Sense Disambiguation, WSD) je proces dodeljevanja pomenov besedam glede na njihov kontekst. Pri ljudeh se ta proces dogaja podzavestno, medtem ko v računalništvu obstajajo metode strojnega učenja, ki se poskušajo približati natančnosti človeka. Obstaja več metod strojnega razdvoumljanja, ki jih delimo v tri kategorije:

Slovarsko podprte metode črpajo informacije iz slovarjev in tezavrov. Dobra lastnost teh metod je dobra zastopanost večine besed v velikih slovarjih.

Nadzorovano strojno učenje uporablja označeni korpus kot učno množico. Te metode dajejo relativno natančne rezultate, a so omejene na velikost označenega korpusa.

Nenadzorovano strojno učenje ne uporablja zunanjih virov. Povedi poskuša ločiti po njihovih pomenih z uporabo informacij iz samega besedila.

Metode nadzorovanega strojnega učenja veljajo za najbolj uspešne. Za delovanje potrebujejo obsežno učno množico označenih primerov, česar pa

trenutno nimamo na voljo. Aplikacijski del te naloge se med drugim ukvarja z ročnim označevanjem pomenov stavkov. Preizkusili bomo nekaj implementacij Leskovega algoritma, ki spada med slovarsko podprte metode, ter dve implementaciji algoritma k-voditeljev (ang. k-means), ki spada med metode nenadzorovanega strojnega učenja.

4.3.1 Leskov algoritem

Leskov algoritem [17] je klasičen algoritem za razdvoumljanje pomena besed. Zasnovan je bil leta 1986, do danes pa je bil deležen številnih revizij in posodobitev. Algoritem naj bi v osnovi služil kot računsko cenejša alternativa algoritmom, ki besedila razdvoumljajo s pomočjo slovnične strukture.

Leskov algoritem uporablja že obstoječe in računalniško berljive slovarske opise. Poved lahko vsebuje večpomenske besede. Vsak pomen določene besede ima v slovarju svoj opis ali gloso. Če med seboj primerjamo glose parov besed, opazimo podobnost med glosami, ki se nanašajo na podoben pomen. Bolj natančno, podobne glose vsebujejo večje število skupnih besed. S pomočjo opisov pomenov lahko najdemo tiste pomene besed, ki se najbolje ujemajo s pomeni ostalih besed v povedi.

```
Primer za ločevanje "ice cream cone" in "pine cone":
PINE
1. kinds of evergreen tree with needle-shaped leaves
2. waste away through sorrow or illness

CONE
1. solid body which narrows to a point
2. something of this shape whether solid or hollow
3. fruit of certain evergreen trees

Največji presek imata opisa PINE 1 in CONE 3.
```

Algoritem 4.1: Originalni Leskov algoritem (OL)

```
max_presek = -1
naj_pomen = None

for razdv_pomen in pomeni(razdv_beseda):
  for beseda in (poved \ razdv_beseda):
   for pomen in pomeni(beseda):
     if presek(razdv_pomen, pomen) > max_presek:
      max_presek = presek(razdv_pomen, pomen)
      naj_pomen = pomen
```

Algoritem 4.1 prikazuje potek originalnega Leskovega algoritma. Razdvoumljamo pomen besede, shranjene v spremenljivki *razdv_beseda*. Vsak pomen te besede primerjamo z vsemi pomeni ostalih besed v povedi in za vsako primerjavo izračunamo število besed v preseku obeh opisov pomenov. Shranimo pomen, ki je dal največji presek s pomenom besede v povedi [39].

Razvite so bile številne izboljšave Leskovega algoritma. Glavne spremenljivke pri novejših implementacijah so izbira glose ter konteksta. Gloso definiramo kot tekstovne podatke o pomenu besede. Preprosto gloso lahko dobimo iz slovarske definicije pomena, medtem ko so kompleksnejše glose sestavljene iz glos originalne besede ter glos njenih nadpomenk ali sopomenk. Če imamo na razpolago besedila, v katerih so bili pomeni ročno označeni, lahko glose zgradimo iz te učne množice.

Kontekst so besede, ki obkrožajo besedo, katere pomen razdvoumljamo. Kot kontekst je lahko izbranih nekaj besed levo in desno od izbrane besede, lahko pa uporabimo celotno poved, v kateri se tarčna beseda nahaja.

V nadaljevanju opišemo nekaj inačic Leskovega algoritma.

Preprosti Lesk (ang. Simple Lesk, SL) [35] je prirejen Leskov algoritem (Original Lesk ali OL), ki uporablja poenostavljen kontekst. Kot kontekst vzame besede okoli tarčne besede in ne njihovih slovarskih definicij. Preprosti Lesk je računsko manj zahteven od originalnega al-

goritma, saj vsak pomen besede, ki jo razdvoumljamo, primerja z istim kontekstom.

Raziskava je pokazala, da SL daje boljše rezultate od OL [35]. V raziskavi so primerjali oba algoritma, različne velikosti konteksta ter različne glose. SL se je v vseh poskusih izkazal za boljšega od OL. Primerjali so kontekste velikosti 2, 3, 8, 10 in 25 besed. Najprimernejša velikost konteksta se je izkazala za 2 besedi pred in za tarčno besedo. Pri izbiri konteksta so upoštevali besede, za katere so obstajale slovarske glose.

Prirejeni Lesk (ang. Adapted Lesk, AL) [5] [9] za vir pomenov besed uporablja slovar WordNet [23]. Algoritem izbere kontekst okoli tarčne besede, nato za vsako besedo v kontekstu v slovarju poišče gloso. Glosa je sestavljena iz definicije osnovnega pomena ter iz definicij pomenov nadpomenk. Najboljše rezultate je dal algoritem, ki je iskal pomene do druge nadpomenke v drevesu. Kontekst, ki je dal najboljše rezultate, je vseboval 2 besedi levo in desno od tarčne besede. Ti dve besedi sta morali biti vsebovani v WordNetu, sicer jih je algoritem nadomestil.

Potek algoritma je podoben kot pri OL. Vsaka tarčna beseda ter beseda v kontekstu dobi enega ali več pomenov iz WordNeta. Naj bo $|b_\mathrm{i}|$ število pomenov besede $\mathrm{b_i}$. Algoritem primerja vse možne kombinacije pomenov besed in izbere tisto z najvišjo oceno. Če je število besed N, je število vseh kombinacij $\prod_{i=0}^{N} |b_\mathrm{i}|$. Eden od problemov, na katerega so raziskovalci naleteli, je veliko število pomensko precej podobnih pomenov v slovarju, ki jih je težko ločevati med sabo.

Lesk z dodatnimi izboljšavami : v raziskavi, opisani v članku [30], so preizkusili več načinov izgradnje in primerjave glos. Algoritem podobno kot AL črpa opise pomenov iz leksikona WordNet. Tudi v tej raziskavi se je izkazalo, da najboljše rezultate dajejo združene glose same besede z dvema nadpomenkama. Za primerjavo glos so izdelali vektorje tf-idf in izračunali kosinusno podobnost.

Tf-idf je utežni vektor, ki upošteva pogostost besede in pogostost pojavitve te besede v različnih dokumentih. V našem primeru dokumente predstavljajo glose. Tf-idf je izračunan za vsako besedo vseh glos, ki jih primerjamo med sabo po formuli (4.3):

$$tf(b) = |b|/B, \tag{4.1}$$

kjer sta:

$|b|$: število pojavitev besede b v dokumentu,
B : število vseh besed v dokumentu;

$$idf(b) = log(D/D_b), \tag{4.2}$$

kjer sta:

D : število vseh dokumentov,
D_b : število dokumentov, ki vsebujejo besedo b;

$$tfidf(b) = tf(b) * idf(b). \tag{4.3}$$

Intuitivno da tf-idf večjo težo manj pogostim besedam v dokumentih ali glosah.

Raziskava je pokazala, da najboljše rezultate daje Leskov algoritem, ki za kontekst uporablja eno poved, pri kateri uporabi osnovne besede brez njihovih glos iz WordNeta. Glose za tarčno besedo izdela iz definicije te besede ter iz definicij prvih dveh nadpomenk.

V Pythonu smo implementirali štiri različice Leskovega algoritma in jih preizkusili na korpusu. Nad besedili smo v vseh primerih izvedli tokenizacijo in korenjenje (*stemming*).

Za korenjenje angleških besedil je bil uporabljen paket **nltk.stem.snowball**. Za korenjenje slovenskih besedil smo uporabili paket **polyglot**, ki omogoča razčlenjevanje slovenskih besed na zloge. Besedam smo odstranili zadnji zlog in tako dobili približke korenov besed.

lesk_nltk je implementacija Leskovega algoritma, ki jo najdemo v knjižnici NLTK. Algoritem poišče glose pomenov glagola v SSKJ, kontekst pa zgradi iz besed v povedi. Algoritem primerja vse glose s kontekstom. Primerjava je presek besed dveh množic.

lesk_sl je implementacija SL. Algoritem črpa glose iz SSKJ. Kontekst zgradi iz štirih besed: dve besedi sta pred tarčno besedo ter dve besedi za tarčno besedo. Vsako gloso primerja s kontekstom ter izbere tisto z največjim prekrivanjem. Prekrivanje je definirano kot presek dveh množic besed.

lesk_al je implementacija AL. Algoritem v slovarju SloWNet poišče glose vseh pomenov tarčnega glagola. Nato poišče še glose vseh besed iz konteksta. Vsaka glosa je sestavljena iz opisov besede in dveh nadpomenk. Kontekst vsebuje dve besedi pred in za besedo, katere pomen razdvoumljamo. Algoritem nato primerja vse kombinacije glos vseh besed ter izbere kombinacijo z največjim prekrivanjem. Prekrivanje je definirano kot kosinusna podobnost dveh *tf-idf* vektorjev.

lesk_ram [30] je implementacija, podobna AL. Razlikuje se v izgradnji konteksta, ki je sestavljen iz posameznih besed povedi. Ta kontekst se s pomočjo tf-idf primerja z vsemi glosami tarčnega glagola. Glose so sestavljene iz opisov besede in dveh nadpomenk.

Natančnost algoritmov smo ocenili s pomočjo ročno označene podmnožice. Rezultate najdemo v razdelku 5.

4.3.2 Algoritem k-voditeljev

Algoritem k-voditeljev se uporablja za razporejanje N točk v večimenzionalnem prostoru v K gruč. Središče vsake gruče je predstavljeno z vektorjem m^k [19].

```
Potek algoritma k-voditeljev:
1. V prostoru izberemo K naključih središč gruč.
2. Vsako točko v prostoru dodamo gruči z najbližjim središčem.
3. Vsaki gruči ponovno izračunamo središče,
   ki predstavlja povprečje točk gruče.
4. Koraka 2 in 3 ponavljamo, dokler središča gruč ne konvergirajo.
```

Članek [4] opisuje gručenje arabskih dokumentov po podobnih tematikah. Za gručenje uporabi algoritma k-voditeljev ter bisekcijski k-voditeljev (od tu naprej BK). Algoritma k-voditeljev in BK se razlikujeta po načinu iskanja gruč. Algoritem k-voditeljev že v prvi iteraciji poišče s parametrom določeno število gruč. BK gruče išče postopoma, tako da v vsaki iteraciji razpolovi največjo gručo. Po zgledu članka smo pripravili dva algoritma, ki na vhodu prejmeta seznam povedi z glagolom, katerega pomen razdvoumljamo. Algoritma izračunata število pomenov glagola ter vhodnim povedim pripišeta pomenske labele.

Ker posamezna poved vsebuje malo informacij, smo jo obogatili z uporabo semantičnega leksikona SloWNet. Vsaka beseda v povedi je zamenjana z verigo njenih nadpomenk, ki jih dobimo z algoritmom 4.2. Pri izdelavi verige nadpomenk smo zaradi večjega števila le-teh uporabili angleške leme. Vsako poved torej pretvorimo v množico angleških lem.

Algoritem 4.2: Veriga nadpomenk

```
def veriga_nadpomenk(lema):
  rezultat = [lema]
  if len(SloWNet[lema].nadpomenke) == 0:
    return rezultat
  for nadpomenka in SloWNet[lema].nadpomenke:
    rezultat.extend(veriga_nadpomenk(nadpomenka))
  return rezultat
```

Množice lem, ki predstavljajo povedi, pretvorimo v vektorje tf-idf (4.3) ter jih pošljemo na vhod prirejenemu algoritmu k-voditeljev. Vektor tf-idf predstavlja utežene frekvence pojavitev posameznih besed v opazovanih dokumentih. Dolžina vektorja je skupno število unikatnih besed v dokumentih, v katerih posamezna vrstica predstavlja posamezno besedo.

Za osnovni algoritem k-voditeljev smo uporabili algoritem, implementiran v Pythonovi knjižnici NLTK. Za ocenjevanje razdalje med vektorji smo uporabili kosinusno razdaljo (4.5). Kosinusno razdaljo izračunamo s pomočjo kosinusne podobnosti, ki nam pove, kako podobna sta si dva vektorja.

$$sim_{cos}(a, b) = \frac{a^T b}{\|a\| \, \|b\|}, \tag{4.4}$$

$$dist_{cos}(a, b) = 1 - sim_{cos}(a, b) \tag{4.5}$$

kjer sta:

a : tf-idf vektor prvega dokumenta,
b : tf-idf vektor drugega dokumenta.

BK smo implementirali po zgledu članka [34]. Za razliko od osnovnega algoritma k-voditeljev, ki začne s k centroidi, bisekcijska različica začne z enim samim centroidom, ki ga razpolavlja, dokler ne doseže želenega števila gruč.

```
Potek algoritma bisekcijskih k-voditeljev:
1. Izberemo največjo gručo.
2. Razdelimo jo na 2 gruči z uporabo osnovnega algoritma k-voditeljev.
3. Korak 2 večkrat ponovimo in vzamemo rezultat z najmanjšimi
   povprečnimi razdaljami med elementi gruč in njihovimi centri.
4. Korake 1,2 in 3 ponavljamo, dokler ne dobimo
   želenega števila gruč.
```

Zgoraj opisane implementacije k-voditeljev zahtevajo vnaprej določeno število gruč (parameter k). Preizkusili smo avtomatično določanje števila gruč z uporabo silhuetne ocene.

Silhuetna ocena nam pove, kako podobna je točka svoji gruči v primerjavi z ostalimi gručami:

$$s(i) = \frac{b(i) - a(i)}{max(b(i), a(i))}, \tag{4.6}$$

kjer so:

i : točka ali vektor v prostoru,
$a(i)$: povprečna razdalja med i in centroidom gruče, ki ji pripada,
$b(i)$: povprečna razdalja med i in drugim najbližjim centroidom.

Oba algoritma (k-voditeljev ter BK) smo pognali z vrednostmi parametra k od 1 do 10. Za vsak k smo izračunali povprečno silhuetno oceno vseh točk. Na koncu smo izbrali rezultat z najboljšo povprečno silhuetno oceno.

Algoritma smo pognali nad vsemi glagoli v korpusu ter ju evalvirali s pomočjo ročno označene podmnožice. Rezultate najdemo v razdelku 5.

4.3.3 Druge metode razdvoumljanja

Poleg Leskovega algoritma in k-voditeljev obstajajo druge metode nenadzorovanega razdvoumljanja, ki bi bile primerne za naše podatke [8]:

Pedersenov pristop [26] je algoritem, ki za razdvoumljanje ne potrebuje zunanjih virov. Algoritem na začetku za vsako tarčno besedo zgradi svoj kontekst. Kontekst je predstavljen kot vektor značilk. Vsebuje oblikoslovne podatke o tarčni besedi, besedne vrste besed iz okolice ter medsebojne pojavitve pogostih besed v korpusu z besedami v povedi, ki jo razdvoumljamo. Algoritem zgradi matriko različnosti med posameznimi konteksti, na podlagi katere gruča podobne kontekste.

HyperLex je algoritem, primeren za iskanje odstavkov s podobnim pomenom, kot ga ima tarčna beseda. Iz besed, ki obdajajo tarčno besedo, zgradi graf, v katerem posamezna vozlišča predstavljajo besede. Povezava med dvema vozliščema je utežena tako, da nižjo utež dobijo pari besed, ki v korpusu pogosto nastopajo skupaj. Algoritem poišče gosteje povezane skupine vozlišč. Znotraj vsake gosteje povezane skupine poišče vozlišče z najvišjo stopnjo, ki ga imenujemo zvezdišče. Gosto povezane skupine z zvezdišči predstavljajo različne pomene. Algoritem doda tarčno besedo ter jo poveže z vsemi zvezdišči. Novim povezavam priredi utež 0, nato poišče minimalno vpeto drevo v grafu. Lastnost nastalega vpetega drevesa je, da ima vsako vozlišče natanko eno pot do vozlišča tarčne besede. Vozlišče tarčne besede je povezano s poddrevesi, ki predstavljajo njene možne pomene. Algoritem vsakemu poddrevesu izračuna oceno, ki odraža gostoto poddrevesa. Poddrevo z najvišjo oceno predstavlja pomen tarčne besede [36].

PageRank je algoritem, ki poišče najpomembnejša vozlišča v grafu. Algoritem lahko uporabimo za razdvoumljanje na grafu pomenskih razmerij [20]. Algoritem na vhodu prejme besedilo, ki ga razdvoumljamo, ter leksikon WordNet [23]. Za vsako besedo v vhodnem besedilu se poiščejo vnosi v leksikonu WordNet. Vnosi se dodajo v graf. Vozlišča so skupine sopomenk in predstavljajo pomene. Povezave so semantične povezave, ki jih razberemo iz leksikona WordNet. Vozlišča, katerih sopomenke vsebujejo isto lemo, ostanejo med seboj nepovezana. Zgrajeni graf po-

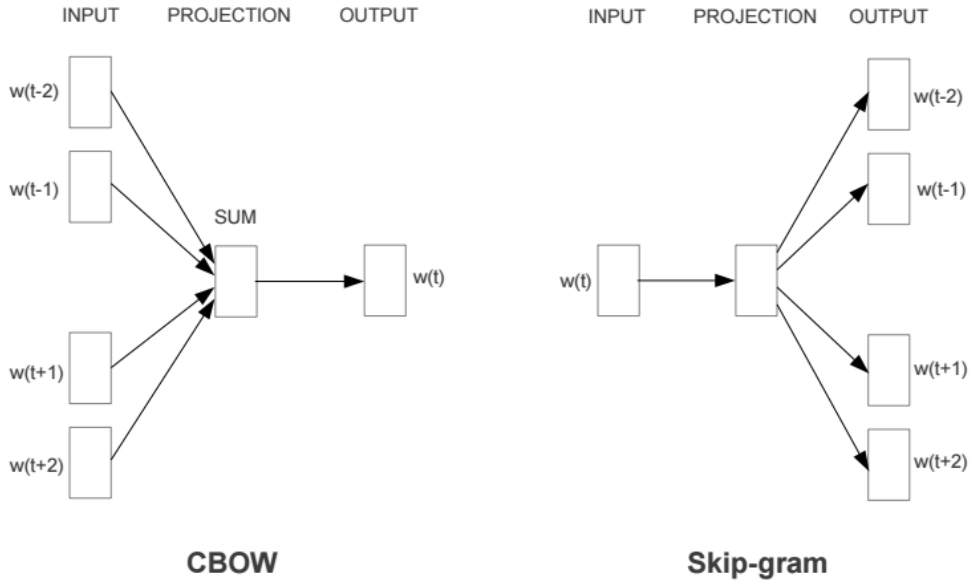
damo na vhod algoritmu PageRank [25]. PageRank vozlišču določi pomembnost glede na število vhodnih povezav ter glede na pomembnost vozlišč, iz katerih te povezave izhajajo. Pomen tarčne besede razdvoumljamo tako, da poiščemo vsa vozlišča, ki vsebujejo tarčno besedo med sopomenkami. Pomen predstavlja izbrano vozlišče z najvišjo PageRank oceno.

Zgoraj omenjeni algoritmi spadajo med starejše pristope k razdvoumljanju. V zadnjih letih dajejo najboljše rezultate algoritmi, ki uporabljajo nevronske mreže. Opisali bomo algoritme, ki bi jih lahko uporabili na naših podatkih [28].

Z uporabo nevronskih mrež lahko slovnične in semantične lastnosti jezika predstavimo v nižjedimenzionalnem prostoru, primernejšem za računalniško obdelavo. Postopek se imenuje vložitev (ang. embedding). Vložitev reši problem dimenzionalnosti, saj posamezno besedo pretvori v vektor sprejemljive velikosti (300 do 400). Izkaže se, da medsebojna geometrična razdalja vektorjev dobro ponazarja semantične lastnosti izvornih besed. Najboljšo vložitev dajejo pristopi z uporabo preproste nevronske mreže, naučene z veliko količino podatkov (nekaj bilijonov besed). Članek [21] opisuje dva modela za vložitev: **CBOW** in **Skip-gram**. CBOW na vhod prejme kontekst tarčne besede (na primer 5 besed pred in 5 besed za tarčno besedo). Na izhod poda verjetnostno porazdelitev za tarčno besedo. Skip-gram deluje obratno. Na vhodu prejme tarčno besedo in na izhodu vrne kontekst tarčne besede. Oba modela imata projekcijsko plast, ki se posodablja pri učenju. Projekcijska plast je končni rezultat učenja in predstavlja vložitev besed. Oba modela sta implementirana in prosto dostopna pod imenom *word2vec*.

Zgoraj opisana vložitev besed ne upošteva različnih pomenov posamezne besede. Vložitev pomenov posameznih besed je težavna, saj je težko ročno označiti učni korpus primerne velikosti za učenje nevronskih mrež. Medtem ko obstaja sprejemljiva rešitev za vložitev posameznih besed, raziskovalci poskušajo v isti vektorski prostor vložiti tudi pomene.

Članek [14] opisuje postopek nenadzorovanega razdvoumljanja z uporabo



Slika 4.3: Grafični prikaz arhitektur CBOW in Skip-gram.

vhodnega korpusa in leksikona pomenskih razmerij med besedami. Raziskovalci predlagajo algoritem, ki v isti vektorski prostor vloži tako leme iz vhodnega korpusa kot pomene besed iz leksikona. Pomen vložene tarčne besede se določa z geometrično najbližjim vloženim pomenom v vektorskem prostoru.

Najsodobjenši model je trenutno **ELMo** [27]. Kratica ELMo pomeni 'vložitve iz jezikovnih modelov' (ang. Embeddings from Language Models). Vložitev besede v modelu ELMo predstavlja funkcijo celotne povedi, iz katere beseda izhaja. Tako posamezna vložitev vsebuje informacije o kontekstu. Vložitve so izračunane na podlagi večplastnega dvosmernega jezikovnega modela (ang. bidirectional Language Model ali biML). Posamezna vložitev besede v ELMo je utežena vsota skritih plasti modela biML. Uteži modela ELMo so naučene za specifično nalogo, kot na primer razdvoumljanje pomenov besed.

Poglavje 5

Evalvacija razdvoumljanja

Pri izdelavi algoritmov za razdvoumljanje smo uporabljali različne zunanje vire, kot sta SSKJ in SloWNet [10], v primeru algoritmov k-voditeljev pa smo vse informacije dobili iz podatkov samih. Vsak algoritem je proizvedel različno množico razredov za posamezni glagol. Preprosti Lesk je na primer za razrede uporabil pomene glagola v SSKJ, prirejeni Lesk pa je za razrede uporabil pomene iz SloWNeta. Pomeni v SSKJ in pomeni v SloWNetu za posamezno besedo niso enotni.

Algoritmi nam vrnejo razrede, ki jih je težko preslikati na razrede ročno označene množice. Zanima nas, kako dobro so se združevale povedi s podobnimi pomeni, ne pa, katerim razredom so bile dodeljene. Za oceni smo uporabili *čistost gručenja* (ang. clustering purity) ter *Randov indeks*.

Čistost gručenja [33] je ocena, ki nam pove delež pravilno dodeljenih razredov. Podatkom v gručah dodelimo razrede iz ročno označene množice. Razred, ki je v določeni gruči prevladujoč, smatramo kot pravilen razred te gruče. Ocena je število pravilno dodeljenih točk v vseh gručah deljeno s številom vseh točk. Čistost gručenja poda vrednost med vključno 0 za slabo gručenje in 1 za dobro gručenje. Definiramo:

$$p(\Omega, \mathbb{C}) = \frac{1}{N} \sum_k \max_j |\omega_k \cap c_j|, \tag{5.1}$$

kjer sta:

$$\Omega = \{\omega_1, \omega_2, ..., \omega_K\} : \text{množica gruč},$$
$$\mathbb{C} = \{c_1, c_2, ..., c_J\} : \text{množica razredov}.$$

Randov indeks [31] za ocenjevanje gruč uporablja pare točk. Predpostavlja, da so vse točke enako pomembne. Točki sta pravilno razvrščeni v dveh primerih:

- algoritem ju razporedi v isti razred ter v ročno označeni množici pripadata istemu razredu,
- algoritem ju razporedi v različna razreda ter v ročno označeni množici pripadata različnima razredoma.

Algoritem vrne vrednost na intervalu $[0, 1]$, pri čemer 1 predstavlja najboljše gručenje:

$$c(Y, Y') = \frac{\sum_{i<j}^{N} \gamma_{i,j}}{\binom{N}{2}}, \tag{5.2}$$

kjer so:

$$\{X_0, X_1, X_N\} : \text{primeri, točke}$$
$$Y : \text{algoritemsko kreirane gruče}$$
$$Y' : \text{ročno označene gruče}$$

$$\gamma_{i,j} = \begin{cases} 1, & \text{obstajata } k \text{ in } k', \text{ kjer sta primera } X_i \text{ in } X_j \\ & \text{skupaj v obeh gručah } Y_k \text{ in } Y'_{k'} \\ 1, & \text{obstajata } k \text{ in } k', \text{ kjer je primer } X_i \text{ v obeh gručah} \\ & Y_k \text{ in } Y'_{k'} \text{ ter } X_j \text{ v nobeni od teh dveh gruč} \\ 0 & \textit{sicer} \end{cases}$$

Prilagojeni Randov indeks (ang. Adjusted Rand Index, ARI) [37] je različica Randovega indeksa, ki bolje ločuje dobra gručenja od naključnih gručenj. Uporabili smo **adjusted_rand_score** iz Pythonove knjižnice **sklearn.metrics**. Algoritem poda oceno na intervalu $[-1, 1]$. Vrednost 0 predstavlja naključni gručenji, vrednost 1 pa identični gručenji.

Za evalvacijo smo ročno označili pomene 60 glagolov. Izbirali smo glagole, ki so se v korpusu nahajali v vsaj štirih povedih. Izbirali smo tudi glagole, katerih povedi smo lahko jasno razvrstili v več različnih pomenskih razredov. Evaluirali smo primere, za katere so vsi algoritmi (štirje Leskovi algoritmi in dva algoritma k-voditeljev) dali rezultate. Algoritmi, ki so uporabljali leksikon SloWNet in SSKJ, so primer preskočili, če ključne besede ni bilo v leksikonu ali slovarju. Ocenili smo delovanje vseh šestih algoritmov na 45 primerih. Rezultate najdemo v tabeli 5.1.

Algoritem	Randov indeks	ARI	Čistost gručenja
lesk_nltk	0.579	0.188	0.834
lesk_sl	0.608	0.248	0.911
lesk_al	0.488	0.069	0.676
lesk_ram	0.576	0.142	0.841
k-voditeljev	0.457	-0.005	0.650
bisekcijski k-voditeljev	0.475	-0.010	0.668

Tabela 5.1: Ocene algoritmov za razdvoumljanje

Izkaže se, da je najboljše rezultate ponudil preprosti Leskov algoritem. Preprosti Lesk razdvoumlja pomen glagola z uporabo štirih besed, ki so v stavku najbližje glagolu. Iz rezultata lahko sklepamo, da besede v ožji okolici glagola najbolj pripomorejo k razdvoumljanju pomena.

Algoritmi so dali prenizke rezultate za praktično uporabo, zato smo orodju za pregledovanje vezljivostnih vzorcev dodali funkcionalnost za ročno urejanje pomenov povedi. Orodje je dostopno preko spleta in pri urejanju lahko sodeluje več uporabnikov. Eden od zaželenih rezultatov uporabe orodja je

izdelava učne množice z označenimi pomeni povedi. S tovrstno učno množico bo mogoče razvijati algoritme nadzorovanega strojnega učenja, ki pri razdvoumljanju pomenov povedi dajejo boljše rezultate od algoritmov nenadzorovanega strojnega učenja [8].

Poglavje 6

Zaključek

Izdelali smo aplikacijo za pregledovanje vezljivostnih vzorcev. V delu smo se osredotočali na korpus ssj500k 2.1 [15], aplikacijo pa smo prilagodili za procesiranje poljubnega korpusa, zgrajenega po smernicah TEI [7]. Pripravlja se korpus Gigafida 2.0, avtomatsko označen z udeleženskimi vlogami, iz katerega bo aplikacija znala razbrati vezljivostne vzorce.

Na korpusu smo preizkusili nabor algoritmov za razdvoumljanje pomenov povedi, ki temeljijo na slovarsko podprti metodi. Natančnost algoritmov je bila prenizka za praktično uporabo, zato smo uporabnikom omogočili ročno dodeljevanje pomenov povedi. Ta lastnost aplikacije nam omogoča postopno grajenje ročno označene učne množice za nadaljnje strojno učenje.

Ključni dejavnik pri izdelavi algoritmov za razdvoumljanje je bogata učna množica. Pri delu smo črpali podatke iz leksikona SloWNet [10] in SSKJ [3]. SloWNet je preveden v slovenščino, a ne v celoti. Število slovenskih primerov je premajhno za strojno učenje. Slovar SSKJ nam ponuja nedvoumne opise in primere za posamezne iztočnice, težava pa je v majhni količini primerov za posamezno iztočnico.

Za hitrejši razvoj algoritmov za razdvoumljanje pomenov besed v slovenskem jeziku bi bil ključnega pomena prostodostopni korpus, po obsegu primerljiv s SSKJ, obogaten z dodatnimi primeri povedi za posamezne pomene iztočnice.

Literatura

[1] CROVALLEX 2.0008. `http://theta.ffzg.hr/crovallex/data/html/generated/alphabet/index.html`. Dostop: 2018-07-01.

[2] Projekt JOS. `http://nl.ijs.si/jos/`. Dostop: 2018-07-29.

[3] Slovar slovenskega knjižnega jezika, druga, dopolnjena in deloma prenovljena izdaja. `https://fran.si/`. Dostop: 2018-07-21.

[4] Diab Abuaiadah. Using bisect k-means clustering technique in the analysis of Arabic documents. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 15(3):17, 2016.

[5] Satanjeev Banerjee and Ted Pedersen. An adapted Lesk algorithm for word sense disambiguation using WordNet. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 136–145. Springer, 2002.

[6] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit.* "O'Reilly Media, Inc.", 2009.

[7] Lou Burnard and Syd Bauman, editors. *TEI P5: Guidelines for Electronic Text Encoding and Interchange*, chapter A Gentle Introduction to XML. Text Encoding Initiative Consortium, 2007.

[8] D Singh Chaplot and Dr Pushpak Bhattacharyya. Literature survey on unsupervised word sense disambiguation. *IIT Bombay, May*, 7, 2014.

[9] Jonas Ekedahl and Koraljka Golub. Word sense disambiguation using WordNet and the Lesk algorithm. Technical report, Lunds Universitet, 2004.

[10] Darja Fišer. Semantic lexicon of slovene sloWNet 3.1. `http://hdl.handle.net/11356/1026`, 2015. Slovenian language resource repository CLARIN.SI.

[11] Polona Gantar, Kristina Štrkalj Despot, Simon Krek, and Nikola Ljubešić. *Towards Semantic Role Labeling in Slovene and Croatian.* PhD thesis, Department of Knowledge Technologies, Jožef Stefan Institute.

[12] Jan Hajic, Jarmila Panevová, Zdenka Urešová, Alevtina Bémová, Veronika Kolárová, and Petr Pajas. Pdt-vallex: Creating a large-coverage valency lexicon for treebank annotation. In *Proceedings of the second workshop on treebanks and linguistic theories*, volume 9, pages 57–68, 2003.

[13] Jan Hajič, Eduard Bejček, Alevtina Bémová, Eva Buráňová, Eva Hajičová, Jiří Havelka, Petr Homola, Jiří Kárník, Václava Kettnerová, Natalia Klyueva, Veronika Kolářová, Lucie Kučová, Markéta Lopatková, Marie Mikulová, Jiří Mírovský, Anna Nedoluzhko, Petr Pajas, Jarmila Panevová, Lucie Poláková, Magdaléna Rysová, Petr Sgall, Johanka Spoustová, Pavel Straňák, Pavlína Synková, Magda Ševčíková, Jan Štěpánek, Zdeňka Urešová, Barbora Vidová Hladká, Daniel Zeman, Šárka Zikánová, and Zdeněk Žabokrtský. Prague dependency treebank 3.5, 2018. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

[14] Richard Johansson and Luis Nieto Pina. Combining relational and distributional knowledge for word sense disambiguation. In *Proceedings of*

the 20th Nordic Conference of Computational Linguistics (NODALIDA 2015), pages 69–78, 2015.

[15] Simon Krek, Kaja Dobrovoljc, Tomaž Erjavec, Sara Može, Nina Ledinek, Nanika Holz, Katja Zupan, Polona Gantar, Taja Kuzman, Jaka Čibej, Špela Arhar Holdt, Teja Kavčič, Iza Škrjanec, Dafne Marko, Lucija Jezeršek, and Anja Zajc. Training corpus ssj500k 2.1. `http://hdl.handle.net/11356/1181`, 2018. Slovenian language resource repository CLARIN.SI.

[16] Simon Krek, Polona Gantar, Kaja Dobrovoljc, and Iza Škrjanec. Označevanje udeleženskih vlog v učnem korpusu za slovenščino. In *Proceedings of the Conference on Language Technologies and Digital Humanities*, pages 106–110. Faculty of Arts, University of Ljubljana, 2016.

[17] Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26. ACM, 1986.

[18] Markéta Lopatková. Valency in the prague dependency treebank: Building the valency lexicon. *Prague Bull. Math. Linguistics*, 79:37–60, 2003.

[19] David JC MacKay and David JC Mac Kay. *An Example Inference Task: Clustering*. Cambridge university press, 2003.

[20] Rada Mihalcea, Paul Tarau, and Elizabeth Figa. PageRank on semantic networks, with application to word sense disambiguation. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1126. Association for Computational Linguistics, 2004.

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

[22] Marie Mikulová, Allevtina Bémová, Jan Hajič, Eva Hajičová, Jiří Havelka, Veronika Kolářová, Lucie Kučová, Markéta Lopatková, Petr Pajas, Jarmila Panevová, et al. Annotation on the tectogrammatical layer in the prague dependency treebank. *Annotation manual. Tech. rep., UFAL MFF UK, Prague, Czech Republic. URL http://ufal. mff. cuni. cz/pdt2. 0/doc/manuals/en/t-layer/pdf/t-man-en. pdf. English translation*, 2006.

[23] George A. Miller. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

[24] Milan Moguš, Maja Bratanić, and Marko Tadić. *Hrvatski čestotni rječnik*. Zavod za lingvistiku Filozofskog fakulteta, 1999.

[25] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[26] Ted Pedersen and Rebecca Bruce. Distinguishing word senses in untagged text. *arXiv preprint cmp-lg/9706008*, 1997.

[27] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[28] Alexander Popov. Neural network models for word sense disambiguation: An overview. *Cybernetics and Information Technologies*, 18(1):139–151, 2018.

[29] Nives Mikelic Preradovic, Damir Boras, and Sanja Kisicek. CROVALLEX: Croatian verb valence lexicon. In *Proceedings of the ITI 2009, 31st International Conference on Information Technology Interfaces*, pages 533–538. IEEE, 2009.

[30] G. Ramakrishnan, B. Prithviraj, and P. Bhattacharya. A gloss-centered algorithm for disambiguation. In *Proceedings of SENSEVAL-3, the*

Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text, 2004.

[31] William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971.

[32] Karin Kipper Schuler. Verbnet: A broad-coverage, comprehensive verb lexicon. 2005.

[33] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. *Introduction to information retrieval*, volume 39. Cambridge University Press, 2008.

[34] Michael Steinbach, George Karypis, and Vipin Kumar. A comparison of document clustering techniques. In *KDD workshop on text mining*, volume 400, pages 525–526. Boston, 2000.

[35] Florentina Vasilescu, Philippe Langlais, and Guy Lapalme. Evaluating variants of the Lesk approach for disambiguating words. In *Proceedings of Language Resources and Evaluation Conference*, 2004.

[36] Jean Véronis. Hyperlex: lexical cartography for information retrieval. *Computer Speech & Language*, 18(3):223–252, 2004.

[37] Ka Yee Yeung and Walter L Ruzzo. Details of the adjusted rand index and clustering algorithms, supplement to the paper an empirical study on principal component analysis for clustering gene expression data. *Bioinformatics*, 17(9):763–774, 2001.

[38] Andreja Žele. *Vezljivostni slovar slovenskih glagolov*. Založba ZRC, 2008.

[39] A. Zouaghi, L. Merhbene, and M. Zrigui. Word sense disambiguation for arabic language using the variants of the Lesk algorithm. *WORLDCOMP*, 11:561–567, 2011.