Jezikovne tehnologije in zapis korpusa

Tomaž Erjavec, Peter Holozan in Nikola Ljubešić

Abstract
This paper provides an overview of the levels of primarily automatic linguistic annotation that should be part of the annotation of corpora to be used inter alia as the basis for lexicographic analyses of contemporary Slovenian language. An overview of existing research in this field is outlined, with the focus then turning to a specific set of open source and mainly language independent tools and their models for Slovenian, with suggestions provided for their improvement. A short description of the proposed corpus encoding is also presented.

Keywords: linguistic annotation, corpora, annotation format

Ključne besede: jezikoslovno označevanje, korpusi, zapis oznak

1 UVOD

Prispevek obravnava jezikoslovne oznake, zapisane v korpusih, ki bi lahko služile kot osnova slovaropisnim dejavnostim, in format zapisa teh korpusnih oznak. Jezikovnotehnološke oznake se v korpuse zapišejo avtomatsko s pomočjo orodij, napisanih ali naučenih za označevanje slovenskega jezika, vendar pa ni izključeno, da se naknadno ne popravljajo, bodisi ročno ali pa avtomatsko, s ciljnimi ekspertnimi programi.

Prispevek ne pokriva vseh jezikovnotehnoloških orodij, ki jih je potrebno ali koristno imeti pri slovaropisnem delu s korpusi, temveč samo tista, za katera se predvideva, da bodo njihovi izhodi, torej oznake, dejansko našli mesto v zapisu korpusa. Zato tu ne obravnavamo orodij, namenjenih luščenju informacij iz korpusov, pač pa se osredotočamo na ravni označevanja, ki lahko služijo kot vir znanja o jeziku za luščilne programe, od konkordančnikov do programov za luščenje sopomenk. Ob tem se tudi večinoma omejimo na programe, ki so že bili razviti za slovenski jezik. Obravnavali bomo naslednje ravni označevanja, ki se večinoma tudi izvajajo v podanem zaporedju:

1. **tokenizacija**, ki razdeli besedilo na posamezne pojavnice, bodisi besede ali ločila; ta korak tipično identificira posebne razrede pojavnic, kot so cifre, okrajšave, naslove URL itd., vanj pa tipično sodi tudi **segmentacija** besedila na povedi, saj je ta neposredno odvisna od tokenizacije;
2. **normalizacija**, ki se uporablja za pretvorbo nestandardnih besednih oblik (kot jih najdemo npr. v uporabniško ustvarjenih besedilih na spletu) v standardne z namenom lažjega iskanja besed, kot tudi zato, da lahko nad normaliziranimi besedami uporabljamo standardna orodja za naslednje korake označevanja;
3. **oblikoskladenjsko označevanje**, ki besednim pojavnicam v besedilu pripiše sobesedilno odvisno enolično oblikoskladenjsko oznako, kot npr. »obči samostalnik moškega spola v imenovalniku dvojine«;
4. **lematizacija**, ki besedni obliki, tipično glede na njeno oblikoskladenjsko oznako, pripiše njeno osnovno obliko;
5. **skladenjsko razčlenjevanje**, ki vsako poved v besedilu skladenjsko razčleni; ta raven označevanja za slovaropisje sicer ni nujna, lahko pa se izkaže za koristno, še posebej pri proučevanju besednih zvez.

Obstajajo še druge ravni označevanja, ki so tudi lahko koristne, a niso nujne za slovaropisje, hkrati pa jih je težje umestiti v zaporedje zgoraj podanih korakov, saj jih lahko, odvisno od pristopa, določamo na podlagi neobdelanega besedila,

pri čemer jih umestimo neposredno po koraku 1 oz. 2, ali pa uporabimo več informacij, kar pomeni, da jih izvajamo po koraku 4 ali celo 5. Od teh orodij so bila nekatera sicer že razvita za slovenski jezik, čeprav samo prototipno, razvoj drugih pa je še povsem v zametkih. Ta označevanja, katerih rezultati se tudi lahko zapišejo v korpus, so:

6. **določanje imenskih entitet**, ki v besedilu identificira lastna imena in jih klasificira, npr. v osebna imena, zemljepisna imena, imena podjetij in ustanov itd.; ob tem nekateri sistemi identificirajo še številske in druge izraze ter jih klasificirajo, npr. v denarne enote, datume itd.;
7. **določanje terminov**, pri čemer se je treba zavedati, da je definicija termina razmeroma problematična, saj koncept termina ni vedno enoumno določen, temveč je pogosto odvisen od področja, ciljne publike ipd.;
8. **pripisovanje semantičnih informacij**, kjer besedam ali besednim zvezam pripišemo njihov pomen glede na neki semantičnoleksikalni vir, lahko pa jih tudi medsebojno povežemo s semantičnimi vlogami; čeprav bi takšno označevanje bilo izredno koristno za leksikografijo, zaradi kompleksnosti tega problema trenutni programi verjetno niso dovolj natančni, da bi dajali res uporabne rezultate.

Za orodja, ki opravljajo našteta označevanja, kot tudi za jezikovnotehnološka orodja na splošno tipično uporabljamo dva načina prilagajanja na določen jezik:

- Orodja uporabijo ročno napisana pravila, ki sicer zahtevajo veliko človeškega dela, lahko pa dajo (odvisno od ravni označevanja) zelo dobre rezultate. Takšna orodja se dostikrat uporabljajo za segmentacijo besedila, npr. v pojavnice ali termine, ali pa, tradicionalno, za morfološko analizo. Pri označevanjih, kot sta oblikoskladenjsko označevanje in skladnja, je teh pravil zelo veliko, pri čemer so dostikrat tudi medsebojno odvisna, kar zelo otežuje njihov razvoj in razhroščevanje v primerih, ko ne dajo želenih rezultatov.
- Orodja se naučijo modela določenega jezika na osnovi učnih podatkov, torej (ročno) označenih korpusov ali drugih jezikovnih virov. Metode strojnega učenja se hitro razvijajo, vendar pa potrebujemo za izdelavo kvalitetnih modelov čim večje označene jezikovne vire – njihova izdelava pa je tipično zamudna in draga.

Obe vrsti orodij tipično uporabljata tudi zaledne vire znanja o jeziku, predvsem leksikon (Dobrovoljc et al. 2015).

2 PREGLED ORODIJ ZA SLOVENSKI JEZIK

Za vse naštete ravni označevanja so bila za slovenski jezik že razvita (vsaj prototipna) orodja. V nadaljevanju razdelka podajamo pregled glavnih, ki jim je skupno to, da se vsaj do neke mere še vedno vzdržujejo in so tipično tudi prosto oz. odprto dostopna. Zato med našteta orodja ne vključimo sicer enega prvih sistemov za oblikoskladenjsko označevanje slovenskega jezika, ki je bil razvit na Inštitutu za slovenski jezik Frana Ramovša ZRC SAZU za potrebe označevanja njihovih korpusov (Jakopin in Bizjak Končar 1997).

2.1 Orodja podjetja Amebis

Orodja podjetja Amebis po eni strani niso odprto dostopna, po drugi pa predstavljajo pri nas najdlje razvijan integriran sistem označevalnih orodij in zalednih virov, ne samo prilagojenih, temveč napisanih posebej za obdelavo slovenskega jezika. Orodja so bila v osnovi razvita za slovnični pregledovalnik Besana (Holozan 2012) in strojni prevajalnik Presis (Romih in Holozan 2002), napisana so v programskem jeziku C++, delujejo v 32- in 64-bitni različici. Njihova struktura nekoliko odstopa od klasične: tudi tukaj je na prvem mestu tokenizacija, njena pomembna lastnost pa je, da kot enoto obravnava tudi posebne pojavnice, kot so spletni in elektronski naslovi, telefonske številke ter smeški. V drugem koraku sledi označevalnik, ki besedam s pomočjo slovarja pripiše vse možne kombinacije lem in oblikoskladenjskih oznak (slovar trenutno vsebuje več kot 7,6 milijona elementov). Tudi označevalnik prepoznava pojavnice, kot so spletni naslovi, kemijske spojine in smeški, hkrati pa išče še morebitne zatipkane besede in nekatere tipične nestandardne oblike, pri čemer je del nestandardnih besed že v slovarju s posebnimi oblikoskladenjskimi oznakami. Zadnji korak je analizator, ki za vsako besedo izbere najverjetnejši par leme in oblikoskladenjske oznake, hkrati pa v Amebisovem vmesnem jeziku (Holozan 2011) zapiše tudi skladenjsko razčlembo in pomene besed, ki so vzeti iz baze Ases (Arhar in Holozan 2009). Analizator lahko po potrebi vpliva tudi na tokenizacijo oz. segmentacijo besedila, npr. pri razreševanju primerov tipa »Videl sem ga. Micka ga je tudi videla.« (v primerjavi s »Prišla je še ga. Micka.«). Tu bi naivni tokenizator »ga.« lahko identificiral kot eno pojavnico (okrajšavo) in besedilo kot eno poved, Besana pa pravilno kot dve pojavnici in dve povedi. Amebisova orodja delujejo s pomočjo ročno napisanih pravil in podatkov v bazi Ases, pri čemer je najpomembnejši koncept glagolskih predlog (Holozan 2011), ki vsebujejo podatke o vezljivosti glagolov, vnesenih pa je tudi veliko lastnih imen, razporejenih v približno 30 kategorij (kar omogoča tudi določanje imenskih entitet). Narejen je bil tudi program, ki besedilo tokenizira,

lematizira in oblikoskladenjsko označi v skladu s specifikacijami, ki so bile uporabljene v projektu Sporazumevanje v slovenskem jeziku in jih implementira tudi označevalnik Obeliks (o tem več v nadaljevanju).

Z Amebisovimi orodji sta bila med drugim označena korpusa Fida in FidaPLUS. Za širšo uporabo Amebisovih orodij je največja prepreka to, da so lastniška in kot taka niso odprtokodna: za njihovo uporabo je, vsaj trenutno, potrebno skleniti dogovor z Amebis, d. o. o.

2.2 Označevalnik To(Tr)TaLe

Na Odseku za tehnologije znanja IJS je bilo v zaporedju več projektov razvito orodje ToTaLe (Erjavec et al. 2005), ki implementira cevovod treh modulov: tokenizatorja, ki besedilo tudi segmentira na povedi, oblikoskladenjskega označevalnika in lematizatorja. Za tokenizacijo ToTaLe uporablja modul mlToken, večjezični tokenizator, ki za prilagoditev na posamezni jezik uporablja jezikovno odvisne sezname, npr. okrajšav ali pravil za zapis števil. ToTaLe za oblikoskladenjsko označevanje uporablja program TnT (Brants 2000), zdaj že razmeroma star označevalnik, ki ga naučimo modela posameznega jezika nad ročno označenim korpusom, lahko pa uporablja tudi zaledni leksikon. Trenutni model je bil naučen na korpusu jos1M (Erjavec et al. 2010; Erjavec in Krek 2010), za zaledni leksikon pa uporablja kar besedišče korpusa FidaPLUS (Arhar in Gorjanc 2007). Za lematizacijo ToTaLe uporablja program CLOG (Erjavec in Džeroski 2004), ki na osnovi besedne oblike in njene oblikoskladenjske oznake besedni obliki določi njeno osnovno obliko. Tudi ta program se nauči modela lematizacije avtomatsko, na osnovi učne množice, ki je v tem primeru seznam trojčkov (besedna oblika, oblikoskladenjska oznaka, lema). Seznam, ki je programu služil kot učna množica, je bil zajet iz kombinacije besedišča korpusa jos100k, ročno pregledanih pojavnic v korpusu jos1M in izbranih besed iz korpusa FidaPLUS. ToTaLe je dostopen za uporabo na spletu, z njim pa je bila označena tudi večina korpusov, ki so dostopni za pregledovanje na konkordančniku noSketchEngine (Rychly 2007), instaliranem na nl.ijs.si (Erjavec 2013).

ToTaLe je napisan v programskem jeziku Perl, ravno tako tudi modula za tokenizacijo in lematizacijo. Perl sicer dandanes ni več zelo popularen programski jezik, vseeno pa obstajajo zanj implementacije za vse glavne operacijske sisteme. Zato pa oblikoskladenjski označevalnik TnT, ki ga uporablja ToTaLe, ni odprtokoden in je dostopen samo za nekomercialno rabo, in to samo v prevedeni obliki za operacijski sistem Linux, tako da ToTaLe v trenutni postavitvi ne more biti niti odprto dostopen niti ni uporaben npr. na operacijskem sistemu Windows.

V nadaljevanju raziskav je bilo razvito orodje ToTrTaLe, ki je enako kot ToTaLe, a z dvema novostma: po modulu za tokenizacijo smo vključili tudi (opcijske) module za transkripcijo, za razliko od orodja ToTaLe – ki pričakuje kot vhod golo besedilo in izhod vrne v obliki tabelarične datoteke – pa ToTrTaLe na vhodu pričakuje datoteko XML, ki uporablja shemo TEI, ravno tako pa vrne izhod v obliki TEI XML. Transkripcijski modul je mišljen za uporabo pri starejših (slovenskih) besedilih z namenom posodobitve posameznih besed, s čimer bistveno olajša delo oblikoskladenjskemu označevalniku in lematizatorju, saj sta oba naučena za obdelavo sodobnih besedil. Za posodabljanje modul za transkripcijo trenutno uporablja orodje Vaam (Reffle 2011) z ročno napisanimi pravili za posodabljanje starejših slovenskih besed. Z orodjem ToTrTaLe je bil zaenkrat označen samo korpus starejše slovenščine IMP (Erjavec 2015).

Obe orodji sicer dajeta razmeroma dobre rezultate, skupno pa jima je to, da nista najbolje vzdrževani: bilo bi ju npr. dobro na novo naučiti modelov za slovenski jezik, saj so sedaj na voljo boljši viri, predvsem leksikon Sloleks (Arhar 2009; Dobrovoljc et al. 2013) in korpus ssj500k. Tudi sicer je okolje, v katerem sta narejeni, sedaj že zelo zastarelo, kar velja tudi za posamezne module programov. Vsaj TnT bi bilo nujno potrebno zamenjati s kakšnim sodobnejšim, predvsem pa odprtokodnim in od operacijskega sistema neodvisnim označevalnikom.

Kot omenjeno, je bila normalizacija v kontekstu posodabljanja starejših besed slovenskega jezika v sklopu orodja ToTrTaLe že implementirana. Vendar so bila pravila tam napisana ročno, izkaže pa se, da boljše rezultate dajejo avtomatsko naučeni modeli normalizacije, ki za osnovo uporabljajo statistično strojno prevajanje na osnovi znakov (Scherrer in Erjavec 2013). Osnovno orodje, ki se uporablja za takšno normalizacijo, je statistični strojni prevajalnik Moses (Koehn et al. 2007), ki ga izšolamo na parih izvorna (nestandardna) beseda : normalizirana beseda. Ta pristop je uporaben ne samo za posodabljanje starejših besed, temveč tudi za predobdelavo sodobnih, nestandardno zapisanih besedil, kakršne npr. najdemo v spletnem komuniciranju, zaradi česar je relevanten tudi za slovar sodobnega slovenskega jezika. Pristop s statističnim strojnim prevajanjem na osnovi znakov je že bil preizkušen za standardizacijo besed v slovenskih tvitih (Ljubešić et al. 2014), in to s spodbudnimi rezultati. Pri normalizaciji besed se seveda pojavi tudi vprašanje, katera besedila normalizirati; če namreč orodje uporabljamo tudi pri standardnih besedilih, je velika verjetnost, da bo »normaliziralo« tudi povsem standardne besede, s čimer bo naredilo več škode kot koristi. Tudi tu so bile že izvedene začetne raziskave, pri katerih se je sistem regresijskega strojnega učenja na osnovi manjše, ročno označene množice slovenskih tvitov in drugih uporabniško generiranih vsebin s spleta naučil pripisati stopnjo nestandardnosti novim besedilom (Ljubešić et al. 2015). Normalizacijo bi tako lahko uporabili samo pri besedilih, ki so avtomatsko označena kot nestandardna.

2.3 Označevalnik Obeliks in skladenjski razčlenjevalnik za slovenščino

V okviru projekta Sporazumevanje v slovenskem jeziku je bilo razvito orodje Obeliks (Grčar et al. 2012), ki vhodno besedilo tokenizira, segmentira na povedi, oblikoskladenjsko označi in lematizira. Za tokenizacijo Obeliks uporablja modul z ročno napisanimi pravili, za oblikoskladenjsko označevanje namensko razvit modul strojnega učenja po sistemu največje entropije, za lematizacijo pa sistem LemmaGen (Juršič et al. 2010), ki prav tako temelji na strojnem učenju. Posebnost oblikoskladenjskega označevalnika je ta, da za označevanje ne uporablja samo modela, avtomatsko naučenega iz učnega korpusa, pač pa tudi ročno napisana ekspertna pravila, ki filtrirajo hipoteze, ki jih je generiral model, poleg tega pa usklajuje rezultate lematizatorja in označevalnika, tako da ne prideta v kontradikcijo. Obeliks je bil naučen na ročno označenem korpusu ssj500k (Arhar 2009; Krek et al. 2013) in izmed javno dostopnih orodij dosega najboljše rezultate za slovenski jezik. Trenutno največji problem tega označevalnika je verjetno ta, da je implementiran v programskem jeziku C#, ki je namenjen sistemom Windows, kar pomeni, da program ni enostavno prenosljiv na druge platforme, kot je Linux.

Z Obeliksom so bili označeni korpus govorjene slovenščine Gos (Verdonik in Zwitter Vitez 2011), korpus besedil odnosov z javnostmi KoRP (Logar 2013), korpus šolskih pisnih izdelkov Šolar (Rozman et al. 2012) in korpus Gigafida; oznake iz Gigafide pa so (skupaj z besedili) prisotne še v korpusih KRES, ccGigafida in ccKRES (Erjavec in Logar 2012).

V okviru projekta Sporazumevanje v slovenskem jeziku je bil razvit tudi skladenjski razčlenjevalnik za slovenščino (Dobrovoljc et al. 2012) oz. natančneje: odvisnostnoskladenjski razčlenjevalnik MSTParser (McDonald et al. 2006) je bil naučen označevanja skladenjskih drevesnic na korpusu ssj500k. Razčlenjevalnik daje razmeroma dobre rezultate, je pa, kot je to tudi sicer običajno pri kakršnemkoli jezikoslovnem označevanju oz. razčlenjevanju, točnost zelo odvisna od zvrsti besedila – bolj ko zvrst besedila za označevanje odstopa od zvrsti učne množice, slabši so rezultati. Pri evalvaciji razčlenjevalnika se je izkazalo tudi to, da je točnost zelo odvisna od vrste odvisnostne povezave, saj sega od 54 do 96 %.

2.4 Druga orodja

Za označevanje imenskih entitet (NER) sta bili za slovenščino razviti dve orodji. Na IJS je bil razvit označevalec NER, ki uporablja strojno učenje na osnovi

pogojnih naključnih polj (Štajner et al. 2012), in to z modelom, naučenim na korpusu ssj500k. Program je sicer objavljen pod odprtokodno licenco, sta pa njegova namestitev in način uporabe razmeroma slabo dokumentirana, kar otežuje njegovo uporabo. V drugi raziskavi (Ljubešić et al. 2013) je bilo uporabljeno orodje StanfordNER (Finkel et al. 2005), ki ravno tako deluje na osnovi pogojnih naključnih polj. Tudi tu je bil model naučen na korpusu ssj500k, vendar v kombinaciji s korpusom spletne slovenščine slWaC (Ljubešić in Erjavec 2011). Ta omogoča boljši zajem distribucijskih značilk, ki se izkažejo za zelo koristne pri zmanjšanju števila napak, kot tudi izboljšanju priklica. Modeli za program so odprto dostopni, StanfordNER pa je vzdrževano in dokumentirano orodje.

Za luščenje terminov so bili za slovenski jezik opravljeni številni eksperimenti in izdelana mnoga orodja (Logar in Vintar 2008; Vintar 2009; 2010; Logar et al. 2013), vendar slednja niso odprto dostopna ali vzdrževana. Orodja temeljijo predvsem na kombinaciji jezikoslovnega védenja o terminih (predvsem nizih oblikoskladenjskih oznak, ki lahko predstavljajo termine) ter izrabi matematičnih lastnosti porazdelitve besed in besednih nizov v korpusih. Vsaj za slovenščino pri identifikaciji terminov še niso bile uporabljene metode strojnega učenja, obenem pa tudi ne obstajajo odprte učne množice, ki bi jih lahko za to uporabili.

3 SMERNICE ZA NADALJNJI RAZVOJ OZNAČEVANJA KORPUSOV

3.1 Izboljšanje označevalnih shem

Preden se posvetimo izboljšanju samih orodij oz. korpusov, na katerih se orodja učijo, je smiselno odpreti vprašanje označevalnih shem, na osnovi katerih so korpusi (ročno) označeni. Zasnovo teh shem bi bilo namreč koristno na novo premisliti in izvesti testiranja, kar bi povečalo točnost orodij, obenem pa ohranilo ali celo izboljšalo jezikoslovno informativnost kategorij posameznih ravni označevanja.

Slovnične informacije o posameznih besedah v korpusih ssj500k, Gigafida, KRES itd., kot tudi v oblikoskladenjskem leksikonu Sloleks, temeljijo na oblikoskladenjskih specifikacijah, razvitih v okviru projekta Jezikoslovno označevanje slovenščine JOS (Erjavec in Krek 2008). Ta sistem ima svoj izvor in je usklajen s specifikacijami MULTEXT (Ide in Véronis 1994) oz. MULTEXT-East, pri čemer so specifikacije MULTEXT-East 4.0 (Erjavec 2012) za slovenščino identične specifikacijam JOS in pokrivajo 12 jezikov, od tega skoraj vse slovanske.

Specifikacije JOS definirajo 12 besednih vrst: samostalnik, pridevnik, glagol, prislov, zaimek, števnik, predlog, veznik, členek, medmet, okrajšavo in neuvrščeno.

Večina besednih vrst ima pripisane oblikoskladenjske lastnosti, kot so npr. pri samostalniku vrsta (obči, lastni) ali sklon. Možne kombinacije besedne vrste in njihovih lastnosti so kodirane kot nizi, v katerih vsaki poziciji v nizu ustreza en atribut, enočrkovna koda pa poda njegovo vrednost. Tako npr. niz Somei pomeni besedna vrsta = samostalnik, vrsta = občno ime, spol = moški, število = ednina, sklon = imenovalnik. Kode v nizu in tudi lastnosti (torej atributi in njihove vrednosti) so prevedene v angleški jezik, da olajšajo uporabo sistema v mednarodnem okolju. Tako je npr. Somei po angleško Ncmsm oziroma Category = Noun, Type = common, gender = masculine, number = singular, case = nominative. Sistem JOS skupaj loči 1.902 različni oznaki, ki so v specifikacijah naštete, razstavljene po lastnostih in ilustrirane s primeri iz korpusa. Oznake JOS, kot je Somei, se uporabljajo v oblikoskladenjsko označenih korpusih, kjer služijo kot učna množica za učenje oblikoskladenjskih označevalnikov, s katerimi nato avtomatsko označujemo nove korpuse. Te oznake se uporabljajo tudi v oblikoskladenjskem leksikonu Sloleks, saj z njimi definiramo besedne oblike v paradigmi posameznih besed.

Oznake JOS se sicer res uporabljajo v velikem številu korpusov, vendar ni nujno, da je ta nabor oznak oz. lastnosti najprimernejši za vse aplikacije. Tako si lahko zamislimo nabor oznak, ki izpusti vse lastnosti, pri katerih se oblikoskladenjski označevalniki najbolj motijo (npr. sklon), ali pa vse pregibne lastnosti, če je za namene projekta dovolj besedam pripisati samo njihove leksikalne lastnosti. Takšne alternative, ki zmanjšajo velik nabor oznak JOS in s tem povečajo točnost označevalnikov, so se že pojavile: poglobljena študija v Krek (2011) predlaga več možnih redukcij oznak, medtem ko Erjavec (2015) omejuje nabor na 32 oznak, ki zajemajo samo besedno vrsto in nekaj njihovih leksikalnih lastnosti. Vseeno bi bilo potrebnih več raziskav, ki bi opredelile optimalen nabor oznak za posamezne potrebe.

V zadnjem času se je pojavila še ena zanimiva možnost, in sicer projekt Universal Dependencies (Nivre et al. 2015), v katerem se izdelujejo specifikacije in drevesnice za veliko število jezikov, mdr. tudi za slovenščino. Ob definiciji skladenjskih povezav prinaša projekt tudi univerzalen nabor oblikoskladenjskih lastnosti (z možnostjo jezikovnospecifičnih razširitev). Kljub temu, da težnja k univerzalnosti nujno privede do slabše prilagojenosti sheme za posamezni jezik, pa dejstvo, da je shema nato primerljiva z mnogimi drugimi jeziki, morda vseeno odtehta posamezne slabše rešitve.

Še bolj kot oblikoskladenjska raven označevanja potrebuje nadaljnje raziskave manj preizkušen nabor skladenjskih oznak in povezav projektov JOS ter Sporazumevanje v slovenskem jeziku (Erjavec et al. 2010; Arhar 2009), pri katerem npr. množično povezovanje pojavnic na koren drevesa predstavlja problem pri izdelavi

avtomatskih skladenjskih razčlenjevalnikov (Javoršek 2015). Tudi pri nadaljnjem razvoju skladenjskega sistema označevanj bi bilo zelo koristno upoštevati priporočila in prakso projekta Universal Dependencies.

Pomanjkljivost trenutnih kategorij se je pokazala tudi pri označevanju imenskih entitet, saj so raziskave (Štajner et al. 2012) pokazale, da z razdelitvijo »ostalih« imenskih entitet (torej tistih, ki niso osebna ali zemljepisna imena) na imena organizacij in »ostale« v učnem korpusu ssj500k dosežemo boljše rezultate prepoznavanja tudi pri drugih razredih. Z uvedbo nove kategorije smo v tem primeru ne samo obogatili nabor označevalnih kategorij, temveč tudi pripomogli h kvaliteti označevanja. To se sklada z ugotovitvami avtorjev češkega korpusa imenskih entitet CNCEC, pri katerem ločijo kar 62 kategorij imenskih entitet (Ševčíková et al. 2007). Avtorji so ugotovili tudi, da zmanjšanje števila kategorij imenskih entitet privede do slabših rezultatov označevanja. Tudi za slovenski jezik bi zato kazalo razmisliti o nadaljnjem povečanju števila kategorij za imenske entitete.

3.2 Izboljšanje točnosti orodij

Opisana orodja se med seboj zelo razlikujejo tako po kvaliteti označevanja kot po enostavnosti uporabe. Pri vseh bi bilo seveda koristno izboljšati točnost označevanja, saj je vsaka napaka problematična z dveh vidikov. Po eni strani predstavlja šum v podatkih, saj slovaropisec s svojo poizvedbo dobi tudi rezultate, ki so napačni. Tako npr. pri poizvedovanju po določeni lemi oz. osnovni obliki lahko besede, ki jim je bila pri lematizaciji pripisana napačna skupna osnovna oblika, zameglijo sliko njenih konkordanc, kolokacij, besednih skic itd. Vendar tu slovaropisec lahko vsaj pregleda primere in se odloči, kateri so pravilni, kar je sicer zamudno, a izvedljivo. Toliko bolj nevaren pa je pomanjkljiv priklic orodij, saj v tem primeru nekaterih podatkov slovaropisec enostavno ne vidi, ker mu jih orodja ne odkrijejo: če je neka beseda popolnoma ali večinoma narobe lematizirana, je z iskanjem po njeni lemi ne bomo našli ali bo ponujenih zelo malo zadetkov. Glavni cilj pri avtomatskem označevanju korpusov bi zato morala biti težnja po izboljšanju točnosti in priklica vseh obravnavanih metod – še zlasti v njihovih prvih korakih (tokenizacija, oblikoskladenjsko označevanje, lematizacija), saj vsaka napaka množi napake vseh nadaljnjih stopenj obdelave in s tem tudi otežuje leksikografsko analizo.

Za skoraj vse ravni označevanja se je že izkazalo, da je mogoče boljše rezultate doseči s strojnim učenjem kot z ročno napisanimi pravili. Strojno učenje potrebuje čim boljše ročno označene učne množice, te pa potrebujemo tudi za preizkušanje kvalitete delovanja, in to tako strojnega učenja kot ročno napisanih pravil. Zato bi bilo za izboljšanje delovanja večine orodij koristno povečati količino in predvsem

raznovrstnost ročno označenih korpusov. Pri tem ni nujno, da se označijo celotna besedila, saj lahko z metodami aktivnega učenja za ročno označevanje izbiramo samo primere, ki bodo označevalniku najbolj pomagali pri izboljšanju naučenega modela. Za različne ravni označevanja bi bilo koristno tudi povečanje podpornih podatkovnih virov, predvsem leksikonov in leksikalnih baz, saj ti nudijo veliko informacij o jeziku v že prečiščeni obliki.

Izpostaviti velja tudi konceptualni model takšnega označevanja oz. povečanja podpornih virov, ki gradi na »krepostnem krogu«: z dodatnimi ročno označenimi korpusi programe naučimo boljšega označevanja, s čimer lahko pripravimo boljšo osnovo za nadaljnji krog ročnega označevanja, ta krog oz. spiralo pa lahko ponovimo večkrat.

Za točnost vseh ravni označevanja je še posebej pomemben tokenizator, saj se njegove napake prenesejo v vse naslednje stopnje označevanja; napake v tokenizaciji pa seveda neposredno onemogočajo tudi iskanje napačno tokeniziranih besed. Za slovenščino je bilo vloženih že kar nekaj naporov v izgradnjo referenčnega tokenizatorja (Krek 2011), ki je v dobršni meri tudi že implementiran v sistemu Obeliks. Seveda bi bilo možno delovanje še izboljšati; tako npr. tokenizator trenutno ne prepozna »ga.« kot okrajšave v stavkih tipa »Spoštovana ga. Micka!«. Ob tem pa se je treba zavedati, da vsaka sprememba tokenizacije glede na že obstoječe (tudi ročno) označene korpuse pomeni medsebojno neskladje virov, kar ima negativne posledice tako za označevanje kot za luščenje slovarskih podatkov iz korpusov. Take primere najdemo, če npr. skupaj uporabljamo korpuse, označene s ToTaLe, in korpuse, označene z Obeliksom. Raziskava specifično spletnega besedišča, v kateri smo iskali ključne besede slWaCa glede na referenčni korpus KRES, je pokazala, da so najbolj »ključne« prav besede, ki so s ToTaLe tokenizirane drugače kot pa z Obeliksom (Erjavec in Ljubešić 2014).

Pri oblikoskladenjskem označevanju bi bilo koristno izvesti eksperimente, s katerimi bi ugotovili, katere metode oz. kombinacije metod res dajo najboljše rezultate. Tako je bilo npr. že pokazano, da daje uporaba metaučenja, ki kombinira rezultata Amebisovega označevalnika, ki temelji na pravilih, in statističnega označevalnika TnT, boljše rezultate kot pa katerikoli od obeh samostojnih označevalnikov (Rupnik et al. 2010).

3.3 Izboljšave tehničnih vidikov orodij

Poleg točnosti označevanja posameznih orodij bi bilo koristno orodja izboljšati tudi po njihovi tehnični plati, torej pri enostavnosti namestitve in uporabe ter pri možnostih in načinih njihove integracije.

V splošnem je najbolje uporabljati odprtokodna jezikovno in od računalniške platforme neodvisna orodja, ki temeljijo na strojnem učenju, so dobro dokumentirana, se vzdržujejo na kateri od platform za kontrolo sprememb (revision control systems), kot je npr. Git, ter imajo aktivno skupino razvijalcev in uporabnikov, vključno s forumom za vprašanja, prijavo napak ali predlogi izboljšav. Primera sta npr. Moses in, do neke mere, StanfordNER. Namensko razvita orodja za slovenski jezik imajo sicer lahko prednost, da jih lažje prilagodimo specifikam svojega jezika (ali še bolj kot to: jezikovnoteoretičnemu okviru), vendar pa je vprašljivo, ali delo z njihovim vzdrževanjem in prenosljivostjo to odtehta. Taka orodja sicer v neki točki razvoja lahko dajo boljše rezultate, vendar pa je zelo verjetno, da bo razvoj strojnega učenja prinesel vse boljše rezultate. Zato je bolj smiselno trud vlagati v razvoj označenih korpusov slovenskega jezika, ki lahko služijo kot dobre učne množice, kot pa v kompleksna na pravilih temelječa orodja namenjena samo slovenskemu jeziku.

V kontekstu izdelave označenih korpusov za slovaropisje lahko zavzamemo tudi stališče, da dostopnost orodij v resnici ni pomembna, vse dokler so bila uspešno uporabljena za označitev korpusa, vendar to otežuje razširitev in tudi vzdrževanje korpusov, poleg tega pa potem ta orodja ne bodo uporabna za druge namene in označevanje drugih, s slovaropisnim projektom nepovezanih projektov oz. raziskav. Z uporabo zaprtih orodij rezultati označevanja tudi niso preverljivi oz. ponovljivi.

Naslednje vprašanje je povezljivost posameznih orodij, tako glede na njihove vhodne in izhodne formate kot tudi glede vezanosti na določene računalniške platforme. Tako je npr. že omenjena implementacija Obeliksa, sicer verjetno trenutno najboljšega dostopnega oblikoskladenjskega označevalnika za slovenščino, vezana na operacijski sistem Windows, kar ga naredi slabo kompatibilnega z okoljem Linux, ki je tradicionalno bistveno bolj opremljeno z odprtokodnimi označevalniki in drugimi orodji. Po drugi strani pa je ToTaLe vezan na Linux, kar otežuje njegovo uporabo pod operacijskim sistemom Windows. Vendar pa se v zadnjem času pojavlja vedno več platform, ki omogočajo določanje in izvajanje spletnih delotokov, kot npr. WebLicht (Hinrichs et al. 2010). V takšnih sistemih posamezni programi oz. moduli tečejo kot spletni servisi na razpršenih računalnikih, medtem ko izvajanje delotoka kot celote (npr. tokenizacija → oblikoskladenjsko označevanje → lematizacija) prevzame centralni strežnik, ki po potrebi kliče spletne servise. Mogoče je tak način izvajanja označevanja res prihodnost, vendar pa je – posebej za obdelavo velikih korpusov – trenutna rešitev še vedno predvsem v lokalnem izvajanju označevanja na v gruče povezanih računalnikih, tipično z velikimi procesorskimi in pomnilniškimi kapacitetami. Zato je pomembno, da so označevalna orodja neodvisna od operacijskega sistema oz. platforme, na kateri se izvajajo. V praksi to pomeni, da so napisana v enem od standardnih odprtokodnih programskih jezikov, kot je npr. Java ali Python.

Poleg neodvisnosti od platforme je potrebno zagotoviti specifikacije medsebojno kompatibilnih vhodnih in izhodnih formatov orodij, podobno kot je bilo npr. narejeno v sklopu nemškega sistema spletnih orodij WebLicht; več o tem piše v razdelku 4.

Zanimiv raziskovalni in razvojni izziv je torej tudi arhitektura sistemov za označevanje besedil. Skoraj vse trenutne implementacije delujejo z izbiro najboljšega kandidata (glede na model) pri vsakem koraku označevanja. Vendar pa se najboljši kandidat pri nekem koraku lahko izkaže kot neustrezen, ko dobimo na voljo več informacij, saj šele kasnejše stopnje obdelave pravilno razdvoumijo med možnimi kandidati. Tako npr. šele z upoštevanjem skladnje lahko določimo, da »ga.« ni okrajšava, temveč zaimek, ki mu sledi konec stavka, kot v že omenjenem primeru »Videl sem ga. Micka ga je tudi videla.« Novejši trendi na tem področju so sistemi, ki namesto enostavnega cevovoda označevalnikov uporabljajo Bayesove mreže (Finkel et al. 2006), v katerih vsak označevalnik ustreza eni spremenljivki sistema, s čimer je nato mogoče izvesti približno sklepanje, ki najde globalno najboljše oznake.

3.4 Predlog verige označevalnih orodij

Za označevanje korpusov slovenskega jezika za namene slovaropisja smo predvideli tiste ravni, za katere je že sedaj mogoče uporabiti obstoječa orodja, raje kot vse možne oz. potencialno koristne, ki pa bi jih bilo potrebno najprej še (skoraj) v celoti razviti. V nadaljevanju podamo predlog verige orodij za označevanje, pri čemer smo izmed predstavljenih orodij izbrali tista, ki so odprtokodna tako glede programske opreme kot tudi glede modelov slovenščine. Pri vsakem orodju navajamo tudi razmeroma enostavne predloge za njegove izboljšave.

- **Obeliks**: tokenizacija, oblikoskladenjsko označevanje in lematizacija. Sistem bi bilo koristno ponovno implementirati v enem od standardnih programskih jezikov ter ga na novo naučiti oblikoskladenjskih in lematizacijskih modelov. Pri tem bi bilo koristno dodati možnost normalizacije besed, ki bi bila sicer implementirana v ločenem modulu.
- **Moses**: normalizacija besednih oblik, pri čemer bi bil program pospremljen z več modeli normalizacije, vsaj enim za nestandardno sodobno in drugim za zgodovinsko slovenščino. Odločitev, ali neko besedilo normalizirati in s katerim modelom, bi bila lahko bodisi avtomatska glede na vsebino posameznega besedila ali pa na osnovi metapodatkov, pripisanih besedilu.
- **MSTParser**: površinskoodvisnostno skladenjsko označevanje. Za označevanje lahko uporablja obstoječi model za skladenjsko analizo, pri čemer bi bilo koristno implementirati konverzijo med shemo JOS in slovenskimi Universal Dependencies ter razčlenjevalnik naučiti še teh.

Koristno bi bilo tudi mestoma popraviti učni korpus ssj500k in ga mogoče tudi povečati z zvrstmi besedil, ki so zaenkrat slabo zastopane v korpusu, se pa predvideva, da so skladenjsko drugačne kot pa že zajete. Koristno bi bilo tudi izvesti eksperimente s kakšnim drugim, sodobnejšim skladenjskim označevalnikom, saj bi mogoče dobili s tem boljše rezultate kot z MSTParserjem.

- **StanfordNER**: določanje imenskih entitet. Učno množico (trenutno samo ssj500k) bi bilo koristno povečati in predvsem narediti bolj raznovrstno.
- Kot rečeno, za označevanje terminov trenutno ni na voljo vzdrževanega in odprto dostopnega orodja, zato bi bilo luščenje terminologije verjetno treba programirati na novo, pri čemer pa bi bilo koristno uporabiti že izdelane nize oblikoskladenjskih vzorcev, ki predstavljajo potencialne termine.

Odprto ostaja še vprašanje, kako zgoraj našteta in precej raznovrstna orodja medsebojno povezati. Za učinkovito avtomatsko označevanje velikih korpusov je najboljša rešitev instalacija in paralelizacija verige označevalnikov na visokozmogljivih strežnikih Linux oz. gručah takšnih strežnikov, pri čemer je treba pretvorbo njihovih vhodno-izhodnih formatov implementirati tako, da so medsebojno kompatibilni. Opcija tu je sicer neposredno format TEI, predviden za končni zapis korpusa, vendar je za bolj učinkovito delovanje označevalnikov predvsem primeren zapis s kazalci, vse to pa obravnavamo v nadaljevanju.

4 Zapis oznak v korpusih

Korpusi imajo lahko zelo kompleksno strukturo, tako v metapodatkih kot jezikoslovnih oznakah. V Sloveniji so se za njihov zapis v veliki meri uveljavile smernice TEI (TEI 2013), ki pokrivajo vse zgoraj obravnavane ravni označevanja kot tudi nekatere druge. Smernice so dobro vzdrževane, saj zanje skrbi mednarodni konzorcij TEI, spremlja pa jih tudi obilica orodij za izdelavo konkretnih shem XML in pretvorb iz različnih formatov in vanje, kot je npr. Word v TEI ali TEI v HTML. Elementi TEI so poslovenjeni, izoblikovala pa se je tudi večja skupina uporabnikov na področju digitalne humanistike (Erjavec et al. 2004; Ogrin et al. 2013).

Skupno priporočilom TEI je, da večino jezikoslovnih oznak zapišemo kot element XML, npr. <w> za besedo ali <name> za ime. Prednost takšnega načina zapisa je neposredna razvidnost oznak in formalna preverljivost pravilnosti zapisa, oznake in tudi besedilo pa je razmeroma lahko popravljati. Rešitev z oznakami, ki so pridružene besedilu, ima tudi več potencialnih slabosti: oznake morajo biti pravilno gnezdene (XML podpira predvsem drevesne strukture), ob večanju števila oznak postanejo elementi XML nepregledni in datoteke z vsemi vsebovanimi

oznakami zelo velike. Zato se za sisteme, v katerih se predvideva povsem avtomatsko označevanje, pogosteje uporabljajo zapisi s kazalci: vhodno besedilo se ne spreminja, oznake posameznega orodja pa kažejo na ustrezna mesta v besedilu oz. v plasti katerih drugih oznak. Tak pristop npr. uporablja že omenjeni WebLicht (Hinrichs et al. 2010) oz. njegov skupni format za zapis korpusov TCF, je pa to tudi pristop, ki ga definira standard MAF, namenjen označevanju oblikoskladnje (ISO 24611, 2012).

Čeprav je zapis s kazalci tehnično bolj enostaven in omogoča večjo fleksibilnost, pri njem dosti težje odkrijemo napake in težje povežemo posamezne ravni označevanja. Predvsem pa podatkov, na katere oznake kažejo, nikakor ne smemo spreminjati, saj s tem kazalci postanejo neveljavni. To se izkaže za problematično v primerih, ko bi hoteli ročno oz. polavtomatsko popraviti (nekatere) oznake ali samo besedilo. Zapis TEI je tako primeren predvsem za referenčne korpuse, pri katerih bi želeli imeti čim bolje preverjene oznake in zapis korpusa, za arhivske namene pa besedilo shranjeno skupaj z vsemi oznakami na čim bolj berljiv način.

Tudi večina korpusov, ki se omenjajo v tej monografiji, je zapisana v TEI, vendar je uporaba teh priporočil kompleksna, poleg tega pa so se v več kot dveh desetletjih, odkar jih v Sloveniji uporabljamo za zapis korpusov, tudi spreminjala. Poleg tega se ob dodajanju novih oznak lahko izkaže, da so bile pretekle odločitve slabo posplošljive. Zato bi bilo koristno obstoječe korpuse ne samo ponovno označiti z boljšimi orodji in modeli, temveč tudi poenotiti njihovo kodiranje, ki bi nato postalo referenčno za nove korpuse, ki bodo potrebni za izdelavo slovarja sodobnega slovenskega jezika.

5 Zaključek

V prispevku smo podali pregled ravni predvsem avtomatskega označevanja, ki bi jih bilo smiselno zapisati v korpuse, ki bodo služili bodisi kot osnova za leksikografski opis sodobnega slovenskega jezika, z njim povezanih slovarskih ali drugih virov, ali za druge namene. Izpostavili smo serijo odprtokodnih in večinoma jezikovno neodvisnih jezikoslovnih označevalnikov ter njihovih modelov za slovenski jezik, kot tudi predloge za njihove izboljšave. Na kratko je bil opisan še predlagani sistem zapisa oznak, torej format korpus.

Oznake, zapisane v velikih korpusih, so vedno dodane avtomatsko, zato se je pri uporabi takšnih korpusov treba zavedati, da orodja delajo tudi napake, te pa imajo za posledico slabše luščenje slovaropisno ali drugače zanimivih informacij. Izboljšanje točnosti teh orodij torej tudi za naprej ostaja prednostna naloga.