2013 ‹ ŠTEVILKA 4 ‹ OKT. NOV. DEC. ‹ LETNIK XXI ‹ ISSN 1318-1882

04 UPORABNA INFORMATIKA

Izpitni centri ECDL

ECDL (European Computer Driving License), ki ga v Sloveniji imenujemo evropsko računalniško spričevalo, je standardni program usposabljanja uporabnikov, ki da zaposlenim potrebno znanje za delo s standardnimi računalniškimi programi na informatiziranem delovnem mestu, delodajalcem pa pomeni dokazilo o usposobljenosti. V Evropi je za uvajanje, usposabljanje in nadzor izvajanja ECDL pooblaščena ustanova ECDL Fundation, v Sloveniji pa je kot član CEPIS (Council of European Professional Informatics) to pravico pridobilo Slovensko društvo INFORMATIKA. V državah Evropske unije so pri uvajanju ECDL močno angažirane srednje in visoke šole, aktivni pa so tudi različni vladni resorji. Posebno pomembno je, da velja spričevalo v 148 državah, ki so vključene v program ECDL. Doslej je bilo v svetu izdanih že več kot 11,6 milijona indeksov, v Sloveniji več kot 17.000, in podeljenih več kot 11.000 spričeval. Za izpitne centre v Sloveniji je usposobljenih sedem organizacij, katerih logotipe objavljamo.

AGORA

LJUDSKA UNIVERZA MURSKA SOBOTA

ŠOLSKI CENTER Novo mesto

spin

ISER

LJUDSKA UNIVERZA CELJE
Cankarjeva ulica 1, 3000 Celje, Slovenija
Telefon:(03) 428-67-50 Telefax: (03) 428-67-50

Micro Team

U P O R A B N A
I N F O R M A T I K A
2013 ŠTEVILKA 3 OKT/NOV/DEC LETNIK XXI ISSN 1318-1882

Uvodnik

Znanstveni prispevki

Strokovni prispevki

Informacije

U P O R A B N A
I N F O R M A T I K A

2013 ŠTEVILKA 4 OKT/NOV/DEC LETNIK XXI ISSN 1318-1882

Ustanovitelj in izdajatelj
Slovensko društvo INFORMATIKA
Litostrojska cesta 54, 1000 Ljubljana

Predstavnik
Niko Schlamberger

Odgovorni urednik
Jurij Jaklič

Gostujoča urednica
Špela Vintar

Tehnična urednica
Mira Turk Škraba

Lektoriranje
Mira Turk Škraba (slov.)
Špela Vintar (angl.)

Oblikovanje
KOFEIN DIZAJN, d. o. o.

Prelom in tisk
Boex DTP, d. o. o., Ljubljana

Naklada
600 izvodov

Naslov uredništva
Slovensko društvo INFORMATIKA
Uredništvo revije Uporabna informatika
Litostrojska cesta 54, 1000 Ljubljana
www.uporabna-informatika.si

Revija izhaja četrtletno. Cena posamezne številke je 20,00 EUR. Letna naročnina za podjetja 85,00 EUR, za vsak nadaljni izvod 60,00 EUR, za posameznike 35,00 EUR, za študente in seniorje 15,00 EUR. V ceno je vključen DDV.

Izdajanje revije Uporabna informatika v letu 2013 sofinancira Javna agencija za raziskovalno dejavnost Republike Slovenije.

Revija Uporabna informatika je od številke 4/VII vključena v mednarodno bazo INSPEC.

Revija Uporabna informatika je pod zaporedno številko 666 vpisana v razvid medijev, ki ga vodi Ministrstvo za kulturo RS.

Revija Uporabna informatika je vključena v Digitalno knjižnico Slovenije (dLib.si).

Vabilo avtorjem

V reviji Uporabna informatika objavljamo kakovostne izvirne članke domačih in tujih avtorjev z najširšega področja informatike v poslovanju podjetij, javni upravi in zasebnem življenju na znanstveni, strokovni in informativni ravni; še posebno spodbujamo objavo interdisciplinarnih člankov. Zato vabimo avtorje, da prispevke, ki ustrezajo omenjenim usmeritvam, pošljejo uredništvu revije po elektronski pošti na naslov ui@drustvo-informatika.si.

Avtorje prosimo, da pri pripravi prispevka upoštevajo navodila, objavljena v nadaljevanju ter na naslovu http://www.uporabna-informatika.si.

Za kakovost prispevkov skrbi mednarodni uredniški odbor. Članki so anonimno recenzirani, o objavi pa na podlagi recenzij samostojno odloča uredniški odbor. Recenzenti lahko zahtevajo, da avtorji besedilo spremenijo v skladu s priporočili in da popravljeni članek ponovno prejmejo v pregled. Uredništvo pa lahko še pred recenzijo zavrne objavo prispevka, če njegova vsebina ne ustreza vsebinski usmeritvi revije ali če članek ne ustreza kriterijem za objavo v reviji.

Pred objavo članka mora avtor podpisati izjavo o avtorstvu, s katero potrjuje originalnost članka in dovoljuje prenos materialnih avtorskih pravic. Nenaročenih prispevkov ne vračamo in ne honoriramo. Avtorji prejmejo enoletno naročnino na revijo Uporabna informatika, ki vključuje avtorski izvod revije in še nadaljnje tri zaporedne številke.

S svojim prispevkom v reviji Uporabna informatika boste prispevali k širjenju znanja na področju informatike. Želimo si čim več prispevkov z raznoliko in zanimivo tematiko in se jih že vnaprej veselimo.

Uredništvo revije

Navodila avtorjem člankov

Članke objavljamo praviloma v slovenščini, članke tujih avtorjev pa v angleščini. Besedilo naj bo jezikovno skrbno pripravljeno. Priporočamo zmernost pri uporabi tujk in – kjer je mogoče – njihovo zamenjavo s slovenskimi izrazi. V pomoč pri iskanju slovenskih ustreznic priporočamo uporabo spletnega terminološkega slovarja Slovenskega društva Informatika Islovar (www.islovar.org).

Znanstveni članek naj obsega največ 40.000 znakov, strokovni članki do 30.000 znakov, obvestila in poročila pa do 8.000 znakov.

Članek naj bo praviloma predložen v urejevalniku besedil Word (*.doc ali *.docx) v enojnem razmaku, brez posebnih znakov ali poudarjenih črk. Za ločilom na koncu stavka napravite samo en prazen prostor, pri odstavkih ne uporabljajte zamika.

Naslovu članka naj sledi za vsakega avtorja polno ime, ustanova, v kateri je zaposlen, naslov in elektronski naslov. Sledi naj povzetek v slovenščini v obsegu 8 do 10 vrstic in seznam od 5 do 8 ključnih besed, ki najbolje opredeljujejo vsebinski okvir članka. Pred povzetkom v angleščini naj bo še angleški prevod naslova, prav tako pa naj bodo dodane ključne besede v angleščini. Obratno velja v primeru predložitve članka v angleščini.

Razdelki naj bodo naslovljeni in oštevilčeni z arabskimi številkami.

Slike in tabele vključite v besedilo. Opremite jih z naslovom in oštevilčite z arabskimi številkami. Vsako sliko in tabelo razložite tudi v besedilu članka. Če v članku uporabljate slike ali tabele drugih avtorjev, navedite vir pod sliko oz. tabelo. Revijo tiskamo v črno-beli tehniki, zato barvne slike ali fotografije kot original niso primerne. Slik zaslonov ne objavljamo, razen če so nujno potrebne za razumevanje besedila. Slike, grafikoni, organizacijske sheme ipd. naj imajo belo podlago. Enačbe oštevilčite v oklepajih desno od enačbe.

V besedilu se sklicujte na navedeno literaturo skladno s pravili sistema APA navajanja bibliografskih referenc, najpogosteje torej v obliki: (Novak & Kovač, 2008, str. 235). Na koncu članka navedite samo v članku uporabljeno literaturo in vire v enotnem seznamu po abecednem redu avtorjev, prav tako v skladu s pravili APA. Več o APA sistemu, katerega uporabo omogoča tudi urejevalnik besedil Word 2007, najdete na strani http://owl.english.purdue.edu/owl/resource/560/01/.

Članku dodajte kratek življenjepis vsakega avtorja v obsegu do 8 vrstic, v katerem poudarite predvsem strokovne dosežke.

Spoštovane bralke in spoštovani bralci,

tokratna številka revije Uporabna informatika je posvečena področju jezikovnih tehnologij. Pomena jezika kot osnovnega sredstva sporazumevanja med ljudmi in temeljnega nosilca kulturne identitete verjetno ni treba posebej izpostavljati, saj se ti dve vlogi skozi zgodovino človeštva nista bistveno spreminjali. V sodobnem svetu poteka velik del sporočanja prek digitalnih medijev, ne komuniciramo le z ljudmi, ampak tudi z napravami, potreba po medkulturnem in medjezikovnem prenosu informacij pa je večja kot kadar koli prej. Jezikovne tehnologije so tako tesno prepletene z razvojem informacijskih tehnologij in pomembno vplivajo na številna področja človekovega delovanja od pisanja in branja besedil v maternem in tujih jezikih, iskanja podatkov na spletu, govornega upravljanja naprav in računalniškega prevajanja, pa vse do rudarjenja podatkov in odkrivanja novega znanja v besedilih.

Tematska številka o jezikovnih tehnologijah se časovno umešča v prelomno leto za jezikovnotehnološki razvoj v slovenskem prostoru, saj je bila julija letos sprejeta resolucija o nacionalnem programu za jezikovno politiko 2014–2018, ki zagotavljanje tehnološko podprtih jezikovnih virov in orodij umešča med najvišje prioritete, v tem okviru pa med drugim predvideva vrsto ukrepov za boljšo opremljenost slovenščine s prosto dostopnimi digitalnimi korpusi, leksikoni, slovarji in orodji za računalniško obdelavo jezika v eno- in večjezičnem kontekstu, v različnih medijih (govorni, pisni, znakovni) in za različne potrebe uporabnikov.

V tej številki objavljeni prispevki naslavljajo različne jezikovnotehnološke vidike, pri tem pa so pregledno zajeta področja jezikovnih virov, terminologije, temeljnih jezikovnih pripomočkov ter govornih in prevajalskih tehnologij. Prispevek Tomaža Erjavca se ukvarja z jezikovnotehnološko obdelavo starejših besedil, kar je pomemben vidik digitalizacije slovenske besedilne dediščine in zagotavljanja iskanja po polnih besedilih naše zgodovine. Peter Holozan se v svojem prispevku posveča samodejnemu pregledovanju in postavljanju vejic v slovenščini, hkrati pa članek daje tudi vpogled v dva glavna pristopa k modeliranju jezika, s pravili in s statističnimi metodami. Gregor Donaj s sodelavci pregledno predstavlja vire in tehnologije za razpoznavanje govorjene slovenščine, kar je ena od najbolj zahtevnih, obenem pa tudi zelo potrebnih jezikovnih aplikacij. Prispevek Špele Vintar pregledno predstavlja sodobne tehnologije za prevajanje, ki korenito spreminjajo ne le vidik večjezičnosti v informacijski družbi, ampak tudi poklicni profil prevajalcev. Strokovni prispevek Tomaža Turka in Katarine Puc pa posega v področje terminografije, in sicer avtorja opisujeta razvojno pot največjega slovenskega slovarja informatike Islovar, ki bo v kratkem zaživel v novi podobi in s sodobnejšo spletno programsko rešitvijo.

Jezikovna industrija je ena najhitreje rastočih na svetu, informacijska podpora za slovenščino pa ključni dejavnik za prepoznavnost države in kulturno, znanstveno in gospodarsko uspešnost njenih prebivalcev. V beli knjigi Slovenski jezik v digitalni dobi,[1] ki je izšla lani pod avtorstvom Simona Kreka v okviru evropskega projekta META-NET, so predstavljeni primerljivi podatki o jezikovni opremljenosti in jezikovnih tehnologijah za vseh (tedanjih) 23 uradnih evropskih jezikov in za še nekatere druge. Slovenščina se uvršča med slabše opremljene jezike, pri čemer je kakovost obstoječih virov sicer zadovoljiva, kot največjo težavo pa študija izpostavlja manjkajoče vire in orodja ter težave pri njihovem dolgoročnem vzdrževanju in distribuciji.

Ob izidu tematske Uporabne informatike si lahko zato le zaželimo, da bi bile jezikovne tehnologije za slovenščino v prihodnosti vse bolj uporabne, dostopne in kakovostne, to pa bo uresničljivo le ob ustrezni razvojni politiki države in usklajenih naporih raziskovalcev, razvijalcev in uporabnikov.

Špela Vintar,
gostujoča urednica

[1] Bela knjiga je dostopna na http://www.meta-net.eu/whitepapers/volumes/slovene.

Posodabljanje starejše slovenščine

Tomaž Erjavec, Institut Jožef Stefan, Odsek za tehnologije znanja, Jamova cesta 39, 1000 Ljubljana
tomaz.erjavec@ijs.si

Izvleček

V prispevku obravnavamo metodo za posodabljanje besed v starejših slovenskih besedilih, ki vključuje posodabljanje besednih oblik s pomočjo računalniških leksikonov in pravil za transkripcijo, oblikoskladenjsko označevanje in lematizacijo. Posodabljanje je koristno predvsem pri iskanju po polnem besedilu digitalnih knjižnic naše kulturne dediščine, pa tudi kot način, da starejša besedila približamo sodobnemu bralcu. Program za posodabljanje uporablja jezikovne vire starejše slovenščine IMP, ki vključujejo ročno označeni korpus besedil in leksikon starejše slovenščine, za oblikoskladenjsko označevanje in lematizacijo pa modele, naučene na virih sodobne slovenščine, razvitih v okviru projekta Sporazumevanje v slovenskem jeziku. Prispevek predstavi uporabljene vire, program za jezikoslovno označevanje ToTrTaLe, evalvacijo natančnosti programa in smernice za nadaljnje delo.

Ključne besede: starejša slovenščina, jezikovne tehnologije, jezikovni viri za slovenski jezik.

Abstract

Modernizing Historical Slovene

The paper presents a method for modernising words in historical Slovene texts, which includes modernising word-forms with the help of computational lexicons and transcription rules, morphosyntactic tagging, and lemmatisation. Modernisation is useful for full-text search in cultural heritage digital libraries as well as a way to make older texts more accessible to today's readers. The program for modernisation uses the IMP language resources for historical Slovene, which include a hand-annotated text corpus and a lexicon of historical Slovene, while morphosyntactic tagging and lemmatization rely on models trained on resources for contemporary Slovene, which were developed in the scope of the "Communication in Slovene" project. The paper introduces the language resources, the ToTrTaLe program for linguistic annotation, an evaluation of the accuracy of the program and directions for future research.

Key words: historical Slovene, language technologies, language resources for Slovene.

1 UVOD

V zadnjih letih smo priča hitremu razmahu digitalnih knjižnic, pri čemer je veliko dostopnih besedil starejšega datuma, saj ni ovir za njihovo razširjanje, ker so jim potekle avtorske pravice, ob tem pa so taka besedila zanimiva za seznanjanje in preučevanje kulturne dediščine posameznih narodov. Za slovenski jezik sta največji digitalni knjižnici dLib.si (Krstulović in Šetinc, 2005) in projekt Googlovih knjig. Ta dela so tipično dostopna predvsem kot faksimili, v najboljšem primeru s predogledom avtomatsko razpoznanega besedila, v katerem pa je zaradi poškodb papirja, starega tiska in uporabe bohoričice veliko napak. Besedila tudi niso strukturno označena, kar onemogoča npr. generiranje kazala in stavljenje besedila. Obstaja tudi več manjših, a zato bolj natančno obdelanih digitalnih knjižnic slovenske pisne kulturne dediščine,[1] na prvem mestu projekt »Slovenska leposlovna klasika« na Wikiviru, kot tudi portal Sistory (Šorn in Hadalin, 2010), knjižnica eZISS (Ogrin in Erjavec, 2009) in veliko projektov posameznih knjižnic.

Za iskanje po polnem besedilu digitalnih knjižnic je, vsaj za jezike z bogato morfologijo, kot je slovenščina, zelo koristno besedila predhodno lematizirati, torej vsaki besedi pripisati njeno osnovno obliko, npr. »ljubezen« za besedne oblike »ljubezni«, »ljubeznijo« itd. Šele tako bo namreč poizvedba za »ljubezen« vrnila tudi besedila s katero koli pregibno obliko te besede. Za sodobno standardno slovenščino je bilo razvitih že več lematizatorjev, tudi prosto dostopnih (Erjavec in Džeroski, 2004; Juršič idr., 2010; Logar Berginc idr., 2012), pri čemer bolj kakovostni najprej opravijo oblikoskladenjsko označevanje, pri čemer vsaki besedni pojavnici pripišejo njene oblikoskladenjske lastnosti, npr. »občni samostalnik moškega spola v orodniku ednine«, saj je v splošnem šele s to informacijo mogoče neko besedno obliko tudi pravil-

* Delo, objavljeno v tem članku, sta podprla projekt EU IP IMPACT *Improving Access to Text* in nagrada Google *Developing Language Models of Historical Slovene* ter raziskovalni program P2-0103 Tehnologije znanja.

[1] Podroben, čeprav že rahlo zastarel pregled je podan v Hladnik (2009).

no lematizirati. Tako je npr. za besedno obliko hotela treba vedeti, ali je glagol ali samostalnik, da ji lahko pripišemo bodisi lemo hoteti bodisi lemo hotel. Za pravilno lematizacijo neznanih besed pa je oblikoskladenjska oznaka še posebno potrebna.

Sodobni lematizatorji in oblikoskladenjski označevalniki se modela jezika naučijo samodejno na podlagi vnaprej pripravljenih jezikovnih virov. Za razliko od ročno napisanih pravil imajo induktivno naučeni modeli prednost, da so bolj robustni in lahko (razmeroma) uspešno obdelajo tudi neznane besede, zato pa potrebujejo za učne množice ročno izdelane jezikovne vire, tj. leksikone za lematizatorje in označene korpuse za oblikoskladenjske označevalnike. Izdelava dovolj natančnih, obsežnih in raznovrstnih jezikovnih virov za posamezen jezik je drag in dolgotrajen postopek, vendar je za slovenščino v zadnjem času postalo dostopnih večje število takšnih virov, predvsem v okviru projektov Jezikoslovno označevanje slovenskega jezika (JOS) in Sporazumevanje v slovenskem jeziku (SSJ), tako da izdelava induktivnih orodij ni več nepremostljiva težava; kot omenjeno, sedaj obstajajo tudi že vnaprej naučeni prostodostopni lematizatorji in oblikoskladenjski označevalniki za sodobno standardno slovenščino.

Stanje pa je drugačno za računalniško obravnavo starejše slovenščine, saj se ta razlikuje od sodobnega jezika, zaradi česar z obstoječimi programi zanjo dobimo zelo slabe rezultate. Besede so se včasih pisale drugače, njihov zapis se je skozi zgodovino tudi spreminjal, ob tem pa pisni jezik ni bil standardiziran, tako da lahko za isto besedo tudi v istem časovnem obdobju najdemo več zapisov. Če k temu prištejemo še bohoričico, ki so jo uporabljali do srede devetnajstega stoletja, ima lahko posamezna lema zelo veliko število oblik, ki so težko predvidljive vnaprej. Tako za lemo ljubezen v korpusu starejših besedil poleg sodobnih oblik ljubezen, ljubezni in ljubeznijo najdemo še ljubesni, ljubesin, lubeſn, lubesen, lubesni, ljubesen, lubeſne, lubeſni, ljubesnijo, ljubezin, lubesnio, lubesne, lubesn, lubiesn in lubiesen. Dodaten problem so besede, ki jih ne uporabljamo več, kot npr. »bukvovez«, ki je danes knjigovez, saj od uporabnika, ki bi rad iskal po besedilih digitalne knjižnice, težko pričakujemo, da se bo zavedal vseh zastarelih ustreznic sodobnim besedam.

V prispevku predstavimo program, ki starejše slovenske besede posodobi, jih oblikoskladenjsko označi in lematizira. V drugem razdelku najprej predstavimo jezikovne vire, ki so omogočili izdelavo programa, v tretjem razdelku opišemo delovanje programa, v četrtem ocenimo njegovo točnost in v petem razdelku podamo sklepe in smernice za nadaljnje delo.

2 UPORABLJENI JEZIKOVNI VIRI

Za označevanje starejših besed uporabljamo več jezikovnih virov, bodisi neposredno ali pa za učenje modelov za posamezne ravni jezikoslovne analize. V tem razdelku opišemo te vire, ki so uporabni tudi zunaj konteksta posodabljanja starejših besedil. Vsi so zapisani po mednarodnih standardih in priporočilih in prosto dostopni pod eno od licenc Creative Commons, tako da so čim bolj odprti (Erjavec, 2009) in lahko v največji meri spodbujajo napredek jezikovnih tehnologij za slovenski jezik.

Večina predstavljenih virov je zapisana skladno s smernicami za zapis besedil TEI, Text Encoding Initiative Guidelines (TEI, 2007). Smernice temeljijo na XML, opredeljujejo formalni zapis besedil za znanstvene namene in se uporabljajo za večino kompleksnejših izdaj v digitalnih knjižnicah, za jezikoslovno označene korpuse, za računalniške slovarje itd. Smernice TEI in s tem spodaj našteti viri so usklajeni z ustreznimi standardi W3C, ISO in IANA, npr. pri kodah za označevanje časov in jezikov. Kot primer izpostavimo oznako za bohoričico, ki do sedaj ni imela svoje standardizirane kode. V postopku izdelave virov starejše slovenščine smo na IANA (Internet Assigned Numbers Authority) prijavili kodo za podjezik »sl-bohoric«, ki je namenjena za označevanje slovenskih besedil, zapisanih v bohoričici, in – čeprav naši viri ne vsebujejo teh pisav[2] – še za »sl-metelko« in »sl-dajnko«.

2.1 Zbirka starejših slovenskih besedil IMP

Podlaga za izdelavo vseh drugih jezikovnih virov starejše slovenščine (Erjavec, 2012a) je zbirka besedil, imenovana IMP, ki je zasnovana kot digitalna knjižnica. Zbirka vsebuje tiskana besedila, večinoma celotne knjige, ki so predstavljene tako s faksimili kot z ročno pregledanimi in označenimi prepisi besedil. IMP trenutno vsebuje 658 del oz. okoli 46.000 strani ali 14 milijonov besed. S par izjemami obsegajo dela obdobje od konca 18. stoletja do leta 1918, večina pa jih je iz druge polovice 19. stoletja.

[2] Zbirka IMP sicer vsebuje knjigo Čelarstvo (Čebelarstvo) Petra Dajnka (1831), ki je zapisana v dajnčici, vendar v prepisu uporabljamo gajico, saj za dajnčico ne obstajajo znaki unikod niti ustrezni fonti za prikaz.

Stopnja označevanja TEI se razlikuje glede na digitalni vir posameznega dela, v vseh primerih pa vsebuje metapodatke (kolofon TEI), prelome strani s kazalci na faksimile, naslove razdelkov in odstavke, tipično pa tudi oznake za posebne dele besedila, kot so verzi, opombe, tiskarska znamenja, uredniški popravki, tuje besede itd. Na spletu je zbirka dostopna v obliki digitalne knjižnice z več kazali, pri čemer je vsaka enota svoja datoteka HTML, samodejno prevedena s stili TEI XSLT iz izvornega zapisa zbirke v XML/TEI.

Na sliki 1 ilustriramo iztržek ene od knjig iz zbirke IMP v zapisu TEI, pri čemer element <pb> pomeni prelom strani, nato se začne razdelek besedila ( ), ki vsebuje naslov (<head>) in začetek prve kitice (<lg>); ta je nato sestavljena iz vrstic (<l>), ki lahko vsebujejo tudi opombe (<note>). Elementi imajo tudi atribute, ki vsebujejo npr. identifikator (@xml:id), preko katerega je mogoče kazati na določen element, opis prikaza elementa (@rend), kazalko na faksimile (pb/@facs) ali dejstvo, da je opomba avtorjeva (note[@type=«authorial«]) in ne uredniška.

```
<pb facs="#WIKI00009-019" n="19"
xml:id="pb.019"/>
<div xml:id="wv-1._Dershi_ali_vmirajozha_.
C5.BFkopo.C5.BFt.">
    <head rend="centered italic">1. Dershi
    ali vmirajozha ſkopoſt.</head>
    <lg>
        <l>Dershi<note xml:id="ref1"
        type="authorial">Dershi, Paſko:
        peſje iména, <hi
        rend="gothic">Hundsnahmen mie
        z. B. Phylar.</hi>
        </note>, ker je v' neki nozhi</l>
        <l>Ne satiſnil ſvojih ózhi,</l>
        <l>Da je sveſti varih bil;</l>
        …
```

Slika 1: **Zapis TEI iztržka besedila iz zbirke besedil IMP**

2.2 Ročno označeni korpus starejših slovenskih besedil goo300k

Iz zbirke IMP smo vzorčili 1.100 strani iz 90 enot in vsako besedno pojavnico (nekaj manj kot 300.000) ročno označili z več jezikoslovnimi lastnostmi, s čimer smo dobili referenčni korpus starejše slovenščine po imenu goo300k (Erjavec, 2012b). Označene jezikoslovne lastnosti so:

1. sodobna ustreznica, torej besedna oblika, kot se piše danes, napisana z malimi črkami, pri čemer za zastarele (izumrle) besedne oblike upoštevamo pravila sodobnega pravopisa;
2. lema oz. osnovna oblika sodobne ustreznice;
3. najbližje sodobne ustreznice oz. kratka razlaga pomena (samo za zastarele besede);
4. leksikalni del oblikoskladenjske oznake JOS (razloženo v nadaljevanju).

Zapis korpusa ponazarja slika 2 z besedilom, ki se glasi: »*Pri vkvartirjanju ni drugači*.« Ta stavek (<s>) ima označene besede (<w>), ločila (<pc>) in presledke (<c>), besede pa nosijo informacijo o lemi (w/@lemma) in oblikoskladenjski oznaki (w/@ana). V primerih, ko se posodobljena beseda (ki je vedno napisana z malimi črkami) razlikuje od besedne oblike iz korpusa, se lahko odločimo (<choice>), ali želimo upoštevati izvorno (<orig>) ali posodobljeno obliko (<reg>). Pri zastarelih besedah je dodan opis (<desc>), sestavljen iz sodobne ustreznice oz. razlage (<gloss>) in vira te razlage (<bibl>); v podanem primeru je bil to kar (širši) kontekst, v katerem se je pojavila beseda »*vkvartirjanju*«. Zapis je bolj kompleksen, kot se zdi potrebno, vendar mora zajeti tudi primere, ko je ena zgodovinska beseda pisana kot več sodobnih ali obratno, npr. »po noči« proti »ponoči«.

```
<s>
    <choice>
        <orig><w>Pri</w></orig>
        <reg><w lemma=«pri« na=«#S«>pri</
        w></reg>
    </choice>
    <c> </c>
     <choice>
        <orig><w>vkvartirjanju</w></orig>
        <reg><w lemma=«ukvartiranje«
        ana=«#Ncn«>ukvartiranju</w>
            <desc><gloss>prenočevanje</
            gloss><bibl>kontekst</bibl></desc>
        </reg>
    </choice>
    <c> </c>
    <w lemma=«biti« ana=«#Va«>ni</w>
    <c> </c>
    <choice>
        <orig><w>drugači</w></orig>
        <reg><w lemma=«drugače«
        ana=«#Rgp«>drugače</w></reg>
     </choice>
    <pc>.</pc>
</s>
```

Slika 2: **Primer iz ročno označenega korpusa goo300k**

Čeprav je natančna definicija vsake od oznak kompleksna, saj v jeziku vedno srečujemo mejne primere, je osnovni pomen vsake od njih vseeno intuitivno jasen. Izjema so oblikoskladenjske oznake, zato jih podrobneje opišemo v nadaljevanju.

Oblikoskladenjske oznake JOS so kratki nizi (npr. »Ggdn«), ki jih lahko pripišemo posamezni besedni pojavnici v korpusu (ali besedni obliki v leksikonu) in kodirajo oblikoskladenjske lastnosti (npr. »glagol, vrsta=glavni, vid=dovršni, oblika=nedoločnik«). Nabor teh oznak (preko 1.900) za slovenski jezik in njihova preslikava v lastnosti so definirane v oblikoskladenjskih specifikacijah JOS (Erjavec in Krek, 2008). Na spletu so dostopne celotne specifikacije tako v izvornem zapisu TEI kot v izvedenem HTML, na voljo pa so tudi tabele, ki oznake preslikajo v lastnosti oz. iz slovenskega v angleški jezik (npr. »Ggdn« ≡ »Vmen« ≡ »Verb, Type=main, Aspect=perfective, VForm=infinitive«). Oznake JOS uporabljajo raznovrstni viri sodobne slovenščine, med drugim v nadaljevanju opisana računalniški leksikon Sloleks in učni korpus ssj500k.

Pri izdelavi jezikovnih virov starejše slovenščine je bil poudarek na ročnem označevanju sodobne oblike in leme, ne pa oblikoskladenjskih lastnosti, kar je zelo zamudno delo. Vseeno smo želeli imeti ročno preverjene vsaj leksikalne lastnosti posameznih lem, zato smo kompleksen nabor vseh oznak JOS reducirali s skoraj dva tisoč na 32. V naboru JOS je tako npr. za besedno pojavnico »ni« zapisano, da je »glagol vrsta=pomožni oblika=sedanjik oseba=tretja število=ednina nikalnost=zanikani«, v goo300k pa samo »glagol vrsta=pomožni« oz. »Va«, ker uporabljamo angleške oznake. Specifikacije oblikoslovnih lastnosti in oznak IMP so, tako kot JOS, tudi formalno zapisane in dostopne na spletu.

2.3 Leksikon starejše slovenščine IMP

Leksikon vsebuje zajete podatke iz korpusa, sestavljen pa je iz gesel, pri čemer posamezno geslo vsebuje lemo, njene oblikoskladenjske lastnosti in (za zastarele besede) sodobne ustreznice, nato seznam sodobnih besednih oblik, za vsako od teh njene zgodovinske ustreznice in nekaj primerov (konkordanc) iz besedil. Leksikon je pretvorjen iz korpusa goo300k, poleg tega pa dopolnjen z ročno obdelanimi pogostejšimi besedami iz večje podmnožice zbirke IMP. Ker leksikon izvira iz označenih korpusnih primerov, so v njem zajete samo dejansko izpričane oblike oz. njihove oznake, zato leksikon tipično ne vsebuje celotnih pregibnih paradigem (tj. vseh besednih oblik) posameznih lem.

Leksikon vsebuje več kot 80.000 zgodovinskih oblik, 58.000 sodobnih besednih oblik in 28.000 lem. Štete so tudi »besede«, kot so cifre, zatipkane in tuje besede, pa tudi besede, ki so enake tistim v sodobni slovenščini. Če štejemo samo vnose, ki imajo vsaj eno besedno obliko različno od sodobne, dobimo okoli 36.000 zgodovinskih oblik, 25.000 sodobnih oblik in 12.000 lem, med katerimi je 4.000 lem zastarelih, zato imajo tudi dodano razlago. Leksikon je dostopen na spletu v formatu HTML, ki je s posebej zato napisanim slogom XSLT pretvorjen iz izvornega TEI/XML.

2.4 Oblikoslovni leksikon sodobne slovenščine Sloleks

Za posodabljanje in lematizacijo potrebujemo tudi leksikon sodobne slovenščine, pri čemer uporabljamo oblikoskladenjski leksikon sodobne slovenščine Sloleks (Arhar, 2009), ki vsebuje okoli 100.000 lem, vse njihove pregibne oblike z oblikoskladenjskimi lastnostmi in s številom pojavitev v korpusu Gigafida, vsega skupaj skoraj 2,800.000 oblik. Leksikon za razliko od drugih naštetih virov ni zapisan v shemi TEI, temveč po XML, ki sledi LMF (Lexicon Markup Framework), standardu ISO 24613:2008 za predstavitev računalniških leksikonov. Ker je struktura LMF razmeroma zahtevna za uporabo, vsebuje pa tudi podatke, ki jih mnoge aplikacije ne potrebujejo, smo leksikon pretvorili še v preprost tabelarični format, v katerem je vsak vnos (vrstica) sestavljen iz besedne oblike, leme, oblikoskladenjske oznake in frekvence tega trojčka na milijon besed. Kot primer podamo v sliki 3 paradigmo samostalnika »skopost«, pri čemer frekvenca nič pomeni, da tega trojčka program ni identificiral v korpusu.

skopostih	skopost	Ncfdl	0.000000
skopostih	skopost	Ncfpl	0.000000
skopostim	skopost	Ncfpd	0.000000
skoposti	skopost	Ncfdn	0.000000
skoposti	skopost	Ncfdg	0.000000
skoposti	skopost	Ncfda	0.000000
skoposti	skopost	Ncfsd	0.000010
skoposti	skopost	Ncfsl	0.000088
skoposti	skopost	Ncfsg	0.000131
skoposti	skopost	Ncfpn	0.000001
skoposti	skopost	Ncfpg	0.000003
skoposti	skopost	Ncfpa	0.000004
skopostjo	skopost	Ncfsi	0.000037
skopostma	skopost	Ncfdd	0.000000
skopostma	skopost	Ncfdi	0.000000
skopostmi	skopost	Ncfpi	0.000000
skopost	skopost	Ncfsn	0.000179
skopost	skopost	Ncfsa	0.000092

Slika 3: **Paradigme ene besede iz leksikona Sloleks v tabelaričnem formatu**

2.5 Učni korpus sodobne slovenščine ssj500k

Za oblikoskladenjsko označevanje potrebujemo učni korpus, za kar uporabimo korpus sodobne slovenščine ssj500k (Arhar, 2009). Korpus vsebuje 500.000 besednih pojavnic; vsaka je ročno označena z oblikoskladenjsko lastnostjo in lemo. Korpus je tudi delno označen s skladenjskimi analizami in imenskimi entitetami, vendar tu ne uporabljamo teh informacij. Zapis je podoben kot za korpus starejše slovenščine, vendar preprostejši, saj ne vsebuje posodabljanja besed.

3 PROGRAM ToTrTaLe

Program za jezikoslovno označevanje starejših besedil ToTrTaLe, katerega prva različica je predstavljena v Erjavec (2011), implementira cevovod, ki iz vhodnega dokumenta TEI izlušči besedilo, nato pa nad njim enega za drugim pokliče posamezne module za označevanje. Za osnovo mu služi program ToTaLe (Erjavec idr., 2005), ki razdeli besedilo na pojavnice (tokenizacija), te oblikoskladenjsko označi (tagiranje) in jim pripiše osnovno obliko (lematizacija). Program, ki ga predstavljamo, doda prepis starinskih oblik v sodobne (transkripcija) takoj za tokenizacijo in se zato imenuje ToTrTaLe. Program na izhod izpiše dokument TEI, v katerem so vhodnim oznakam TEI dodane jezikoslovne oznake, kot so bile prikazane na primeru ročno označenega korpusa goo300k na sliki 2; izhod iz programa na delčku besedila iz slike 1 je prikazan na sliki 3.

```
<div xml:id="wv-1._Dershi_ali_vmirajozha_.
C5.BFkopo.C5.BFt.">
   <head rend="centered italic">
      <s>
         <w lemma="1." ana="Mdo">1.</w>
         <c> </c>
         <choice>
            <orig><w>Dershi</w></orig>
            <reg><w lemma="držati"
            ana="Vmpr3s">drži</w></reg>
         </choice>
         <c> </c>
         <w lemma="ali" ana="Cc">ali</w>
         <c> </c>
         <choice>
            <orig><w>vmirajozha</w></orig>
            <reg type="pattern"
            n="[u←v+č←zh]">
               <w lemma="umirajoč"
               ana="Agpfsn">umirajoča</w>
            </reg>
         </choice>
         <c> </c>
         <choice>
            <orig><w>ſkopoſt</w></orig>
            <reg type="pattern"
            n="[s←ſ+s←ſ]">
               <w lemma=«skopost«
               ana=«Ncfsn«>skopost</w>
            </reg>
         </choice>
         <pc>.</pc>
      </s>
```

Slika 4: **Primer besedila, označenega s ToTrTaLe**

Program je v glavnem jezikovno neodvisen, saj uporablja zunanja pravila in modele, ki jih je mogoče napisati oz. se jih induktivno naučiti za večino evropskih jezikov, čeprav je mišljen predvsem za jezike z bogato morfologijo, kot je slovenščina. Program je napisan v programskem jeziku Perl, vendar je glavni program v resnici samo ovojnica, ki kliče druge programe in nato kombinira njihove rezultate. V nadaljevanju razdelka predstavimo posamezne module ToTrTaLe, pri čemer se najbolj posvetimo specifikam obdelave starejše slovenščine.

3.1 Tokenizacija

Za razdelitev besedila na stavke, besede, ločila in presledke uporabljamo večjezični tokenizator mlToken, ki je del paketa To(Tr)TaLe. Program jezikovno odvisne podatke hrani v ločenih datotekah, predvsem seznam okrajšav (besede, ki se končajo s piko in ne končajo nujno stavka), seznam večbesednih enot (pojavnice, ki so sestavljene iz več s presledki ločenih besed) in seznam levih ali desnih naslonk (besed, ki jih je treba obravnavati kot del neke pojavnice). V kontekstu posodabljanja starejše slovenščine sta posebno zanimiva seznama večbesednih enot in naslonk, saj se precej besed, ki so se včasih pisale skupaj, sedaj piše narazen oz. obratno, npr. »nemore« proti »ne more« oz. »še le« proti »šele«. Te besede so dodane v ustrezen seznam, tako da že mlToken poskrbi za njihovo tokenizacijo v skladu s sodobno normo. Potrebni seznami za tokenizacijo starejše slovenščine za ToTrTaLe niso napisani posebej za to orodje, pač pa so zajeti neposredno iz leksikona IMP.

Trenutni pristop k reševanju teh posebnih pojavnic ima dve slabosti.

- Tokenizator pozna samo tiste posebne pojavnice, ki so v leksikonu, in torej ne obravnava pravilno novih, neznanih okrajšav, večbesednih enot oz. naslonk. Problem je posebno opazen pri presežniku pridevnikov, ki so se včasih pisali narazen (npr. »nar večji«), saj bomo s leksikonom težko zajeli vse oblike vseh stopnjevanih pridevnikov.
- Kot pri vseh drugih jezikoslovnih analizah se tudi pri posebnih pojavnicah srečamo s problemom dvoumnosti, pri čemer je klasifikacija neke pojavnice ali kombinacije pojavnic odvisna od sobesedila, npr. »Vesoljni potop je *po tem* vso deželo potopil«, kjer mora biti sodobna oblika »potem«, in »To se vidi tudi *po tem*, da vse tuje bolj ceni«, kjer pa mora biti »po tem«. Da vsaj deloma rešimo ta problem, v leksikon vedno vključimo oba primera, torej ne samo, ko se starinski »po tem« piše sodobno »potem«, temveč tudi ko so piše »po tem«. V tokenizator nato dodamo posebne primere samo tam, kjer je njihova frekvenca višja od navadnih, torej nezdruženih oz. nerazdeljenih pojavnic.

3.3 Transkripcija

Transkripcija zgodovinskih besednih oblik v sodobne je ključni modul za procesiranje starejšega jezika. Pri posodabljanju besednih oblik so le-te najprej normalizirane, tj. zapisane z malimi črkami, odstranjena pa so tudi naglasna znamenja nad samoglasniki; naglase so namreč pogosto, a neenotno uporabljali predvsem v 19. stoletju, v sodobni normi pa jih skoraj ni zaslediti.

V procesu iskanja sodobne ustreznice program najprej išče normalizirano zgodovinsko besedno obliko v leksikonu IMP; če jo najde, je s tem našel tudi sodobno ustreznico, če ne, pa besedno obliko išče v Sloleksu. Če nobeden od leksikonov ne vsebuje iskane oblike, program njen sodobni zapis skuša najti s pomočjo t. i. transkripcijskih vzorcev.

Veliko sprememb v pisavi lahko namreč izrazimo v obliki pravil, ki podajo vzorec, v katerem se sodobna beseda razlikuje od zgodovinske, npr. »r → er« za pare kot je »brž → berž«, »srce → serce«, pri čemer je na levi sodobni in na desni zgodovinski zapis. Pri uporabljenem pristopu v leksikonu sodobnih oblik Sloleks skušamo najti tiste, ki jih je mogoče izpeljati iz zgodovinske oblike z uporabo enega ali več takih pravil. Ta pristop je tipičen za posodabljanje starejših besedil (Pilz idr., 2008; Gotscharek idr., 2009; Bennett idr., 2010; Sánchez-Marco idr., 2010), se pa pristopi razlikujejo v tehnologiji, ki jo uporabljajo za preverjanje ujemanja zgodovinske oblike s sodobnimi oblikami s pomočjo takšnih vzorcev.

V paketu ToTrTaLe ujemanje prek transkripcijskih vzorcev implementira knjižnica Vaam, Variant aware approximate matching (Gotscharek idr., 2009; Reffle, 2011), ki jih modelira kot (razširjene) končne avtomate, zaradi česar je prostorsko, predvsem pa časovno nezahtevna. Seznam sodobnih kandidatov, ki ga vrne za posamezno zgodovinsko besedo, je urejen glede na število vzorcev, ki jih je bilo treba uporabiti. Proces določanja sodobnih ustreznic je torej nedeterminističen, je pa v danem kontekstu seveda pravilna samo ena posodobitev. Trenutno modul za transkripcijo izbere tistega kandidata, ki ima v leksikonu Sloleks najvišjo frekvenco, vendar so mogoči tudi kompleksnejši modeli, ki bi odložili izbiro najboljšega kandidata, dokler nista opravljena še oblikoskladenjsko označevanje in lematizacija vseh (variant) pojavnic, saj bi s tem imeli več informacij za pravilno odločitev.

Za posodabljanje trenutno uporabljamo okoli sto vzorcev, ki smo jih določili s pomočjo ročno označenega korpusa goo300k; razdeljeni so na vzorce za starejša besedila v gajici (torej sodobni abecedi) in na vzorce za bohoričico. Razlog za dve množici ni samo

razlika v pisavah, temveč so v besedilih izpred leta 1850 vzorci pogosto drugačni.

3.4 Oblikoskladenjsko označevanje

V naslednji stopnji označevanja program pripiše vsaki besedni pojavnici njeno (od konteksta odvisno) oblikoskladenjsko oznako JOS. Sodobni oblikoskladenjski označevalniki se modela jezika naučijo iz ročno označenega korpusa, vendar pa je razvoj dovolj velikega korpusa dolgotrajen in drag proces, ki bi ga težko ponovili za zgodovinski jezik. Ker so bile besedne oblike v predhodnem koraku posodobljene, lahko označevalniku kot vhod ponudimo posodobljeno besedilo in nato uporabimo model, naučen na sodobnem jeziku. Seveda model še vedno deluje slabše kot nad sodobnim jezikom, saj so zgodovinska besedila drugačna ne samo v pisavi posameznih besed, temveč tudi na skladenjski ravni, pa tudi nekatere besedne oblike, kot npr. deležja na -vši, so bila v preteklosti bistveno bolj pogosta, kot so danes.

Za oblikoskladenjsko označevanje uporabljamo program TnT, Tri-grams and tags (Brants, 2000), ki je robusten in hiter trigramski označevalnik, označevati pa zna tudi neznane besede, čeprav je tu natančnost manjša kot za znane. Model označevanja je bil naučen na učnem korpusu sodobne slovenščine ssj500k, pri čemer je kot zaledni leksikon uporabljen Sloleks.

3.5 Lematizacija

Zadnja stopnja jezikoslovne obdelave je pripis osnovne oblike vsaki besedni pojavnici. Kot pri oblikoslovnem označevanju se tudi pri tem večina sodobnih lematizatorjev nauči modela jezika iz vnaprej pripravljenih jezikovnih virov, v tem primeru iz leksikona sodobnih besednih oblik, v našem primeru Sloleksa. Seveda bi lahko leme besednih oblik, vsebovanih v leksikonu Sloleks, preprosto prepisali iz leksikona, vendar imajo lematizatorji to prednost, da znajo lematizirati tudi neznane besede. Če je beseda pravilno posodobljena in ji je pripisana pravilna oblikoskladenjska oznaka, deluje lematizator s precej visoko stopnjo natančnosti.

Kot lematizator uporabljamo CLOG (Erjavec in Džeroski, 2004), ki se na podlagi vhodnih primerov (parov besedna oblika – lema, pri čemer je model za vsako oblikoskladenjsko oznako obravnavan posebej) nauči odločitvene sezname prvega reda, pri čemer je definirana operacija povezovanje nizov. Naučene strukture so predikati v programskem jeziku Prolog, vendar jih za lažjo povezljivost s ToTrTaLe prevedemo v Perl.

Zanimiva lastnost lematizatorja CLOG je, da mu ne uspe lematizirati poljubnega para oblika – oblikoskladenjska oznaka. Pri starejših besedilih so taki primeri skoraj vedno zastarele besede, ki niso bile pravilno posodobljene, tako da so nelematizirane besede dobri kandidati za dodajanje v leksikon IMP.

3.6 Izhod TEI

Zadnja stopnja obdelave je zapis označenega besedila v dokument TEI, kar dosežemo s kombinacijo obdelave v jeziku Perl s skriptami XSLT, čemur sledi še validacija dobljenega dokumenta XML glede na shemo TEI, pri čemer je ta izražena v Relax NG (ISO/IEC 19757-2). Če pride pri validaciji do napak, je to indikator, da vhodni dokument krši (mogoče implicitne) predpostavke označevanja; v tem primeru je treba bodisi popraviti oznake v vhodnem dokumentu ali pa – če je bilo uporabljeno označevanje smiselno – dopolniti program ToTrTaLe, da bo zajel tudi takšne primere. Označevanje v dokumentih TEI je namreč lahko zelo kompleksno, zato je v splošnem težko zagotoviti, da vstavljanje novih (jezikoslovnih) oznak v tak dokument ne privede do nepravilnih struktur. Vendar je ToTrTaLe razmeroma robusten, saj označi vseh 658 del iz zbirke IMP tako, da je izhod pravilen TEI.

4 EVALVACIJA OZNAČEVANJA

V tem razdelku poskusimo odgovoriti na vprašanje, kako dobro ToTrTaLe posodablja, lematizira in oblikoskladenjsko označuje neznane besedne oblike glede na časovno obdobje, v katerem je nastalo besedilo.

Kot je bilo omenjeno v razdelku 2.3, je leksikon zgodovinskih besednih oblik IMP sestavljen iz:

1. vseh besednih oblik iz korpusa goo300k,
2. besednih oblik z ročno preverjenimi oznakami iz vzorca celotne zbirke besedil IMP; ta vzorec tu poimenujemo korpus IMPtest.

Za eksperiment smo programu ToTrTaLe dali na voljo samo prvi leksikon, drugi leksikon pa smo uporabili kot testno množico. Povedano bolj natančno, korpus IMPtest smo najprej razdelili v tri podkorpuse, vsakega za eno časovno obdobje, in sicer za drugo polovico 18. stoletja (18B), prvo polovico 19. stoletja (19A) in drugo polovico 19. stoletja (19B). Nato smo vsakega od podkorpusov označili s ToTrTaLe in

iz njih izločili leksikon ročno pregledanih besednih oblik, skupaj z njihovimi ročnimi ter avtomatskimi oznakami za posodobljeno obliko, lemo in oblikoskladenjsko oznako.

V tabeli 1 podamo nekaj kvantitativnih podatkov o tem testnem leksikonu. V tabeli posebej izpostavimo zgodovinske in sodobne oblike, pri čemer kot sodobne štejemo tiste, v katerih je besedna oblika iz besedila enaka kot sodobna, četudi s transliteracijo iz bohoričice v gajico, kot zgodovinske pa vse ostale. Tako kot sodobno štejemo npr. *bojiš* → *bojiš* kot tudi *bojiſh* → *bojiš*, za zgodovinsko pa npr. *boh* → *bog*. Za zgodovinske, sodobne in vse oblike podamo število vseh vnosov v leksikonu, število različnih besednih oblik, število različnih posodobljenih besed in število različnih lem.

Tabela 1: **Velikost testnega leksikona**

Obdobje	**Zgodovinske oblike**				**Sodobne oblike**				**Vse oblike**			
	Vnosov	Oblik	Poso.	Lem	Vnosov	Oblik	Poso.	Lem	Vnosov	Oblik	Poso.	Lem
18B	3.400	3.224	2.843	1.885	1.105	1.090	1.090	902	4.505	4.270	3.841	2.535
19A	3.484	3.366	3.168	2.228	3.385	3.326	3.298	2.483	6.820	6.572	6.245	4.166
19B	2.104	2.040	2.012	1.581	10.668	10.320	10.320	7.677	12.745	12.220	12.078	8.596
Σ	8.790	8.407	7.209	4.629	14.677	14.239	13.932	9.660	23.341	22.270	20.050	12.288

Kot je razvidno iz tabele, ima leksikon nekaj čez 23.000 vnosov oz. 22.000 besednih oblik, 20.000 posodobljenih oblik in 12.000 lem. Od tega je v 18B sodobnih okoli 25 odstotkov besednih oblik, v 19A jih je 50 odstotkov, v 19B pa 85 oz. 64 odstotkov, ne glede na časovno obdobje; tolikšna bi bila torej tudi natančnost identifikacije sodobnih besednih oblik sistema, ki ne bi opravljal posodabljanja.

V tabeli 2 podamo točnost ToTrTaLe z leksikonom goo300k nad leksikonom neznanih besednih oblik iz tabele 1. Točnost posodabljanja čez vsa obdobja je okoli 70 odstotkov, kar vključuje tako sodobne kot zgodovinske besede. Samo za zgodovinske je točnost pod 30 odstotki za besedne oblike in nekoliko večja za lematizacijo. Zanimivo je, da je točnost posodabljanja največja pri najstarejših besedilih, pri katerih je nekaj manj kot 35-odstotna. Obratno, kar je tudi pričakovano, pa točnost oblikoskladenjskega označevanja pada s starostjo besedil, od skoraj 70 pri 19B do 57 odstotkov pri 19B.

Za sodobne oblike morda preseneča, da je točnost »posodabljanja« manjša od sto odstotkov, za 18B je napaka celo štiriodstotna. Te napake so posledica dejstva, da sodobni leksikon Sloleks ne vsebuje posodobitev vseh besed, ki jih najdemo v testnem leksikonu – v takih primerih sistem poskusi posodobiti neznano (sodobno) besedo, pri čemer mu to v nekaterih primerih tudi uspe, vendar dobimo kot rezultat napačno obliko.

Tabela 2: **Točnost posodabljanja, lematizacije in oblikoskladenjskega označevanja testnega leksikona**

Obdobje	**Zgodovinske oblike**			**Sodobne oblike**			**Vse oblike**		
	Poso.	Lem.	Oblikoskl.	Poso.	Lem.	Oblikoskl.	Poso.	Lem.	Oblikoskl.
18B	34,7 %	38,5 %	56,8 %	96,2 %	87,7 %	79,3 %	49,8 %	50,6 %	62,3 %
19A	26,5 %	31,1 %	57,8 %	97,3 %	90,8 %	84,4 %	61,3 %	60,5 %	70,8 %
19B	24,2 %	32,2 %	68,6 %	99,3 %	93,0 %	85,1 %	86,9 %	82,9 %	82,4 %
Σ	28,8 %	33,9 %	59,9 %	98,6 %	92,0 %	84,3 %	72,4 %	70,2 %	75,1 %

Kot je razvidno iz rezultatov, je točnost sistema trenutno razmeroma slaba, vendar se je treba zavedati, da je posodabljanje kompleksen proces, pa tudi da sistem v praksi deluje bolje, kot nakazujejo številke. Predstavili smo namreč rezultate na neznanih besedah, ne na vseh, pri tem pa ima produkcijski ToTrTaLe na voljo ves leksikon, vključno s testnim, ki smo ga tu izločili, zaradi česar je njegova točnost na

vseh besedah bistveno boljša. Predstavljeni rezultati so slabši tudi zaradi tega, ker testni leksikon vsebuje besede, ki jih – vsaj trenutno – program ne more najti v Sloleksu, tj. zastarele besede, tujke in zatipkane besede, ki skupaj sestavljajo več kot deset odstotkov vnosov v testnem leksikonu.

5 SKLEP

V prispevku smo predstavili metodologijo, jezikovne vire in program za posodabljanje, lematizacijo in oblikoskladenjsko označevanje starejših besedil ter izvedli poskus, s katerim smo ocenili točnost programa na neznanih besedah. Rezultati kažejo, da je točnost mogoče še zelo povečati, kar lahko dosežemo na več načinov, ki ostajajo za nadaljnje delo. Najbolj preprosto (pa tudi najbolj zamudno oz. drago) bi bilo dodajati nove besede in njihove posodobitve v leksikon IMP, v katerem so nato neposredno dostopne. Zelo koristno, vendar prav tako zamudno, bi bilo dodajati nove besede tudi v leksikon sodobnih besed, saj analiza napak posodabljanja pokaže, da bi vzorci včasih pravilno predvideli sodobno obliko, a te ni Sloleksu. Ravno tako bi bilo dobro v leksikon sodobnih oblik dodati tudi (najpogostejše) tuje besede, predvsem v latinščini, nemščini in francoščini. Več dela bi lahko vložili tudi v transkripcijske vzorce, saj nismo pokrili vseh regularnih sprememb. Vendar se ob tem pojavi problem lažnih ustreznic, saj s preveč pravili hitro najdemo neko sodobno besedo za skoraj poljubno zgodovinsko obliko, zaradi česar je treba nove vzorce dodajati s sprotnim testiranjem njihovega učinka na večji testni množici.

Zadnja od možnosti za izboljšavo sistema bi bila uporaba povsem drugačnega načina posodabljanja, ki je že dalo spodbudne rezultate (Scherrer in Erjavec, 2013), pri katerem učno množico (leksikon iz goo300k) izkoristimo za učenje statističnega strojnega prevajanja na ravni posameznih črk v besedi. Princip strojnega prevajanja bi lahko razširili tudi na prevajanje celotnih besedil, pri čemer bi za učno množico potrebovali izvorno besedilo (ali besedilo, posodobljeno na ravni posameznih besed), ki je poravnano s »prevodom« tega besedila v sodobno slovenščino. S takim pristopom bi lahko zajeli tudi spremembe na skladenjski ravni, vendar je pri tem pristopu največja težava pridobivanje zadosti velike in splošne učne množice.

LITERATURA

[1] Arhar, Š. (2009). Učni korpus SSJ in leksikon besednih oblik za slovenščino. *Jezik in slovstvo, 54(3–4)*, str. 43–56. URL: http://www.jezikinslovstvo.com/pdf/2009-03-04-Razprave-Spela-Arhar.pdf.

[2] Bennett, P., Durrell, D., Scheible, S., Whitt, R. J. (2010). Annotating a historical corpus of German: A case study. *Proceedings of the LREC 2010 workshop on Language Resources and Language Technology Standards*. Valletta, Malta, 18 May 2010. str. 64–68.

[3] Erjavec, T. (2009). Odprtost jezikovnih virov za slovenščino. V: Infrastruktura slovenščine in slovenistike (Obdobja, Simpozij, = Symposium, 28). Ljubljana: Znanstvena založba Filozofske fakultete, str. 115–121. URL: http://www.centerslo.net/files/file/simpozij/simp28/Erjavec.pdf.

[4] Erjavec, T. (2011). Automatic linguistic annotation of historical language: ToTrTaLe and XIX century Slovene. V: LaTeCH 2011: The 5th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities. Portland, ZDA. Portland: Association for Computational Linguistics, str. 33–38. URL: http://aclweb.org/anthology-new/W/W11/W11-1505.pdf.

[5] Erjavec, T. (2012a). Jezikoslovni viri starejše slovenščine. *Knjižnica, 56(3)*, str. 205–221.

[6] Erjavec, T. (2012b). The goo300k corpus of historical Slovene. V: Eight International Conference on Language Resources and Evaluation (LREC'12), Istanbul. European Language Resources Association (ELRA). URL: http://www.lrec-conf.org/proceedings/lrec2012/summaries/445.html .

[7] Erjavec, T., Džeroski, S. (2004). Machine Learning of Language Structure: Lemmatising Unknown Slovene Words. *Applied Artificial Intelligence*, 18(1):17–41.

[8] Erjavec, T., Ignat, C., Pouliquen, B., Steinberger, R. (2005). Massive Multi-Lingual Corpus Compilation: Acquis Communautaire and ToTaLe. V: *Proceedings of the 2nd Language & Technology Conference*, April 21-23, 2005, Poznan, Poljska. str. 32–36.

[9] Erjavec, T. in Krek. S. (2008). Oblikoskladenjske specifikacije in označeni korpusi JOS. V: *Zbornik Šeste konference Jezikovne tehnologije*. Ljubljana, Inštitut Jožef Stefan. URL: http://nl.ijs.si/jos/bib/jos_isltc08.pdf.

[10] Gotscharek, A., Neumann, A., Reffle, U., Ringlstetter, C., Schulz, K. U. (2009). Enabling Information Retrieval on Historical Document Collections – the Role of Matching Procedures and Special Lexica. *Proceedings of the ACM SIGIR 2009 Workshop on Analytics for Noisy Unstructured Text Data (AND09),* Barcelona.

[11] Hladnik, M. (2009). Infrastruktura slovenistične literarne vede. V: *Obdobja 28 – Infrastruktura slovenščine in slovenistike*, str. 161–169. URL: http://www.centerslo.net/files/file/simpozij/simp28/Hladnik.pdf.

[12] Juršič, M., Mozetič, I., Erjavec, T., Lavrač, N. (2010) LemmaGen: multilingual lemmatisation with induced Ripple-Down rules. *Journal of universal computing science*. 16/9, str. 1190–1214.

[13] Krstulović, Z. in Šetinc, L. (2005). Digitalna knjižnica Slovenije – dLib.si. *Informatika kot temelj povezovanja: zbornik posvetovanja*, str. 683–689.

[14] Logar Berginc, N., Grčar, M., Brakus, M., Erjavec, T., Arhar Holdt, Š., Krek, S. (2012) *Korpusi slovenskega jezika Gigafida, KRES, ccGigafida in ccKRES : gradnja, vsebina, uporaba*. (Zbirka Sporazumevanje). Ljubljana: Trojina, zavod za uporabno slovenistiko: Fakulteta za družbene vede, 2012.

[15] Pilz, T. Ernst-Gerlach, A. Kempken, S., Rayson P., Archer, D. (2008). The Identification of Spelling Variants in English and German Historical Texts: Manual or Automatic? *Literary and Linguistic Computing*, 23/1, str. 65–72.
[16] Ogrin, M., Erjavec, T. (2009). Ekdotika in tehnologija: elektronske znanstvenokritične izdaje slovenskega slovstva. *Jezik in slovstvo, 54/6*, str. 57–72.
[17] Reffle, U. (2011). Efficiently generating correction suggestions for garbled tokens of historical language, Journal of Natural Language Engineering, Special Issue on Finite State Methods and Models in Natural Language Processing.
[18] Sánchez-Marco, C., Boleda, G., Maria Fontana, J., Domingo, J. (2010). Annotation and Representation of a Diachronic Corpus of Spanish. *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*. ELRA, Pariz.
[19] Scherrer, Y., Erjavec, T. (2013). Modernising historical Slovene words with character-based SMT. *Proceedings of the* ACL Workshop on *Balto-Slavic Natural Language Processing, BSNLP 2013. Sofija, Bulgarija.*
[20] Šorn, M. in Hadalin, J. (2010). Spletni portal SIstory: prost dostop do dosežkov slovenskega zgodovinopisja. *Zbornik prispevkov 4. skupnega posvetovanja Sekcije za specialne knjižnice in Sekcije za visokošolske knjižnice Zveze bibliotekarskih društev Slovenije*, Ljubljana, 27. in 28. oktober 2010, str. 103–107.
[21] TEI (2007). TEI P5: Guidelines for Electronic Text Encoding and Interchange. URL http://www.tei-c.org/Guidelines/P5/.

▪

Tomaž Erjavec je višji raziskovalni sodelavec na Odseku za tehnologije znanja na Institutu Jožef Stefan. Področja njegovega raziskovanja so jezikovne tehnologije in digitalna humanistika s poudarkom na izdelavi in označevanju ter predstavitvi jezikovnih virov slovenskega jezika. Na področjih jezikovnih tehnologij in korpusnega jezikoslovja je poučeval na univerzah v Novi Gorici in v Gradcu ter na mednarodni podiplomski šoli Jožefa Stefana. Je član uredniških odborov revij Journal for Language Resources and Evaluation, Journal of Corpus Linguistics in Slovenščina 2.0: empirične, aplikativne in interdisciplinarne raziskave; bil je ustanovni predsednik slovenskega Društva za jezikovne tehnologije, član svetov European Chapter of the Association for Computational Linguistics in Text Encoding Initiative Consortium ter sodeluje pri izdelavi standardov za zapis jezikovnih virov pri SIST in ISO TC 37.

Uporaba strojnega učenja za postavljanje vejic v slovenščini

Peter Holozan, Amebis, d. o. o., Kamnik, Bakovnik 3, 1241 Kamnik
peter.holozan@amebis.si

Izvleček

Za slovenščino obstajata dva programa, ki postavljata vejice v besedilo s pomočjo pravil, ni pa še bilo preizkušeno strojno učenje, ki je že bilo uspešno uporabljeno za postavljanje vejic v drugih jezikih. Za preizkušanje je bil uporabljen seznam primerov z napakami pri vejicah iz korpusa Šolar (209.156 besed). V prvem delu je bilo strojno učenje uporabljeno za problem iskanja vseh vejic, doseženi rezultat je primerljiv z drugimi jeziki (natančnost 0,861 in priklic 0,641) in s programoma s pravili, najboljši rezultat je bil dosežen z uporabo skladenjskega analizatorja, lematizator, oblikoslovni označevalnik in skladenjski analizator pa so bili naučeni z učno množico brez vejic, uporabljen je bil klasifikator ADTree. Preizkušena je bila še uspešnost popravljanja realnih napak v besedilu, pri čemer je bil rezultat slabši (natančnost 0,676 in priklic 0,545 za manjkajoče vejice).
Ključne besede: postavljanje vejic, popravljanje napačnih vejic, slovenščina, strojno učenje, ADTree.

Abstract

Using Machine Learning for Comma Placing in Slovene

For the Slovene language there currently exist two software solutions able to place commas into text using rules, however Machine Learning that has already been successfully used for comma placing in other languages has never been tried with Slovene. For testing, a list of examples with comma mistakes from the corpus Šolar (209156 words), was used. In the first part of the experiment machine learning was used for searching all commas, the obtained result is comparable with other languages (precision 0.861 and recall 0.641) and the rule-based programs. The best result was achieved using the syntax analyser. The lemmatiser, the PoS tagger and the syntax analyser were trained on a corpus without commas, the ADTree classifier was used. Real comma mistakes were also tested but the results were worse (precision 0.676 and recall 0.545 for missing commas).
Key words: comma placing, comma error correction, Slovene, machine learning, ADTree.

1 UVOD

Program, ki bi pravilno postavljal vejice v besedilo, ni uporaben le za pisce, ki tipkajo besedila in pri tem spregledajo kakšno vejico (postavljanje vejic povzroča hude težave celo bodočim učiteljem na razredni stopnji (Šek Mertük, 2011)), temveč tudi za druge namene. Pravilno postavljene vejice tako npr. izboljšajo oblikoslovno označevanje besedil (Hillard idr., 2006), pomembne pa so tudi pri sistemih za razpoznavo govora, ki le iz govora ne morejo pravilno postaviti vejic (Huang & Zweig, 2002).

Za slovenščino že obstajata dva programa (Besana[1] in LanguageTool[2]), ki postavljata manjkajoče vejice; oba temeljita na ročno napisanih pravilih (Holozan, 2012). Nihče pa za slovenščino še ni preizkusil, kako uspešne so pri tem statistične metode, ki uporabljajo strojno učenje iz primerov za izpeljavo pravil za vejice. Strojno učenje zahteva veliko število primerov, iz katerih lahko izpelje pravila; taki primeri napačne oz. pravilne rabe vejic so zdaj na voljo v korpusu Šolar, v katerem so zbrana besedila, ki so jih napisali učenci in dijaki, skupaj z učiteljskimi popravki.

2 PREDHODNE RAZISKAVE

Strojno učenje je bilo že večkrat uporabljeno za učenje postavljanja vejic v drugih jezikih, večinoma pa so raziskovali problem, ko je treba v besedilo postaviti vse vejice (oz. nekateri celo vsa ločila), kar je pomembno predvsem pri sistemih za razpoznavo govora (Huang & Zweig, 2002).

Beeferman idr. (1998) so preizkušali postavljanje vejic v angleščini s pomočjo skritega markovskega modela in z uporabo Viterbijevega algoritma.

Hardt (2001) je preizkušal postavljanje vejic v danščini, in sicer z uporabo Brillovega označevalnika, vendar se je omejil le na ugotavljanje odvečnih vejic, pri čemer so bile odvečne vejice dodane naključno.

[1] http://besana.amebis.si
[2] http://www.languagetool.org/

Zhang idr. (2002) so preizkušali strojno učenje za vejice v angleščini in nemščini, in sicer z odločitvenimi drevesi z uporabo skladenjskih podatkov.

Shieber in Tao (2003) sta preizkušala postavljanje vejic za angleščino; pomembna je njuna ugotovitev, da je smiselno naučiti statistični označevalnik na učnem korpusu brez vejic.

Alegria idr. (2006) so preizkušali strojno učenje v baskovščini. Uporabili so program WEKA[3] in preizkušali različne metode strojnega učenja.

Israel idr. (2012) so se ob problemu postavljanja vseh vejic v angleščini lotili tudi problema popravljanja napačnih (manjkajočih in odvečnih) vejic v besedilu.

Programa za postavljanje vejic v slovenščini je preizkusil Holozan (2012), in to za problem, ko je treba popraviti napačne vejice v besedilu. Uporabljen je bil vzorec, narejen iz korpusa Šolar, ki vsebuje napake, ki so jih naredili učenci osnovnih in srednjih šol.

3 ZASNOVA POSKUSA

Namen poskusa je preizkusiti metode strojnega učenja v slovenščini, in sicer najprej za problem postavljanja vseh vejic (na kar je bila osredinjena do zdaj večina tujih raziskav in kar je uporabno pri razpoznavi govora), potem pa še za problem popravljanja napačnih vejic (kar je uporabno v slovničnih pregledovalnikih, ki tako pomagajo piscem besedil postavljati vejice).

Osnova ideja poskusa postavljanja vseh vejic je povzeta po Alegria idr. (2006) in je taka, da uporabimo korpus s pravilno postavljenimi vejicami, ga oblikoskladenjsko označimo, lematiziramo in skladenjsko razčlenimo (pri čemer je treba upoštevati, da pri praktični uporabi nimamo vejic vnaprej, zato je treba preizkusiti označevanje tudi brez vejic, na kar sta opozorila že Shieber in Tao (2003), medtem ko Alegria idr. (2006) tega niso posebej preizkušali). Vsako besedo z določenim okoliškim oknom pretvorimo v seznam atributov in dodamo atribut, ali ji sledi vejica (ta atribut je potem razred pri klasifikacijskem problemu). Tako zapisane besede uvozimo v program za strojno učenje, v katerem izvedemo eksperimente.

Enako kot pri Alegria idr. (2006) je bil uporabljen program WEKA, ki ima vgrajeno veliko klasifikatorjev. Preizkušeno je bilo večje število klasifikatorjev, potem pa izbranih nekaj najboljših (pri čemer smo upoštevali, da so čim bolj različni), ki so bili potem uporabljeni v nadaljnjih preizkusih, v katerih so bili preizkušeni različni atributi, velikost okna, vpliv označevanja in parametri klasifikatorja.

Za preizkušanje je bilo uporabljeno desetkratno prečno preverjanje, pri čemer primere razdelimo na deset delov, devet delov uporabimo za učenje, preostali del pa za preizkušanje, kar ponovimo desetkrat z različnim delom za preizkušanje in izračunamo povprečni priklic in natančnost.

Za primerjavo sta bila na isti nalogi preizkušena še Besana in LanguageTool.

Drugi poskus je prenos ugotovitev iz prvega poskusa v popravljanje napačnih vejic in primerjava s programoma Besana in LanguageTool. Preizkušanje v tem poskusu je namreč bolj zapleteno, zato je najboljšo kombinacijo za strojno učenje laže poiskati pri problemu iskanja vseh vejic in jo potem uporabiti še pri popravljanju napačnih vejic.

3.1 Korpus

V raziskavi je bila uporabljena posodobljena verzija korpusa (popravljenih je bilo nekaj napačnih vejic), ki je bil uporabljen v Holozan (2012). To je podkorpus, narejen iz korpusa Šolar,[4] ki je zbirka besedil, ki so jih napisali učenci v šoli, in ki vključuje tudi popravke napak. Ta podkorpus vsebuje le povedi z napačnimi vejicami (bodisi manjkajočimi bodisi odvečnimi), pri čemer so mesta manjkajočih vejic označena z znakom ¤, odvečne vejice pa so nadomeščene z znakom ÷; velikost tega podkorpusa je 209.156 besed (vključno z ločili, razen vejic), v podkorpusu je 11.892 pravilno postavljenih vejic, 11.399 manjkajočih vejic in 2709 odvečnih vejic.

Za problem postavljanja vseh vejic (in tudi za učenje pri popravljanju vejic) je bil korpus predelan tako, da so bile vse vejice popravljene (znaki ¤ zamenjani z vejicami, znaki ÷ pa pobrisani), s čimer je bil narejen korpus s pravilno postavljenimi vejicami.

Predvsem za ta problem postavljanja vseh vejic (pa tudi za realno natančnost pri popravljanju napačnih vejic, čeprav je tu težava, da je ta odvisna od deleža napak v korpusu in se je tako težko odločiti, katera besedila vsebujejo povprečno število napačnih vejic) bi bilo sicer bolje uporabiti korpus, ki bi vseboval tudi povedi s pravilno postavljenimi vejicami, vendar takega korpusa ob izvajanju poskusa ni bilo

[3] http://www.cs.waikato.ac.nz/ml/weka/

[4] http://www.slovenscina.eu/korpusi/solar

na voljo. Tudi popravki v korpusu Šolar namreč niso povsem natančni, zato so bili primeri v podkorpusu ročno preverjeni in ustrezno popravljeni.

Druga možnost za postavljanje vseh vejic bi bila uporaba dela katerega od obstoječih korpusov (npr. Gigafide),[5] vendar se tu postavi vprašanje, kako natančno so lektorirana besedila, vključena v korpuse. Se je pa za to rešitev odločila večina tujih raziskovalcev (tudi Alegria idr. (2006), ki so med drugim uporabili časopisna besedila).

3.1.1 Označevanje

Tako Hardt (2001) kot tudi Alegria idr. (2006) so eksperimentirali z označenimi korpusi, saj lahko pravilne oblikoskladenjske oznake in poznavanje strukture povedi pomagajo pri postavljanju vejic.

Zato je bilo tudi za slovenščino uporabljeno označevanje, in sicer oblikoslovni označevalnik in lematizator Obeliks[6] ter skladenjski razčlenjevalnik,[7] ki sta bila razvita v okviru projekta Sporazumevanje v slovenskem jeziku.[8]

Pri poskusih za baskovščino in danščino ni posebej specificirano, ali so označevali korpus s pravilno ali z napačno postavljenimi vejicami, zdi se, da so uporabili različico s pravilno postavljenimi vejicami. Ker pa pravilnost vejic lahko vpliva na natančnost označevalnika (Hillard idr., 2006) in ker pri praktični uporabi (npr. popravljanju napačnih vejic v besedilu) ni mogoče vnaprej imeti pravilno postavljenih vejic, sta bili preizkušeni obe različici označevanja.

3.2 Ocenjevanje rezultatov

Za ocenjevanje rezultatov sta bili uporabljeni metriki natančnost (delež pravilno postavljenih vejic) in priklic (delež odkritih manjkajočih vejic) ter metrika F1, ki je harmonična sredina natančnosti in priklica in se izračuna kot 2 * natančnost * priklic / (natančnost + priklic). Problem postavljanja vejic predstavimo z razredom, ki pove, ali neki besedi sledi vejica. V korpusu je 23.291 mest, kjer mora biti vejica, vejica torej mora biti za 11,1 odstotka besed, večinski razred pa je, da besedi ne sledi vejica, kar je v 88,9 odstotka primerov.

Program WEKA je rezultate izračunal tako za primer, ko ni vejice, kot za primere, ko vejica je. Ker je cilj postaviti vejice v besedilo, je zanimiv predvsem rezultat pri primerih, ko vejica je, saj nam to pove, koliko manjkajočih vejic bi odkrila metoda. Natančnost je pomembnejša od priklica, ker npr. pri slovničnem pregledovalniku nočemo preveč lažnih opozoril, seveda pa tudi priklic ne sme biti premajhen (npr. vsaj 50 %), da je metoda uporabna, zato je pomemben tudi rezultat za F1, ki ga prav tako izračunava program WEKA.

Rezultati so izračunani na besede, ker je beseda (z okoliškim oknom) element pri strojnem učenju.

Referenčna vrednost uspešnosti je rezultat, ki ga dosežeta programa, ki postavljata vejice s pomočjo pravil. Programa sicer nista namenjena za reševanje problema, ko je treba postaviti vse vejice, vendar je vseeno zanimivo videti, kako dobro poiščeta vse vejice.

3.3 Priprava podatkov

Program WEKA potrebuje podatke v formatu ARFF, v katerem glavi z opisom atributov sledi podatkovni del, v katerem vsaka vrstica predstavlja en primer. Rezultat označevanja besedil je v formatu XML-TEI,[9] zato je bil napisan za pretvorbo program v Perlu. Ta za vsako besedo določi atribute, potem pa pri izvozu v ARFF ob sami besedi izpiše še atribute za prejšnje in naslednje besede glede na nastavitev okna (privzeta vrednost je –5 +5, torej pet besed spredaj in pet besed zadaj, s čimer so začeli tudi Alegria idr. (2006)). Vejice niso besede, ampak le atribut *je-vejica* na besedi neposredno pred vejico. Ta atribut je potem uporabljen kot razred pri strojnem učenju.

Program za izvoz v ARFF izvozi vse atribute (razen podatka o obstoju vejice) kot nize, s čimer pa večina klasifikatorjev ne zna delati, zato jih je treba najprej spremeniti v nominalne atribute, pri čemer je pri definiciji atributa našteta zaloga možnih vrednosti. V ta namen je bil v programu WEKA uporabljen filter StringToNominal.

3.3.1 Atributi

Osnovni atributi za vsako besedo so oblika (sama beseda, taka kot je napisana, npr. mize), lema (osnovna oblika besede, npr. miza) in oblikoskladenjska oznaka (ali MSD – morpho-syntactic descriptor, npr. Sozer) po oblikoskladenjskih specifikacijah JOS,[10] ki pove besedno vrsto, podatke o sklonu, spolu, številu

[5] http://www.gigafida.net
[6] http://www.slovenscina.eu/tehnologije/oznacevalnik
[7] http://www.slovenscina.eu/tehnologije/razclenjevalnik
[8] http://www.slovenscina.eu
[9] http://www.tei-c.org/Guidelines/P5/
[10] http://nl.ijs.si/jos/msd/html-sl/index.html

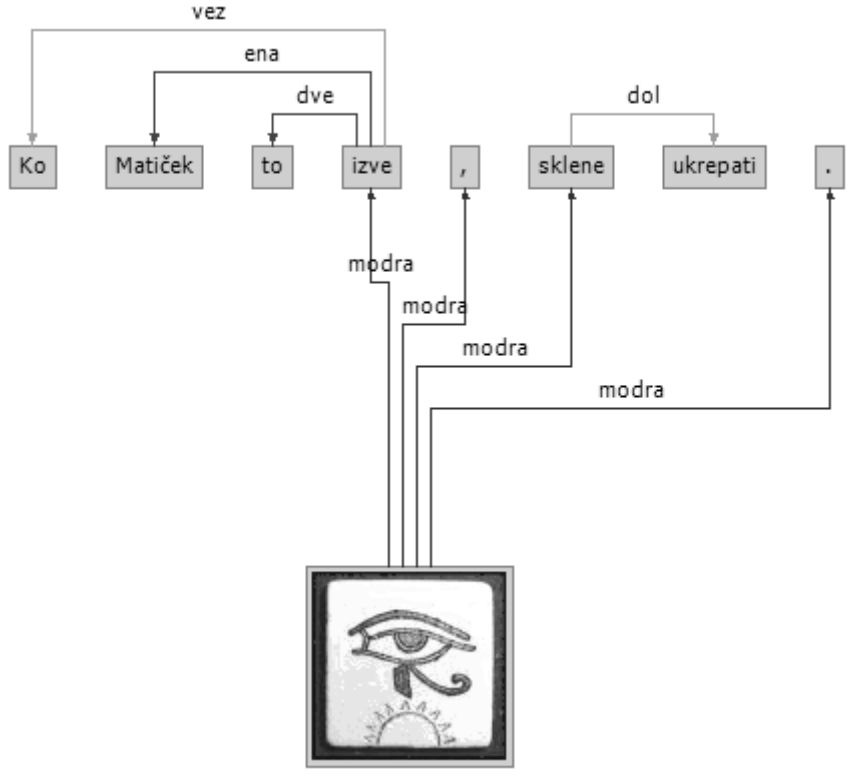
ipd. Ker ločila nimajo oblikoskladenjskih oznak, jim je bila pripisana oznaka Y, da jih lahko obravnavamo enako kot besede. Neobstoječim besedam znotraj okna so bili vsi atributi nastavljeni na *, vsak stavek je enota zase in okno ne sega na sosednje stavke.

Atributi so našteti tako, da so najprej atributi za samo besedo (položaj 0), temu sledijo atributi za predhodne besede (od -1 do -5) in temu atributi za naslednje besede (od +1 do +5).

Po celotnem MSD je bil narejen še poskus z delnim MSD, v katerem so atributi ločeno prvi znak MSD, drugi znak MSD in pri samostalnikih še sklon.

Delni MSD2 je bil poskus, kako čim bolj prenesti informacije iz MSD in se izogniti uporabi celotnega MSD (zaradi predpostavke, da veliko število različnih MSD lahko ovira učenje). Vsak MSD bil razdeljen v dva atributa, prvi je kot prvo črko vseboval besedno vrsto, druga črka pa je bila vrsta pri posamezni besedni vrsti (pri samostalnikih, pridevnikih, glagolih, zaimkih, številnikih in veznikih). Drugi atribut je vseboval sklon pri samostalnikih, pridevnikih, zaimkih, predlogih in številnikih, sicer pa **.

Naslednji poskus je bil uporaba podatkov skladenjskega razčlenjevalnika, pri katerem pa je rezultate težje pretvoriti v atribute kot pri oblikoslovnem označevalniku in lematizatorju, saj so rezultat skladenjskega razčlenjevalnika povezave, ki gradijo drevo.



Slika 1: **Rezultat skladenjskega razčlenjevalnika**

Slika 1 kaže rezultat skladenjske razčlembe za poved »Ko Matiček to izve, sklene ukrepati«. Za postavljanje vejic so pomembne predvsem povezave[11] »vez«, ki kaže na veznike, »modra«, ki kaže na del povedi, in rdeče povezave »ena«, »dve«, »tri« in »štiri«, ki kažejo na osebke, predmete in prislovna določila, pri čemer nas pri modrih in rdečih povezavah zanima začetek bloka, zato mora upoštevati še vse naslednje povezave, da pridemo do začetka tega bloka.

Rezultat razčlenjevalnika (skupaj z rezultatom lematizatorja in oblikoskladenjskega analizatorja) je zapisan v formatu XML, kot prikazujemo na sliki 2 (izpuščene so značke »<S />«, ki označujejo presledke).

[11] Vsi tipi povezav so opisani na http://www.slovenscina.eu/tehnologije/razclenjevalnik.

```
<s xml:id="0.0">
    <w lemma="ko" msd="Vd" xml:id="0.0.1">Ko</w>
    <w lemma="Matiček" msd="Slmei" xml:id="0.0.2">Matiček</w>
    <w lemma="ta" msd="Zk-set" xml:id="0.0.3">to</w>
    <w lemma="izvedeti" msd="Ggdste" xml:id="0.0.4">izve</w>
    <c xml:id="0.0.5">,</c>
    <w lemma="skleniti" msd="Ggdste" xml:id="0.0.6">sklene</w>
    <w lemma="ukrepati" msd="Ggnn" xml:id="0.0.7">ukrepati</w>
    <c xml:id="0.0.8">.</c>
    <links>
        <link afun="vez" dep="0.0.1" from="0.0.4" />
        <link afun="ena" dep="0.0.2" from="0.0.4" />
        <link afun="dve" dep="0.0.3" from="0.0.4" />
        <link afun="modra" dep="0.0.4" from="0.0.0" />
        <link afun="modra" dep="0.0.5" from="0.0.0" />
        <link afun="modra" dep="0.0.6" from="0.0.0" />
        <link afun="dol" dep="0.0.7" from="0.0.6" />
        <link afun="modra" dep="0.0.8" from="0.0.0" />
    </links>
    </s>
```

Slika 2: **Zapis označevanja in skladenjske razčlembe v formatu XML**

Slika 2 je primer, zapisan v formatu XML, ki je rezultat označevanja in skladenjskega razčlenjevalnika. Značke "<s" so povedi, značke "<w" besede, značke "<c" ločila in značke "<link" skladenjske povezave.

Za skladenjske atribute so bili izbrani (vrednost je 1, če je trditev resnična, oz. 0, če ni):

- je vez: beseda, na katero kaže povezava "vez";
- začetek modrega bloka: prva beseda v bloku, na katerega kaže povezava "modra";
- začetek rdečega bloka: prva beseda v bloku, na katerega kaže rdeča povezava.

```
'Ko','ko','Vd','1','0','1','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','Mati
ček','Matiček','Slmei','0','1','0','to','ta','Zk set','0','1','0','izve','izvedeti','Ggdste','0','0','0','sklene','skle
niti','Ggdste','0','0','0','ukrepati','ukrepati','Ggnn','0','0','0',ni vejice 'Matiček','Matiček','Slmei','0','1',
'0','Ko','ko','Vd','1','0','1','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','to','ta','Zk set','
0','1','0','izve','izvedeti','Ggdste','0','0','0','sklene','skleniti','Ggdste','0','0','0','ukrepati','ukrepati','Gg
nn','0','0','0','.','.','Y','1','0','0',ni vejice 'to','ta','Zk set','0','1','0','Matiček','Matiček','Slmei','0','1','0','Ko','k
o','Vd','1','0','1','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','*','izve','izvedeti','Ggdste','0','0','0','sklene','
skleniti','Ggdste','0','0','0','ukrepati','ukrepati','Ggnn','0','0','0','.','.','Y','1','0','0','*','*','*','*','*','*',ni vejice
'izve','izvedeti','Ggdste','0','0','0','to','ta','Zk set','0','1','0','Matiček','Matiček','Slmei','0','1','0','Ko','ko','V
d','1','0','1','*','*','*','*','*','*','*','*','*','*','*','*','sklene','skleniti','Ggdste','0','0','0','ukrepati','ukrepati','Ggnn'
,'0','0','0','.','.','Y','1','0','0','*','*','*','*','*','*','*','*','*','*','*','*',je vejica
```

Slika 3: **Zapis začetka zgornjega primera v formatu ARFF z oknom –5+5**

Slika 3 kaže, kako je začetek zgornjega primera zapisan v formatu ARFF, ki ga zna brati programski paket WEKA.

4 PREIZKUŠANJE

Za problem, ko je treba postaviti vse vejice besedilu, je bilo narejenih več preizkusov, da bi našli najboljšo kombinacijo klasifikatorja, atributov, velikosti okna, načina označevanja in parametrov klasifikatorja.

Zaradi velikega števila možnih kombinacij ni bilo mogoče preizkusiti vseh, ampak se je po posameznih delnih preizkusih ožil izbor (na podlagi natančnosti in delno tudi F1 na mestih, kjer so vejice), katere kombinacije je najbolj smiselno preizkušati naprej.

4.1 Izbiranje klasifikatorja in vpliv velikosti korpusa

Preizkušeno je bilo večje število klasifikatorjev, ki jih podpira program WEKA, vsi so bili uporabljeni s privzetimi parametri.

Tabela 1: **Šolar, celotni MSD, brez skladenjskih atributov**

		Ni vejice			Je vejica		
	Klasifikator	**Natančnost**	**Priklic**	**F1**	**Natančnost**	**Priklic**	**F1**
100 %	ZeroR	0,889	1	0,941	0	0	0
	HyperPipes	0,892	0,989	0,938	0,340	0,045	0,079
	J48						
	NaiveBayes	0,965	0,947	0,956	0,632	0,726	**0,676**
	Decision Table	0,948	0,986	0,966	**0,830**	0,565	0,672
	BayesNet	0,973	0,918	0,945	0,549	0,797	0,65
	Stacking	0,889	1	0,941	0	0	0
	VFI	0,919	0,928	0,923	0,347	0,345	0,359
	ADTree	0,945	0,977	0,961	0,751	0,546	0,632
	RBFNetwork	0,948	0,975	0,961	0,740	0,570	0,644
	AdaBoostM1	0,928	0,985	0,956	0,768	0,386	0,514
	NaiveBayesUpdateable	0,965	0,947	0,956	0,632	0,726	0,676
	DecisionStump	0,928	0,985	0,956	0,768	0,386	0,514
50 %	ADTree	0,943	0,979	0,961	0,761	0,533	0,627
	DecisionStump	0,927	0,985	0,955	0,766	0,384	0,511
25 %	J48	0,89	1	0,942	0	0	0
	NaiveBayes	0,925	0,992	0,958	0,848	0,351	0,497
	Decision Table	0,948	0,984	0,966	0,817	0,563	**0,666**
	Stacking	0,89	1	0,942	0	0	0
	ADTree	0,944	0,978	0,961	0,746	0,531	0,620
	LWL	0,931	0,986	0,958	0,78	0,409	0,537
	RBFNetwork	0,914	0,995	0,953	**0,854**	0,245	0,381
	AdaBoostM1	0,929	0,986	0,956	0,773	0,389	0,517
	NaiveBayesUpdateable	0,925	0,992	0,958	0,848	0,351	0,497
	DecisionStump	0,929	0,986	0,956	0,773	0,389	0,517

Preizkušeno je bilo še več klasifikatorjev, pri katerih pa izračunavanje bodisi ni uspelo (SMO, LibSVM, HNB, MultilayerPerceptron, Bagging, FT, Prism, J48) bodisi je trajalo predolgo (LWL, KStar, Id3, NBTree, BFTree, LADTree, SimpleCart, REPTree). Je pa seveda mogoče, da bi se dala katera od teh metod usposobiti z ustreznimi parametri klasifikatorja, ustrezno zmanjšanim oknom, manjšim korpusom ali več potrpljenja (počakati nekaj dni na rezultat).

Če želimo iskati manjkajoče vejice, nas zanima predvsem natančnost pri možnosti, ko vejica je, vendar seveda tudi priklic ne sme biti preslab, tako da iščemo tudi dober F1.

Kot uspešni klasifikatorji so se pokazali Decision Table, NaiveBayes, ADTree in RBFNetwork. Slaba

stran klasifikatorja Decision Table pa je, da je preizkušanje neuporabno počasno, zato je bil pri nadaljnjem preizkušanju namesto njega uporabljen AdaBoostM1 (klasifikatorji za nadaljnje preizkušanje so bili namerno izbrani tako, da pripadajo različnim skupinam klasifikatorjem in niso preveč podobni med seboj).

Klasifikatorji, ki niso bili uspešni na celotnem korpusu, so bili preizkušeni še na zmanjšanem korpusu, da bi morda bili uspešni tam (nekateri klasifikatorji pa so bili ponovljeni za primerjavo, koliko vpliva velikost korpusa).

Klasifikator J48, ki je bil uporabljen v Alegria idr. (2006), se je uspešno izvedel le pri 25 odstotkih primerov (vendar je tudi tu uporabil le večinski razred in je dal povsod odgovor, da ni vejice), pri 50 odstotkih in polnem korpusu preizkus ni bil uspešen. Klasifikator SMO pa sploh ni bil uspešen niti pri 25 odstotkih. Ta rezultat je presenetljiv, Alegria idr. (2006) so uporabljali korpus s 130.000 besedami za preizkuse (100.000 besed za učenje in 30.000 za preizkušanje) in malo manjše okno (–5+2), kar pomeni, da 25 odstotkov korpusa v našem poskusu ne bi smelo pomeniti težave. Zato bi bilo smiselno to še enkrat preizkusiti v prihodnosti z ustrezno nastavitvijo parametrov klasifikatorjev.

Manjšanje korpusa je poslabšalo rezultate pri klasifikatorjih NaiveBayes in RBFNetwork, na klasifikatorje Decision Table, ADTree in AdaBoostM1 pa ni bistveno vplivalo.

4.2 Atributi

Vprašanje je, kateri podatki so pomembni, da jih dodamo kot atribute. Osnovna podatka sta sama beseda in lema besede, narejen pa je bil poskus, kako uporabiti oblikoskladenjske oznake (MSD).

Tabela 2: **Šolar**

		Ni vejice			Je vejica		
	Klasifikator	**Natančnost**	**Priklic**	**F1**	**Natančnost**	**Priklic**	**F1**
Celotni MSD	NaiveBayes	0,965	0,947	0,956	0,632	0,726	0,676
	RBFNetwork	0,948	0,975	0,961	0,740	0,57	0,644
	ADTree	0,945	0,977	0,961	0,751	0,546	0,632
	AdaBoostM1	0,928	0,985	0,956	0,768	0,386	0,514
Delni MSD	NaiveBayes	0,971	0,924	0,947	0,563	0,781	0,654
	RBFNetwork	0,958	0,946	0,952	0,607	0,667	0,636
	ADTree	0,944	0,984	0,964	0,811	0,537	0,646
	AdaBoostM1	0,943	0,968	0,955	0,677	0,53	0,595
Brez oblik	NaiveBayes	0,975	0,904	0,938	0,515	0,812	0,630
	RBFNetwork	0,957	0,943	0,950	0,593	0,662	0,626
	ADTree	0,944	0,984	0,964	0,811	0,537	0,646
	AdaBoostM1	0,943	0,968	0,955	0,677	0,53	0,595
Delni MSD2	NaiveBayes	0,967	0,935	0,951	0,592	0,749	0,661
	RBFNetwork	0,953	0,958	0,955	0,648	0,620	0,634
	ADTree	0,930	0,989	0,959	**0,827**	0,402	0,541
	AdaBoostM1	0,928	0,985	0,956	0,768	0,386	0,514
MSD + delni MSD2	NaiveBayes	0,972	0,925	0,948	0,568	0,784	0,658
	RBFNetwork	0,960	0,949	0,954	0,625	0,683	0,653
	ADTree	0,930	0,989	0,959	0,827	0,402	0,541
	AdaBoostM1	0,928	0,985	0,956	0,768	0,386	0,514
MSD + skladnja	NaiveBayes	0,973	0,920	0,946	0,555	0,793	0,653
	RBFNetwork	0,956	0,949	0,953	0,616	0,652	0,634
	ADTree	0,950	0,983	0,966	0,815	0,588	**0,683**
	AdaBoostM1	0,950	0,964	0,957	0,675	0,594	0,632

Delni MSD (ločeno prvi znak MSD, drugi znak MSD in pri samostalnikih še sklon), je malce izboljšal rezultate pri klasifikatorjih ADTree in AdaBoostM1, poslabšal pa pri NaiveBayes in RBFNetwork.

Zanimiv rezultat je prinesla ukinitev atributov z oblikami (torej so ostale le leme), pri čemer je bil rezultat pri ADTree in AdaBoostM1 popolnoma enak, pri NaiveBayes in RBFNetwork pa se je poslabšal.

Delni MSD2 je bil poskus, kako čim bolj prenesti informacije iz MSD in se izogniti uporabi celotnega MSD (zaradi predpostavke, da veliko število različnih MSD lahko ovira učenje). Vendar je tudi ta poskus samo poslabšal rezultate (je sicer izboljšal natančnost pri ADTree, vendar za ceno velikega poslabšanja priklica) (rezultat je poslabšal celo delni MSD 2 in dodani celotni MSD), tako da je očitno najbolj smiselno uporabiti kar celotni MSD.

Atributi s podatki o skladnji so sicer poslabšali rezultat pri klasifikatorjih NaiveBayes in RBFNetwork, vendar so ga popravili pri ADTree in AdaBoostM1, in to toliko, da je F1 pri ADTree postal najboljši, zato je bila za nadaljnje poskuse izbrana ta kombinacija.

4.3 Velikost okna

Preizkušen je bil vpliv velikosti okna, tj. števila besed pred besedo, za katero ugotavljamo, ali ji sledi vejica, in za njo.

Tabela 3: **Šolar, ADTree, MSD + skladnja**

	Ni vejice			Je vejica		
Okno	**Natančnost**	**Priklic**	**F1**	**Natančnost**	**Priklic**	**F1**
–5+5	0,950	0,983	0,966	0,815	0,588	0,683
–4+5	0,950	0,983	0,966	0,815	0,588	0,683
–3+5	0,950	0,983	0,966	0,815	0,588	0,683
–2+5	0,950	0,983	0,966	0,815	0,588	0,683
–1+5	0,950	0,983	0,966	0,815	0,588	0,683
–0+5	0,950	0,983	0,966	0,815	0,588	0,683
–5+2	0,950	0,983	0,966	0,815	0,588	0,683
–5+1	0,950	0,984	0,966	0,818	0,582	0,680
–5+0	0,889	1,000	0,941	0,000	0,000	0,000
–0+2	0,950	0,983	0,966	0,815	0,588	0,683

Tabela 3 kaže, da klasifikator ADTree uporablja le trenutno besedo in še dve naprej. Vendar razen na hitrost večanje okna ne vpliva negativno na rezultat, zato je pri nadaljnjih preizkusih uporabljeno kar okno –5+5, tudi zaradi domneve, da pri spreminjanju parametrov klasifikatorja ADTree (torej večanjem drevesa) začne klasifikator upoštevati tudi besede zunaj okna –0+2, ki se je pokazalo kot zadostno tukaj (drevo, ki je rezultat poskusa s parametrom –B 50, res vsebuje tudi položaje +3, –1 in –2 in celo –5, torej bi bilo tam optimalno drevo –5+3, kar potrjuje to domnevo). Ta domneva je tudi razlog, da za nadaljnje preizkušanje nismo uporabili okna –5+1, ki je sicer malenkostno izboljšalo natančnost.

Mogoče vpliva na druge klasifikatorje velikost okna drugače, tako da bi bilo smiselno izvesti poskuse še za druge klasifikatorje, prav tako pa tudi za druge parametre klasifikatorja ADTree.

4.4 Vpliv označevanja

Rezultati postavljanja vejic so zelo uspešni, vendar vsebujejo problematično predpostavko: pri oblikoslovnem označevanju in skladenjski razčlembi je bilo uporabljeno besedilo, ki je vsebovalo pravilno postavljene vejice. To pa seveda ni realna situacija, saj v primeru, da hočemo v neko besedilo postaviti vejice, tega vnaprej seveda ne vemo.

Zato je bil naslednji poskus ugotoviti, kaj se zgodi, če oblikoslovni označevalnik in skladenjski razčlenjevalnik nimata vejic v vhodnem besedilu. Iz korpusa so bile izbrisane vse vejice in korpus je bil ponovno označen in pretvorjen v format ARFF. Ker pa je bil seveda povsod podatek, da ni vejice, je bilo treba iz datoteke ARFF za korpus z vejicami prenesti stolpec s podatki za vejico v datoteko ARFF korpusa brez vejic. Pri tem postopku je potrebna previdnost: nujno je treba preveriti, da se ujema število besed in se besede

pokrivajo. Nekateri tipi napak v izvornem korpusu namreč naredijo težave pri brisanju vejic, tak primer je npr. manjkajoč presledek za vejico, pri čemer brisanje vejice potem zlepi besedi in povzroči, da je v korpusu brez vejic ena beseda manj. Težava je tudi, da tokenizator (rezalnik na besede) včasih spreminja vezavo pike na predhodno besedo različno (npr. pri arabskem zapisu vrstilnih številnikov), če je blizu vejica. Te primere je bilo treba v označenem XML potem popraviti ročno, da so se besede ujemale.

Tabela 4: **Šolar, MSD + skladnja, –5+5**

		Ni vejice			Je vejica		
	Klasifikator	**Natančnost**	**Priklic**	**F1**	**Natančnost**	**Priklic**	**F1**
Označeno z vejicami	NaiveBayes	0,973	0,920	0,946	0,555	0,793	0,653
	RBFNetwork	0,956	0,949	0,953	0,616	0,652	0,634
	ADTree	0,950	0,983	0,966	0,815	0,588	0,683
	AdaBoostM1	0,950	0,964	0,957	0,675	0,594	0,632
Označeno brez vejic	NaiveBayes	0,971	0,916	0,943	0,538	0,783	**0,638**
	RBFNetwork	0,955	0,943	0,949	0,588	0,647	0,616
	ADTree	0,943	0,982	0,962	**0,787**	0,526	0,630
	AdaBoostM1	0,940	0,957	0,948	0,595	0,510	0,550
Označevalnik, naučen brez vejic	NaiveBayes	0,971	0,917	0,943	0,542	0,785	0,641
	RBFNetwork	0,954	0,947	0,951	0,601	0,639	0,619
	ADTree	0,947	0,982	0,964	**0,794**	0,563	**0,659**
	AdaBoostM1	0,947	0,976	0,961	0,745	0,566	0,643
	DecisionTable	0,954	0,989	0,971	**0,873**	0,617	**0,723**

Tabela 4 pove, da so se rezultati ugotavljanja vejic v primeru, ko besedilo pri označevanju ni imelo vejic, poslabšali (čeprav ne zelo izrazito, največja razlika je bila pri klasifikatorju AdaBoostM1), kar se sklada tudi s splošnimi ugotovitvami Hillarda idr. (2006), da pravilno postavljene vejice izboljšajo oblikoslovno označevanje besedil.

Preizkušeno pa je bilo še, ali lahko označevanje (in s tem posledično določanje vejic) izboljšamo s tem, da lematizator, oblikoslovni označevalnik in skladenjski razčlenjevalnik naučimo iz učnega korpusa brez vejic (to sta uporabila že Shieber in Tao (2003)). V ta namen so bile v učnem korpusu SSJ500k izbrisane vse vejice (in povezave na vejice pri skladenjski razčlenitvi) in na novo naučeni modeli za lematizator, oblikoslovni označevalnik in skladenjski razčlenjevalnik (ta postopek predvsem za oblikoslovni označevalnik porabi veliko procesorskega časa (dobrih 20 ur), vendar ga je treba narediti le enkrat). Rezultati so se izboljšali, niso pa dosegli primera, ko je bilo besedilo označeno z vejicami, kar kaže na to, da so vejice pomembne za razdvoumljanje. Vseeno pa se je pokazalo, da je v primeru, ko je treba v besedilu dodati vse vejice, smiselno naučiti označevalnike z učnim korpusom brez vejic.

Tukaj je bil dodatno preizkušen še klasifikator DecisionTable, ki je bil pri izbiranju klasifikatorjev zelo uspešen, vendar ni bil izbran za nadaljnje preizkušanje zaradi dolgotrajnosti preizkušanja.

4.5 Parametri klasifikatorja

Klasifikator DecisionTable je sicer dosegel najboljši rezultat, vendar je posamezni poskus trajal tri dni. Zato je bilo pri drugouvrščenem klasifikatorju ADTree (alternirajoče odločitveno drevo), ki je bil občutno hitrejši, preizkušeno, kako vplivajo nanj parametri.

Tabela 6: **Šolar, ADTree, MSD + skladnja, −5+5, označeno brez vejic**

	Ni vejice			Je vejica		
Parametri	**Natančnost**	**Priklic**	**F1**	**Natančnost**	**Priklic**	**F1**
-B 10 -E -3	0,943	0,982	0,962	0,787	0,526	0,630
-B 8 -E -3	0,943	0,981	0,962	0,779	0,524	0,627
-B 6 -E -3	0,943	0,981	0,962	0,779	0,524	0,626
-B 4 -E -3	0,939	0,983	0,960	0,779	0,490	0,602
-B 2 -E -3	0,940	0,978	0,958	0,735	0,499	0,549
-B 1 -E -3	0,940	0,948	0,944	0,553	0,515	0,533
-B 12 -E -3	0,944	0,982	0,962	0,785	0,534	0,635
-B 15 -E -3	0,946	0,982	0,964	0,796	0,555	0,654
-B 20 -E -3	0,949	0,984	0,966	0,819	0,578	0,678
-B 30 -E -3	0,949	0,989	0,969	0,868	0,580	0,695
-B 50 -E -3	**0,954**	**0,987**	**0,971**	**0,861**	**0,622**	**0,723**
-B 10 -E -2	0,938	0,986	0,961	0,808	0,480	0,603
-B 30 -E -2	0,945	0,989	0,967	0,865	0,541	0,666
-B 50 -E -2	**0,949**	**0,991**	**0,969**	**0,883**	**0,572**	**0,694**
-B 50 -E -1	0,949	0,991	0,969	0,883	0,572	0,694

Tabela 6 prikazuje spreminjanje rezultatov spreminjanja parametrov. Parameter -B pove število ponovitev dodajanj vozlišč pri gradnji drevesa in tako povečuje drevo, ki je rezultat učenja, hkrati pa podaljšuje čas, ki je potreben za izračun.

Parametri –3, –2 in –1 povedo, na kakšen način išče klasifikator nova potencialna vozlišča. Pri parametru 3 preveri vse možnosti, pri –2 in –1 pa omeji preiskovanje, kar pospeši iskanje, rezultat pa ni nujno optimalen (najboljše možno odločitveno drevo za dano število vozlišč).

Tabela 7: **Šolar, ADTree, MSD + skladnja −5+5, označeno brez vejic, označevalnik, naučen brez vejic**

	Ni vejice			Je vejica		
Parametri	**Natančnost**	**Priklic**	**F1**	**Natančnost**	**Priklic**	**F1**
-B 10 -E -3	0,947	0,982	0,964	0,794	0,563	0,659
-B 30 -E -3	0,953	0,988	0,970	0,865	0,612	0,717
-B 50 -E -3	**0,956**	**0,987**	**0,971**	**0,861**	**0,641**	**0,735**

Tabela 7 prikazuje rezultate za bolj realen primer, ko je označeno besedilo brez vejic, označevalnik pa je tudi naučen brez vejic. Tudi tukaj večanje drevesa izboljšuje rezultat, seveda pa zato preizkušanje traja dlje. Zadnji rezultat (s 101 listom v odločitvenem drevesu) je najboljši doseženi rezultat, ki je presegel tudi rezultat s privzetimi parametri pri klasifikatorju DecisionTable. V prihodnosti bi bilo smiselno preizkusiti različne parametre tudi pri drugih klasifikatorjih, da bi našli optimalno kombinacijo.

Dodatna prednost klasifikatorja ADTree je, da izpiše odločitveno drevo, ki bi se ga dalo relativno preprosto uporabiti v drugih programih.

```
: -1.039
| (1)je_vez1 = 1: 1.145
| (1)je_vez1 != 1: -0.335
| | (2)msd3 = *: -1.327
| | (2)msd3 != *: 0.092
| (3)lem1 = in: -1.407
| (3)lem1 != in: 0.058
| | (4)je_vez0 = 0: 0.075
| | | (6)lem0 = biti: -1.09
| | | (6)lem0 != biti: 0.087
| | | | (8)zac_modrega0 = 1: -0.526
| | | | (8)zac_modrega0 = 0: 0.092
| | | (9)msd0 = Dm: -2.691
| | | (9)msd0 != Dm: 0.021
| | | | (10)lem1 = kot: -1.264
| | | | (10)lem1 != kot: 0.026
| | (4)je_vez0 = 1: -1.14
| | (5)msd1 = Vd: 0.797
| | (5)msd1 != Vd: -0.102
| | (7)zac_modrega1 = 1: 0.419
| | (7)zac_modrega1 != 1: -0.134
Legend: -ve = ni-vejice, +ve = je-vejica
```

Slika 4: **Odločitveno drevo za ADTree -B 10 -E -3**

Slika 4 prikazuje primer odločitvenega drevesa pri -B 10 (z 21 listi). Na verjetnost, da gre za vejico, najbolj vpliva podatek iz skladenjskega razčlenjevalnika, da na naslednjo besedo kaže povezava »vez«. Zanimiv je vpliv msd3 z vrednostjo * (kar pomeni, da te besede ni), kar z drugimi besedami pomeni, da vejica tik pred koncem stavka ni posebno verjetna. V devetem volišču je zanimiv mds0 Dm, torej predlog, ki zahteva vezavo z mestnikom, ki zmanjša verjetnost, da je neposredno za njim vejica.

5 PRIMERJAVA Z DRUGIMI REZULTATI

Najboljši pridobljeni rezultat je bilo na koncu treba primerjati s prejšnjimi rezultati, najprej z rezultati metod s pravili za slovenščino, potem pa s statističnimi metodami za druge jezike.

5.1 Primerjava z metodami, ki uporabljajo pravila

Oba programa za postavljanje vejic s pravili (Besana in LanguageTool), ki sta bila preizkušena v Holozan (2012), sta bila preizkušena še za primer, ko v besedilu manjkajo vse vejice, s čimer sta bila programa, ki sta sicer namenjena popravljanju napak pri vejicah, prisiljena postaviti vse vejice v besedilo.

Postavilo se je vprašanje, kako obravnavati rezultate Besane. Ta namreč poleg opozoril, kjer točno postavi vejico, opozarja na manjkajočo vejico tudi v primerih, ko sicer ugotovi, da vejica nekje manjka, ne zna je pa točno postaviti. Ti primeri zahtevajo uporabnika, ki zna potem sam postaviti vejico na ustrezno mesto in niso primerni za samodejno postavljanje vejic, npr. pri razpoznavi govora. Zato ima Besana v tabeli dva rezultata, pri prvem so upoštevane le vejice, ki jih Besana točno postavi, pri drugem pa še tiste, za katere le ugotovi, da bi morala vejica nekje biti.

Tabela 8: **Šolar, vse vejice, ADTree (-B 50 -E -3) (označeno brez vejic, označevalnik, naučen brez vejic)**

	Ni vejice			Je vejica		
Klasifikator	**Natančnost**	**Priklic**	**F1**	**Natančnost**	**Priklic**	**F1**
ADTree	0,956	0,987	0,971	0,861	**0,641**	**0,735**
LanguageTool	0,934	0,991	0,961	0,876	0,509	0,644
Besana	0,953	0,991	0,971	**0,888**	0,572	0,696
Besana + nekje	0,950	0,988	0,969	0,871	0,624	0,727

Tabela 8 kaže, da je statistično postavljanje vejic doseglo najboljši priklic in F1, vendar je natančnost še vedno najvišja pri Besani, čeprav razlika ni velika.

5.2 Primerjava z rezultati za druge jezike

Rezultati samodejnega postavljanja vejic so zelo odvisni od jezika, kar so npr. pokazali Zhang idr. (2002), ki so preizkusili isti metodi na angleščini in nemščini.

Tabela 9: **Šolar, vse vejice, ADTree (-B 50 -E -3) (označeno brez vejic, označevalnik, naučen brez vejic)**

Jezik	Preizkus	Je vejica		
		Natančnost	Priklic	F1
Angleščina	Beeferman idr. (1998), algoritem A	0,756	0,656	0,702
Angleščina	Beeferman idr. (1998), algoritem B	0,784	0,624	0,694
Angleščina	Zhang idr. (2002), Amalgam	0,744	0,676	0,709
Angleščina	Zhang idr. (2002), jezikovno modeliranje	0,782	0,624	0,694
Angleščina	Shieber in Tao (2003)	0,797	0,626	0,748
Angleščina	Israel idr. (2012)	0,858	0,663	0,748
Nemščina	Zhang idr. (2002), Amalgam	0,854	0,875	0,865
Nemščina	Zhang idr. (2002), jezikovno modeliranje	0,896	0,746	0,815
Baskovščina	Alegria idr. (2006)	0,696	0,486	0,572
Slovenščina	Ta članek	0,861	0,641	0,735

Tabela 9 kaže, da je natančnost pri slovenščini podobna kot pri nemščini, priklic pa je slabši. Tudi najboljši rezultat za angleščino (Israel idr., 2012) ima podobno natančnost in priklic slovenskemu rezultatu.

6 ISKANJE REALNIH NAPAK

Dosedanji rezultati povedo, kako dobro postavijo programi vejice v besedilo, v katerem ni na začetku nobenih vejic, kar je npr. uporabno pri razpoznavi govora, ki ne zazna vejic. Vprašanje pa je, kako dobro se programi obnesejo pri popravljanju pravih napak, saj te niso naključno razporejene, ampak določeni tipi vejic delajo piscem več težav kot drugi. Za tak preizkus je treba dobiti korpus napak pri vejicah, kar je bilo mogoče s korpusom Šolar. Vendar pa je primerov napačnih vejic veliko manj kot vseh primerov vejic, pa še štiri možna stanja so (ob *je vejica* in *ni vejice* še *ni manjkajoče vejice* in *je odvečna vejica*) in je zato vprašanje, ali bi bilo 11399 primerov manjkajoče vejice in 2709 primerov odvečne vejice dovolj za uspešno učenje, še večji korpus primerov napak pri vejicah pa bo težko dobiti.

Zato je bil izbran drugačen postopek: program WEKA nastavimo tako, da je prvih (izključimo privzeto naključno izbiranje) 80 odstotkov primerov učni korpus, zadnjih 20 odstotkov pa uporabimo kot testni korpus, pri čemer se rezultat preizkušanja izpiše za vsak primer posebej. Ker so v korpusu Šolar primeri sicer razporejeni po razredih in letnikih oz. vrstah šol, ne bi bilo v redu, če bi vsi preizkusni primeri prišli iz istega letnika oz. šole (Holozan (2012) je pokazal, da so rezultati popravljanja vejic različni glede na letnik oz. šolo), je bil najprej izveden postopek, ki je delno premešal primere tako, da je bila najprej izločena vsaka peta poved, te izločene povedi pa so bile potem dodane na koncu.

Rezultat preizkušanja (stolpec, ki pove, katero stanje vejice je izbral klasifikator) je bil potem poravnan s podatki o vejicah iz korpusa (pri čemer je bilo treba paziti, da se je poravnalo z zadnjimi primeri in ne s prvimi), oboje je bilo sestavljeno v eno tabelo, potem pa prešteto, kolikokrat se je pojavila katera kombinacija.

```
1653-je-vejica      je-vejic
 694-je-vejica      ni-vejic
1453-manjka-vejica  je-vejic
 885-manjka-vejica  ni-vejic
 575-ni-vejice      je-vejic
36037-ni-vejice     ni-vejic
 197-prevec-vejica  je-vejic
 337-prevec-vejica  ni-vejic
```

Slika 5: **Rezultat primerjave rezultatov preizkušanja s podatki iz korpusa**

Slika 5 prikazuje tak (surov) rezultat za primer, ko je bil korpus označen z vsemi vejicami pravilno postavljenimi, spredaj je število primerov, drugi stolpec je stanje v korpusu in tretji stolpec je rezultat preizkušanja klasifikatorja, torej je npr. v 1453 primerih, ko je vejica manjkala, klasifikator menil, da bi tam morala biti vejica, v 885 primerih pa, da tam ni vejice, po drugi strani pa je v 575 primerih postavil vejico, kjer je ne bi smelo biti, natančnost (kakšen delež dodanih vejic je pravilen) je tako 1453 / (1453 + 575) oz. 71,7 odstotka.

Tak postopek je bil ponovljen za različne načine označevanja, ni pa bilo izvedeno desetkratno prečno preverjanje, ker bi bil ta postopek precej zapleten (in bi ga bilo treba prej bolj avtomatizirati, zdaj so bili nekateri koraki izvedeni ročno za vsak primer posebej). Samo 10 odstotkov primerov pri preizkušanju pa bi bilo morda tudi premalo, da bi lahko potem dovolj zanesljivo dobili rezultat pri primerjavi z napakami v korpusu, zato je bila izbrana razdelitev 80 : 20. Preizkušanje je bilo izvedeno le s klasifikatorjem ADTree s parametri (-B 14 -E -3), da ne bi trajalo predolgo.

Tabela 10: **Rezultat iskanja realnih napak, ADTree (-B 14 -E -3)**

	Popravljanje manjkajočih vejic			**Popravljanje odvečnih vejic**		
Način	**Natančnost**	**Priklic**	**F1**	**Natančnost**	**Priklic**	**F1**
Označeno z vsemi vejicami	0,717	0,622	0,666	0,327	0,631	0,431
Označeno brez vejic	0,690	0,482	0,567	0,283	0,642	0,393
Označeno brez vejic, označevalnik brez vejic	**0,676**	**0,545**	**0,603**	**0,298**	**0,633**	**0,406**
Označeno z vejicami v besedilu	0,675	0,491	0,568	0,293	0,564	0,385
Označeno z vejicami v besedilu, označevalnik brez vejic	0,672	0,541	0,600	0,292	0,592	0,391
LanguageTool	0,812	0,442	0,572	/	/	/
Besana	0,862	0,505	0,636	**0,902**	0,094	0,170
Besana + nekje	0,876	0,702	0,779	0,902	0,094	0,170

Tabela 10 prikazuje rezultat iskanja realnih napak in primerjavo z LanguageTool in Besano. Zanimivo je, da je najboljši rezultat dosežen, če pri označevanju na vhodu izbrišemo vse vejice in potem uporabimo označevanje, naučeno brez vejic (če seveda izvzamemo označevanje, pri katerem so vse vejice postavljene pravilno, česar seveda normalno nimamo). Če že postavljene vejice pri označevanju pustimo v besedilu, je rezultat torej slabši, in sicer ne glede na to, ali je označevalnik naučen z vejicami ali brez njih.

Zanimiv je tudi rezultat pri odkrivanju odvečnih vejic, pri čemer statistična metoda sicer doseže veliko boljši priklic (0,633 proti 0,094), vendar hkrati tudi neuporabno nizko natančnost (0,298 proti 0,902) (tukaj bi bilo smiselno preizkusiti še idejo iz Israel idr. (2012), da ne upoštevamo le dejstva, da se je klasifikator odločil, da neke vejice ni, temveč tudi njegovo oceno te odločitve, tako da vejico označi kot odvečno le, če ta ocena preseže določeno mejo). Tudi pri manjkajočih vejicah je težava predvsem natančnost, priklic je boljši od LanguageTool in Besane (razen če pri Besani upoštevamo še opozorila, da nekje manjka vejica).

Opozoriti je treba še, da je gostota napak v teh primerih velika, saj so bile preverjene le povedi, v katerih je bila bodisi kakšna odvečna bodisi manjkajoča vejica. Zato bi bilo treba pripraviti boljši korpus napak, ki bi vključeval tudi pravilne stavke, da bi dobili pravo natančnost. Je pa natančnost zelo odvisna od kakovosti vhodnega besedila, če natančnost preizkušamo na besedilu, ki nima (ali skoraj nima) napak, bo natančnost slabša, kot če je napak veliko.

Za angleščino so Israel idr. (2012) dosegli natančnost 0,849 pri priklicu 0,200 (F1 0,324), vendar je to rezultat za vse napačne vejice, ni pa posameznih rezultatov za manjkajoče oz. odvečne vejice.

7 SKLEP

Poskusi so pokazali, da je postavljanje vejic z uporabo strojnega učenja zelo uporabno v primeru, ko želimo poiskati vse vejice v besedilu. Za najboljši rezultat je treba uporabiti označevanje z označevalniki, ki so bili naučeni z učnimi korpusi z odstranjenimi vejicami, uporabiti je treba skladenjsko razčlenjevanje, kot najbolj uporaben se je pokazal klasifikator ADTree (alternirajoče odločitveno drevo), njegova prednost je tudi preprosto odločitveno drevo, ki bi se dalo hitro sprogramirati tudi v kakšnem programu. Rezultati se izboljšujejo z večanjem drevesa, vendar hkrati narašča potrebni čas za izračun, najuspešnejši poskus je bil izveden z nastavitvami -B 50 -E 3 z oknom –5+5. Rezultat za slovenščino je primerljiv z rezultati za druge jezike, dosežena je bila natančnost 0,861, priklic 0,641 in F1 0,735.

Glede na to, da program WEKA podpira veliko število klasifikatorjev, čisto vsi niso bili preizkušeni, pa tudi pri tistih, ki so bili, je odprtih še veliko možnih poskusov s parametri klasifikatorjev. Problem je tudi čas, ki je potreben za izračunavanje; pri klasifikatorju ADTree se je pokazalo, da večanje drevesa izboljšuje rezultat, vendar zgornja meja ni bila dosežena, ker postane preračunavanje pri tako velikih drevesih prepočasno (najboljši rezultat se je računal skoraj tri dni). Vsekakor je še veliko možnih kombinacij klasifikatorjev, parametrov, različnih atributov, oken, pri katerih bi bilo verjetno mogoče doseči še boljši rezultat.

Odločitveno drevo, ki je rezultat, bi se morda dalo uporabiti za izboljšavo postopkov postavljanja vejic s pravili, označevanje besedila je sicer relativno zahtevna operacija, kar bi lahko povzročilo težave pri praktični uporabi (npr. kot slovnični pregledovalnik v urejevalniku besedil). V ta namen bi bilo zato morda smiselno poskusiti zgraditi odločitveno drevo s pomočjo atributov, ki jih je laže dobiti, morda celo samo iz samih besed.

Uspeh pri iskanju realnih napak je slabši kot pri iskanju vseh vejic. Rezultati s strojnim učenjem imajo sicer dober priklic (0,545), vendar je natančnost (0,676) slabša od Besane in LanguageTool. Še posebno pa je to očitno pri popravljanju odvečnih vejic, česar LanguageTool sploh ne opravlja, Besana pa ima tudi priklic le 0,094, vendar doseže natančnost 0,902, medtem ko je statistično popravljanje doseglo priklic kar 0,633, vendar je natančnost le 0,298. Zanimivo je, da je bil najboljši rezultat dosežen v primeru, ko so bile v besedilu pred označevanjem izbrisane vse vejice (in je bil tudi označevalnik naučen brez vejic); tudi pravilne vejice so označevanje motile, kar je presenetljiv rezultat. Se pa lahko ta rezultat spremeni, če se bo povečal delež pravilnih vejic v preizkusnem korpusu, zdaj so namreč v njem le povedi z napačnimi vejicami, zaradi tega je tudi natančnost nerealno visoka.

Naloga za prihodnost je razširiti dosedanji preizkusni korpus, pridobljen iz korpusa Šolar, še s pravilnimi povedmi iz korpusa Šolar, ki nastopajo ob povedih z napakami, in potem ponoviti ta poskus. Smiselno bi bilo dodati še primere iz drugih virov, ki so dostopni pod licenco Creative Commons (npr. Wikipedije), in označiti napačne vejice in tako zgraditi in objaviti referenčni korpus za učenje/popravljanje vejic, ki bi bil dostopen pod licenco Creative Commons, s čimer bi ga lahko za eksperimente uporabljali tudi drugi, tako da bi bili rezultati bolj primerljivi.

8 VIRI IN LITERATURA

[1] Alegria, I., Arrieta, B., de Ilarraza Sánchez, A. D., Izagirre, E. & Maritxalar, M. (2006). *Using Machine Learning Techniques to Build a Comma Checker for Basque*. V N. Calzolari, C. Cardie & P. Isabelle (ur.), ACL : The Association for Computer Linguistics.

[2] Beeferman D., Berger A. & Lafferty J. (1998). *Cyberpunc: A lightweight punctuation annotation system for speech*. IEEE Conference on Acoustics, Speech and Signal Processing. Seattle, WA, USA.

[3] Hardt, D. (2001). *Comma checking in Danish*. Paper presented at Corpus Linguistics 2001 conference: Lancaster University (UK), 266–271.

[4] Hillard, D., Huang, Z., Ji, H., Grishman, R., Hakkani-Tur, D., Harper, M., Ostendorf, M., Wang, W. (2006). *Impact of Automatic Comma Prediction on Pos/Name Tagging of Speech*. V zborniku IEEE/ACL 2006 Workshop on Spoken Language Technology.

[5] Holozan, P. (2012). *Kako dobro programi popravljajo vejice v slovenščini*. V zborniku Jezikovne tehnologije: Zbornik C 15. mednarodne multikonference Informacijska družba IS 2012, 8. do 12. oktober 2012, Erjavec, T., Žganec Gros, J.; Ljubljana: Institut Jožef Stefan, okt. 2004, str. 101–106.

[6] Huang, J. & Zweig, G. (2002). *Maximum entropy model for punctuation annotation from speech*. V J. H. L. Hansen & B. L. Pellom (ur.), INTERSPEECH : ISCA.

[7] Israel R., Tetreault J. & Chodorow M. (2012). *Correcting Comma Errors in Learner Essays, and Restoring Commas in Newswire Text*. 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies; Montreal, Canada, June 3–8, 2012, str. 284–294.

[8] Shieber, S. M. & Tao, X. (2003). *Comma restoration using constituency information*. V Proceedings of the 2003 Human Language Technology Conference and Conference of the North American Chapter of the Association for Computational Linguistics.

[9] Šek Mertük, P. (2011). *Vejica premalo ali preveč pri študentih razrednega pouka*. Revija za elementarno izobraževanje. Letnik 4, št. 1–2. 123–146.

[10] Zhang, Z., Gamon, M., Corston-Oliver, S., Ringger, E. (2002). *Intra-sentence punctuation insertion in natural language generation*. Tehnično poročilo MSR-TR-2002-58. Microsoft Research.

▪

Peter Holozan je razvijalec v podjetju Amebis, d. o. o., Kamnik in raziskovalec v Amebisovem razvojnem centru. Magistriral je na Fakulteti za računalništvo in informatiko Univerze v Ljubljani in je doktorski študent na Filozofski fakulteti Univerze v Ljubljani (slovenistika). Ukvarja se predvsem z jezikovnimi tehnologijami za slovenščino, med drugim s črkovalniki, slovničnim pregledovalnikom, strojnim prevajanjem, oblikoskladenjskim označevanjem, korpusi (Fida, FidaPLUS) in slovarji (ASP32).

Govorni in jezikovni viri slovenščine za samodejno razpoznavanje tekočega govora

Gregor Donaj, Andrej Žgank, Mirjam Sepesy Maučec
Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, Smetanova ul. 17, 2000 Maribor
gregor.donaj@um.si, andrej.zgank@uni-mb.si, mirjam.sepesy@uni-mb.si

Izvleček

Govor je za ljudi najbolj naravno komunikacijsko sredstvo. Govorno komunikacijo s strojem omogočajo sistemi za samodejno razpoznavanje govora. Različne aplikacije razpoznavanja govora so za stroj različno zahtevne. Med najzahtevnejše štejemo samodejno razpoznavanje tekočega govora. Aplikacije razpoznavanja govora temeljijo na statistični obdelavi govornega signala ter gradnji akustičnih in jezikovnih modelov. Za izdelavo teh modelov je pomembna uporaba kakovostnih govornih in jezikovnih virov. V prispevku opisujemo govorne in jezikovne vire za slovenščino, ki se uporabljajo za samodejno razpoznavanje govora. Predstavimo tudi modularno zgradbo razpoznavalnika. V eksperimentalnem sistemu analiziramo vpliv uporabe modelov v razpoznavalniku tekočega govora v domeni dnevnoinformativnih oddaj.
Ključne besede: govorni viri, jezikovni viri, akustični modeli, jezikovni modeli, samodejno razpoznavanje govora.

Abstract

Slovene Speech and Language Resources for Automatic Speech Recognition

Speech is the most natural way of communicating. Speech communication with machines is made possible with systems for automatic speech recognition. Different applications of speech recognition are differently challenging. Among the most challenging is continuous speech recognition. Speech recognition systems are based on statistical speech signal processing and the building of acoustical and language models. Quality speech and language resources are needed to build these models. This paper gives an overview of speech and language resources for Slovene, which are used in automatic speech recognition. A modular structure of a speech recognizer is also presented. In an experimental system the impact of using different models on the accuracy in a Broadcast News speech recognition system is analyzed.
Key words: speech resources, language resources, acoustical models, language models, automatic speech recognition.

1 UVOD

Govor kot človekovo najbolj naravno komunikacijsko sredstvo pomeni za stroj zelo kompleksno nalogo. Razpoznavanje tekočega govora in razpoznavanje spontanega govora sta za raziskovalce polna izzivov. Posebnosti posameznih jezikov razpoznavanje govora še dodatno zapletejo. Tudi slovenščina kot visoko pregibni jezik spada v skupino bolj zahtevnih jezikov za razpoznavanje.

Poznamo različne pristope samodejnega razpoznavanja govora (angl. Automatic Speech Recognition, ASR). Med preprostejše štejemo razpoznavanje izoliranih besed z majhnim slovarjem, med zahtevnejše pa razpoznavanje tekočega govora z velikim slovarjem (Sepesy Maučec, Rotovnik, Kačič & Brest, 2009). Za obe aplikaciji je pomembno, da imamo izdelane dobre modele govora. V primeru razpoznavanja izoliranih besed so predvsem pomembni akustični modeli, ki modelirajo akustične značilnosti govora. Ti modeli služijo prepoznavanju fonemov in besed. Razpoznavanje tekočega govora pa pomeni še večjo zahtevnost za akustično modeliranje, saj je treba upoštevati tudi prehode med besedami, ker so v tekočem govoru zabrisane meje med besedami. Dodatno so pri razpoznavanju tekočega govora velikega pomena statistični jezikovni modeli. Z njimi modeliramo verjetnosti zaporedij besed v jeziku. Pri izdelavi jezikovnih modelov se pogosto poslužujemo pisnih virov jezika. Posledično so jezikovni modeli bolj primerni za razpoznavanje branega govora, manj pa za razpoznavanju spontanega govora (Žgank & Sepesy Maučec, 2010).

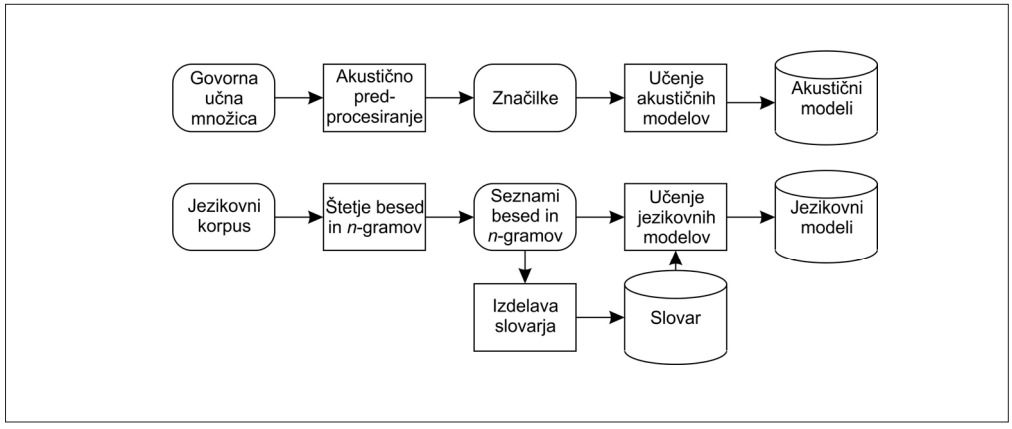
Tako za izdelavo akustičnih kot jezikovnih modelov so pomembni kakovostni in dovolj obsežni govorni oz. pisni viri jezika. V članku bomo predstavili nekatere takšne vire, ki so na voljo za slovenski jezik.

Njihovo uporabnost bomo predstavili na primeru razpoznavalnika tekočega govora UMB Broadcast News, ki je bil razvit na Fakulteti za elektrotehniko, računalništvo in informatiko v Mariboru.

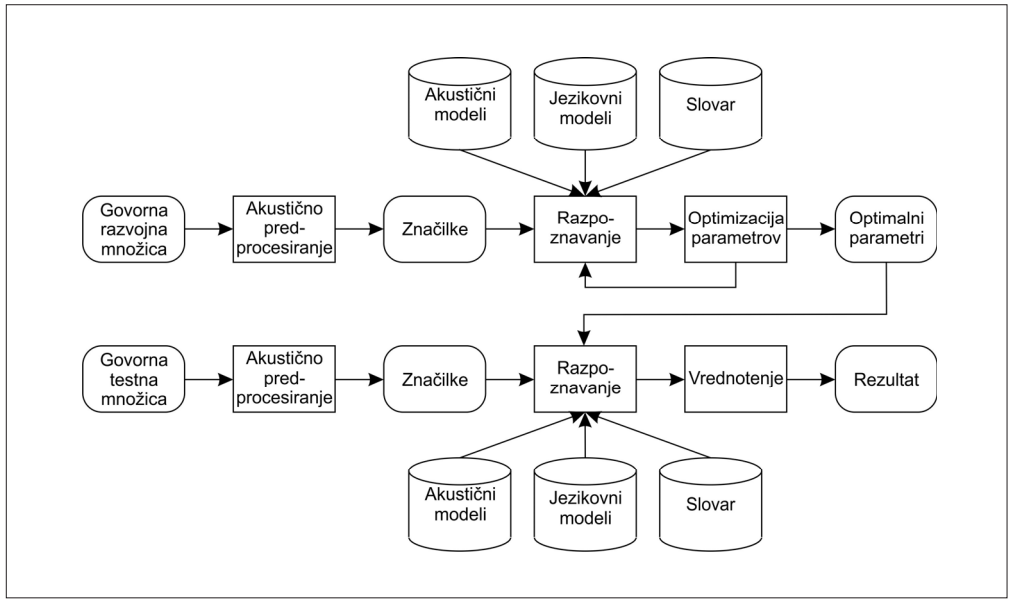
V drugem razdelku bomo predstavili osnovno zgradbo in module sistema za ASR. V tretjem razdelku bomo opisali posebnosti slovenščine, zaradi katerih je ta za razpoznavanje govora večji izziv. Sledi opis osnovnih govornih in jezikovnih virov za slovenščino, ki so uporabni za gradnjo sistemov ASR. V četrtem razdelku je opisan eksperimentalni sistem, v petem razdelku pa rezultati eksperimentov. V šestem razdelku sledi sklep.

2 SAMODEJNO RAZPOZNAVANJE GOVORA

Delovanje sistemov za samodejno razpoznavanje govora delimo na dve fazi. Prva faza je učenje jezikovnih in akustičnih modelov. Blokovna shema učenja modelov je prikazana na sliki 1. Končni rezultat te faze so akustični in jezikovni model ter slovar besed.



Slika 1: **Postopek učenja akustičnih in jezikovnih modelov**



Slika 2: **Delovanje razpoznavanja govora**

Druga faza je razpoznavanje. Njena blokovna shema je prikazana na sliki 2. Sistem za razpoznavanje govora na vhodu sprejme zvočni signal, na izhodu pa posreduje razpoznano zaporedje besed. Sistem ima modularno zgradbo, module pa lahko razdelimo v dve skupini: na module za predprocesiranje

govora in module za razpoznavanje govora. Vhodni zvočni signal najprej obdela modul za akustično segmentacijo, ki zvočni signal razdeli na akustično homogene dele. Modul za akustično analizo izlušči informacijo v govoru in jo predstavi z vektorjem akustičnih značilk. Postopek izločanja značilk mora biti popolnoma enak kot pri učenju akustičnih modelov. Niz vektorjev značilk je vhodni podatek iskalnega algoritma, ki poišče najbolj verjetno zaporedje izgovorjenih besed. Pri tem uporablja informacijo iz akustičnih in jezikovnih modelov. Akustični modeli opisujejo akustične lastnosti govora na ravni fonemov, jezikovni modeli pa jezikovne lastnosti govora na ravni besed. Oboji, tako akustični kot jezikovni modeli, temeljijo na statističnem procesiranju govora oz. jezika. Razpoznavanje na razvojni množici poteka z namenom iskanja optimalnih parametrov razpoznavanja – uteži akustičnih in jezikovnih modelov. Končni rezultat uspešnosti razpoznavanja dobimo na testni množici, pri čemer uporabimo optimizirane vrednosti parametrov.

2.1 Akustični modeli

Akustični modeli so ključni gradnik samodejnega razpoznavalnika govora s stališča procesiranja govornega signala. Njihova naloga je modelirati akustično-fonetične lastnosti govora, pri tem pa v primeru razpoznavanja govora neodvisnega govorca uspešno zmanjšati razlike med posameznimi govorci. Osnovna enota akustičnih modelov je običajno fonem, ki ga zaradi modeliranja učinka koartikulacije modeliramo v širšem kontekstu predhodnega in naslednjega fonema. Takšen akustični model poimenujemo trifon. Na trifon lahko gledamo kot na posplošitev pojma alofon. Alofoni so različne možne izgovorjave nekega fonema glede na njegov kontekst. Za vsak fonem imamo običajno le majhno množico alofonov. Definicija trifona pa zajema vse možne kombinacije treh zaporednih fonemov (za *N* fonemov pomeni to *N*3 trifonov). Medtem ko definicija alofona izhaja iz fonologije, pa trifone uvajamo v obdelavi govora zaradi zveznih sprememb vokalnega trakta, ki nastopijo pri prehodu iz izgovarjave enega fonema na naslednjega in se odražajo v akustičnem signalu govora ob tem prehodu. Primer fonetične in grafemske oblike vnosa besede »avtomatskega« v slovarju razpoznavalnika govora je prikazan v tabeli 1.

Tabela 1: **Primer fonetične in grafemske oblike vnosa v slovar razpoznavalnika govora**

Beseda	Kategorija transkripcije	Transkripcija
avtomatskega	MRPA fonemi	a U t O m “a: ts k E g a
avtomatskega	Grafemi	a v t o m a t s k e g a

Za akustično modeliranje pri ASR se uporabljajo različni pristopi (Aubert, 2002), najpogostejši so prikriti modeli Markova (angl. Hidden Markov Model, HMM), uteženi končni pretvorniki (angl. Weighted Finite State Transducer, WFST) in nevronske mreže (angl. Artificial Neural Network, ANN). V predstavljenem eksperimentu smo uporabljali tristanjske levo-desne prikrite modele Markova z zveznimi Gaussovimi porazdelitvenimi funkcijami verjetnosti. Za slovenski jezik je pretvorba med grafemi in fonemi netrivialen proces, ki lahko k rezultatom razpoznavanja govora vnese dodatno napako.

2.2 Jezikovni modeli in slovarji

Pri razpoznavanju govora so meje med besedami zabrisane, saj v tekočem govoru med besedami ni premorov. Za določanje zaporedja besed so najprej uporabljali deterministične besedne mreže, ki so jih nasledili jezikovni modeli, temelječi na pravilih slovnice jezika. Sestavljanje slovničnih pravil, ki bi pokrila jezik kot celoto, je zelo zahtevna naloga, ki zahteva poglobljeno znanje o jeziku. Po drugi strani pa imamo v spontano govorjenem jeziku veliko slovnično nepravilnih zaporedij. Ideja jezikovnega modela je določiti verjetnost poljubnemu zaporedju besed. Jezikovni model lahko obravnavamo tudi kot model, ki v procesu razpoznavanja napoveduje najbolj verjetno naslednjo besedo. Za jezikovni model velja tudi to, da verjetnost zaporedja besed ni nikoli enaka nič, kar je še posebno dobrodošlo pri razpoznavanju spontanega govora. V praksi so se najbolj uveljavili statistični n-gramski jezikovni modeli, ki verjetnost poljubnega zaporedja besed izračunajo s sestavljanjem verjetnosti n-gramov. V jezikovnih modelih označuje n-gram zaporedje *n* besed, *n* pa določa red n-grama. Najpogostejši so bigrami (2-grami) in trigrami (3-grami), zasledimo pa tudi uporabo jezikovnih modelov do reda 5 (tj. 5-gramov). Smiselnost uporabe jezikovnih modelov višjih redov je povezana z velikostjo učnega korpusa, tj. besedila, v katerem štejemo modelirane n-grame. Da je verjetnost poljubnega zaporedja besed vedno večja od 0,

zagotavljajo metode glajenja verjetnosti (Chen & Goodman, 1999), ko določeno, resda majhno, verjetnost pripišejo tudi n-gramom, ki se nikoli ne pojavijo v učnem korpusu. Preliminarne raziskave so pokazale, da je za modeliranje slovenskega jezika najučinkovitejše glajenje, ki temelji na Good-Turingovem glajenju (Good, 1953) in sestopanju po Katzu (1987).

Jezikovni modeli opisujejo verjetnostne lastnosti n-gramov besed. Katere besede vsebujejo n-grami, določa slovar. Vse besede zunaj slovarja se preslikajo v simbol OOV (angl. Out-Of-Vocabulary). To pomeni, da bo beseda, ki ni v slovarju, napačno razpoznana. Napačno razpoznana beseda pa vpliva tudi na razpoznavanje besed, ki ji sledijo, saj predstavlja njihov kontekst. Pomembna je tudi velikost slovarja, saj je z velikostjo neposredno povezana kompleksnost razpoznavalnika in s kompleksnostjo tudi hitrost razpoznavanja. V sistemih razpoznavanja visoko pregibnih jezikov so neizogibni veliki slovarji, razen če je razpoznavanje omejeno na zelo specifično domeno (npr. razpoznavanje vremenske napovedi).

Beseda je praviloma osnovna enota v slovarju. Za modeliranje pregibnih jezikov so bile izvedene številne raziskave uporabe manjših osnovnih enot (morfemov, osnov in končnic besed ipd.), ki pa se niso izkazale kot bistveno boljše, saj je napovedna moč jezikovnih modelov s prehodom na manjše osnovne enote oslabljena (Sepesy Maučec idr., 2009).

2.3 Iskalni algoritmi

Naloga razpoznavalnika govora je poiskati najbolj verjetni niz besed za zajeti vhodni govor. Iskanje izvedemo s pomočjo iskalnih algoritmov (Aubert, 2002). Pri iskanju najbolj verjetnega zaporedja besed ni moč pregledati celotnega iskalnega prostora, ga pa omejujemo z različnimi hevrističnimi metodami. Razlikujemo statično omejevanje (npr. drevesna predstavitev slovarja) in dinamično omejevanje iskalnega prostora (npr. snopovno omejevanje, pogled naprej v jezikovni model ipd). Same iskalne algoritme delimo na časovno sinhrone in asinhrone glede na to, ali hipoteze v iskalnem prostoru ocenjujemo vzporedno od začetka do konca govornega segmenta ali pa vse ocenjujemo ob koncu segmentov.

Poznamo tudi dvoprehodne algoritme (Lee, Kawahara & Doshita, 1998), ki predstavljajo eno od metod za izboljšanje hitrosti delovanja algoritmov. Pri teh algoritmih najprej uporabimo samo določene jezikovne vire za samodejno razpoznavanje segmenta govora. To imenujemo prvi prehod. Kot njegov rezultat dobimo ali seznam najboljših hipotez (običajno od 100 do 1000) ali pa besedno mrežo. V drugem prehodu nato uporabimo vse razpoložljive vire in modele za ocenjevanje hipotez v seznamu oz. mreži.

2.4 Vrednotenje uspešnosti razpoznavalnika

Predlagane metode in algoritme na področju ASR najpogosteje vrednotimo posredno z uporabo rezultatov razpoznavanja govora. Vrednotenje praviloma izvajamo z ločenim testnim naborom posnetkov, ki je sicer po svojih lastnostih podoben učnemu setu, vendar ni bil uporabljen nikjer v postopku učenja akustičnih modelov. Tako je eden izmed ključnih vidikov učenja akustičnih modelov skrb, da ne pride do efekta »prenaučenja«, s čimer bi se zmanjšala njihova splošnost, nujno potrebna za uspešno vrednotenje.

Pri vrednotenju rezultatov ASR je treba upoštevati tako delež pravilno razpoznanih besed, kot tudi tiste besede, ki so bile vrinjene. Tako lahko definiramo pravilnost razpoznanih besed (ACC) kot:

$$ACC = \frac{H - I}{N}\ 100\ \%$$

pri čemer je H število vseh pravilno razpoznanih besed, I število vrinjenih besed in N število vseh besed v testni množici.

3 RAZPOZNAVANJE SLOVENSKEGA JEZIKA

Za jezikovno modeliranje je skoraj idealna angleščina. Ima malo besednih oblik in vnaprej določen vrstni red besed v povedih. Slovenščina je za razpoznavanje eden od zahtevnejših jezikov. Težave povzroča predvsem bogato pregibanje besed in relativno sproščen vrstni red, izrazit predvsem v spontanem govoru. Bogato pregibanje besed se odraža na velikosti slovarja. Za zadovoljivo pokritost besedišča mora slovar vsebovati več kot 200.000 besed, saj pomeni vsaka besedna oblika nov vnos v slovar. Po drugi strani je za učenje jezikovnega modela s tako velikim slovarjem potreben večji učni korpus, saj imamo pri majhnih korpusih težave zaradi prevelike razpršenosti podatkov. Velikost učnega korpusa danes ni več tako pereča, saj obstajajo zelo obsežne besedilne zbirke (Arhar & Gorjanc, 2007). Opozoriti pa velja, da so to zbirke pisanega jezika, ki ne odražajo značilnosti govorjenega jezika.

Razpršenost podatkov lahko zmanjšamo z lematizacijo. Lematizacija je določanje osnovne slovarske

oblike posameznim besedam v korpusu. Slovarski obliki pravimo lema. Slovar lem je v primerjavi s slovarjem besednih oblik nekajkrat manjši. Seveda pa jezikovnega modela besednih oblik ne moremo preprosto zamenjati z jezikovnim modelom lem, saj je za razpoznavalnik pomembna besedna oblika in ne zgolj lema. Uveljavilo se je modeliranje, ki razen lem modelira tudi t. i. oblikovno skladenjske oznake (angl. Morpho-Syntactic Description tags – MSD), ki če so pripete lemi, enolično določajo besedno obliko. Ker se izbrana lema lahko pojavi v mnogo različnih besednih oblikah, je število različnih MSD oznak za slovenski jezik nekajkrat večje kot za angleški jezik.

3.1 Govorni viri

Govorni in jezikovni viri so ključni pogoj za razvoj samodejnega razpoznavalnika govora. Pri tem je bistvenega pomena jezikovna odvisnost virov, saj v normalnih scenarijih razvoja samodejnega razpoznavalnika govora ne moremo uporabljati virov drugega jezika. Izdelava novega vira je časovno, stroškovno in organizacijsko zelo zahteven proces, saj je treba ročno izdelati transkripcije (prepise) z dobesednim zapisom izgovorjenega, označiti govorce, meje med segmenti, akustično ozadje itn. V povprečju je treba za izdelavo ure transkribirane govorne baze opraviti približno trideset ur dela. Navedene omejitve pri izgradnji govornih virov so še posebno izrazite pri jezikih z manjšim številom govorcev, pri čemer je manjši tudi komercialni interes. Zaradi specifičnih lastnosti jezikov virov ne moremo neposredno primerjati med seboj, temveč je treba pri primerjavi upoštevati jezikovno specifično komponento.

Slovenski jezik spada v skupino jezikov z izdelanimi osnovnimi viri za gradnjo samodejnih razpoznavalnikov govora (Kačič, 2002; Žganec Gros, Mihelič & Dobrišek, 2003). Začetki razvoja govornih virov za slovenski jezik segajo v devetdeseta leta prejšnjega stoletja. Prvi slovenski govorni viri so spadali v kategorijo razpoznavanja izoliranih in vezanih besed v telefonskem ali studijskem okolju. Na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru so bile tako razvite govorne baze SNABI, Slovenian 1000 FDB SpeechDat(II) (Kačič & Kaiser, 1998) in Polidat (Žgank, Kačič & Horvat, 2002). S stališča razvoja samodejnih razpoznavalnikov govora sta še posebno pomembni bazi SpeechDat(II) in Polidat, saj spadata v družino mednarodnih standardiziranih govornih baz, ki omogočajo razvoj govorno vodenih telekomunikacijskih storitev. Na Fakulteti za elektrotehnoiko Univerze v Ljubljani je bila za razvoj samodejnih razpoznavalnikov govora razvita baza Gopolis (Mihelič, Žganec Gros, Dobrišek, Žibert & Pavešić, 2003), ki je bila v kombinaciji z dodatnima bazama uporabljena za razvoj razpoznavalnika govora za omejeno domeno (Dobrišek, Vesnicer, Žganec Gros & Mihelič, 2006).

S stališča ASR je bistveno kompleksnejši problem razpoznavanje tekočega govora neodvisnega govorca z velikim slovarjem besed. Prva slovenska govorna baza, ki je podpirala to kategorijo govora, je bila baza Slovenian BNSI Broadcast News (Žgank, Verdonik, Zögling Markuš & Kačič, 2005), razvita leta 2005 v sodelovanju med Fakulteto za elektrotehniko, računalništvo in informatiko Univerze v Mariboru in RTV Slovenija. Govorna baza je dostopna prek mednarodne organizacije ELRA/ELDA. Namenjena je samodejnemu razpoznavanju tekočega slovenskega govora v različnih televizijskih oddajah. To bazo smo uporabili tudi v okviru eksperimentov, predstavljenih v tem članku. Na Fakulteti za elektrotehniko Univerze v Ljubljani je bila razvita baza SiBN Broadcast News (Žibert & Mihelič, 2004), ki je prav tako namenjena razpoznavanju tekočega govora v televizijskih oddajah. V okviru sodelovanja med Fakulteto za elektrotehniko, računalništvo in informatiko Univerze v Mariboru in državnim zborom Republike Slovenije je bila razvita govora baza SloParl (Žgank, Rotovnik, Grašič, Kos, Vlaj & Kačič, 2006), ki vsebuje posnetke sej državnega zbora. Baza obsega sto ur govora in je tako trenutno najobsežnejši govorni vir za slovenski jezik. Od preostalih slovenskih govornih baz se loči po transkripcijah govora, saj so bile transkripcije narejene na podlagi magnetogramov in ne vsebujejo dobesednega zapisa izgovorjenega. Takšno govorno bazo uporabljamo v posebnih postopkih učenja akustičnih modelov, pri čemer upoštevamo prisotnost napak v učnih transkripcijah.

Govorni bazi Slovenian BNSI Broadcast News in SloParl vsebujeta tudi besedilni korpus za učenje jezikovnih modelov samodejnega razpoznavalnika govora. Oba besedilna korpusa sta po svojih značilnostih identična govoru v govorni bazi. Tako lahko besedilna korpusa uporabljamo za izdelavo interpoliranih jezikovnih modelov, ki uspešno modelirajo tudi značilnosti govorjenega jezika. Jezikovni modeli so zaradi potrebe po dovolj velikem učnem vzorcu (reda 100 M besed) običajno izdelani na besedilnih

korpusih pisanega jezika (časopisi, knjige, splet), ki po svojih značilnostih bistveno odstopa od govorjenega jezika.

Slovenski govorni viri sicer pokrivajo osnovna področja razvoja samodejnih razpoznavalnikov govora, vendar je obseg razpoložljivih slovenskih govornih virov manjši v primerjavi z jeziki z večjim številom govorcev (angleščina, nemščina, španščina, kitajščina). Hkrati pa je slovenski jezik zaradi svojih značilnosti za ASR bistveno kompleksnejši problem. Glavni značilnosti slovenščine, ki otežita razpoznavanje govora, sta visoka pregibnost in relativno prosti vrstni red besed v stavku. Glede na izvedene analize bi tako za slovenski jezik potrebovali vsaj desetkrat večje govorne vire kot za angleški jezik (Rotovnik, Sepesy Maučec & Kačič, 2007). Če je stanje na področju osnovnih slovenskih govornih virov zadovoljivo, pa za slovenski jezik ne obstajajo bolj specifični govorni viri, ki jih poznamo za jezike z večjim številom govorcev. V to kategorijo spadajo npr. govorni viri, posneti v avtomobilu ali na motorju, govorni viri, posneti v različnih šumnih okoljih, govorni viri, posneti na sestankih, govorni viri, posneti v inteligentnem okolju itn.

V predstavljenih eksperimentih smo uporabili govorno bazo Slovenian BNSI Broadcast News. Baza vsebuje transkribirane posnetke 42 dnevnoinformativnih oddaj RTV Slovenija (TV Dnevnik, Odmevi) iz obdobja 1999–2003. Kot učni korpus uporabljamo trideset ur posnetkov, tri ure so namenjene razvojnemu testiranju ter tri ure vrednotenju. Posnetki vsebujejo 1565 različnih govorcev, od tega 1069 moških in 477 žensk. Za 19 govorcev ni bilo mogoče zanesljivo določiti spola zaradi značilnosti akustičnega kanala (kratki odseki, prekrivajoči se govori). Za vsakega govorca je bilo ustrezno določeno njegovo narečje. V transkripcijah so ustrezno označene akustične lastnosti (studio/telefon, akustično ozadje) posnetkov ter lastnosti govora in govorcev (brani/spontani govor, prekrivanje govorcev, tuji govorci). Na podlagi teh lastnosti so segmenti razdeljeni v ustrezne »f-kategorije«. Glede na vsebino prispevka so bili posnetki razdeljeni v petnajst različnih topikov, s pomočjo katerih je mogoče omejiti domeno samodejnega razpoznavalnika govora in tako izboljšati rezultate. V transkripcijah baze BNSI je 268.000 besed, od tega 37.000 različnih.

3.2 Jezikovni viri

Za izdelavo jezikovnih modelov potrebujemo dovolj velike korpuse jezika, ki nam služijo kot učna množica. Prvi obsežen korpus slovenskega jezika je bil korpus FIDA, ki se je kasneje nadgradil v korpus FidaPLUS (Arhar & Gorjanc, 2007), ki ga tudi uporabljamo za gradnjo jezikovnih modelov v razpoznavalniku UMB Broadcast News. FidaPLUS je največji korpus, ki nam je trenutno na voljo. Vsebuje približno 621 milijonov besed. Največji delež besedil glede na zvrst predstavljajo neumetnostna nestrokovna besedila. Glede na tip prevladujeta časopisno in revijalno gradivo. Podrobnejše podatke o sestavljenosti korpusa lahko najdemo v Arhar & Gorjanc (2007). Besede v korpusu so tudi samodejno označene s pripadajočimi lemami in oznakami MSD.

Korpus FidaPLUS je bil kasneje nadgrajen še v korpus Gigafida (Arhar Holdt, Kosem & Logar Berginc, 2012), ki nam trenutno še ni na voljo. Ta korpus vsebuje približno 1,1 milijarde besed, ki so prav tako označene z lemami in oznakami MSD.

Za razpoznavanje govora so se poleg osnovnih besednih oblik izkazale kot uporabne tudi dodatne jezikovne informacije. Za slovenski jezik so tukaj lahko uporabne besedne leme in oznake MSD. Da jih lahko uporabimo v razpoznavanju govora, potrebujemo jezikovne vire s čim bolj natančnimi oznakami in pomoč označevalnika med samim postopkom razpoznavanja.

Ker vsako samodejno označevanje korpusov z oznakami MSD vnaša napake, je smiselno uporabiti korpuse, ki so bili označeni ali vsaj pregledani ročno. Tak korpus je npr. jos100k (Erjavec & Krek, 2008), ki je nastal v okviru projekta Jezikovno označevanje slovenščine (JOS). Korpus je bil kasneje v projektu Sporazumevanje v slovenskem jeziku (SSJ) razširjen v korpus ssj500k (Arhar, 2009). Ta vsebuje približno 500.000 besed, označenih z oznakami MSD, ki so pregledane ročno.

Ta korpus je sicer veliko manjši od korpusa FidaPLUS, vendar je kljub temu uporaben za izdelovanje statističnih modelov oznak MSD. Medtem ko slovarji besed lahko vsebujejo do več sto tisoč enot, lahko vsebujejo slovarji oznak MSD le nekaj sto do nekaj tisoč enot, odvisno od kompleksnosti oznak. V okvirju projekta JOS so bila definirana tudi pravila za obliko oznak MSD. Po sistemu JOS poznamo skupaj 1.903 različnih oznak MSD. Število teh oznak lahko zmanjšamo s poenostavljanjem. Tako lahko iz oznak izpu-

ščamo podatke, ki so manj pomembni za razpoznavanje. Zaradi veliko manjšega števila različnih enot v slovarju je treba za gradnjo statističnega modela oceniti bistveno manj parametrov. Zato za gradnjo modelov oznak MSD ni potrebna tako velika učna množica kot pri modelih besed.

Prav tako je v okviru projekta SSJ nastal oblikoskladenjski označevalnik in lematizator Obeliks (Grčar, Krek & Dobrovoljc, 2012). Označevalnik prav tako potrebuje statistične modele, ki so naučeni na neki učni množici. Označevalnik pripisuje besedam leme in oznake MSD po sistemu JOS.

4 EKSPERIMENTALNI SISTEM

Vsi predstavljeni eksperimenti so bili izvedeni na razpoznavalniku tekočega govora UMB Broadcast News (Žgank & Sepesy Maučec, 2010). Trenutno v njem uporabljamo dvoprehodni algoritem razpoznavanja. Za učenje akustičnih modelov in razpoznavanje v prvem prehodu smo uporabljali orodja iz zbirke HTK (Young, Jansen, Odell, Ollason & Woodland, 1996), za gradnjo slovarjev, jezikovnih modelov in razpoznavanje v drugem prehodu pa orodja iz zbirke SRILM (Stolcke, Zheng, Wang & Abrash, 2011).

Prvi korak v postopku akustičnega modeliranja je izločanje značilk iz govornega signala. Vhodni signal s funkcijo okna dolžine 25 ms, ki ga premikamo s koraki10 ms, razdelimo na kratkočasovne vzorce. Po izvedbi predpoudarjanja izračunamo 12 mel-kepstralnih koeficientov in energijo ter njihove prve in druge odvode. Končni vektor značilk ima tako 39 elementov.

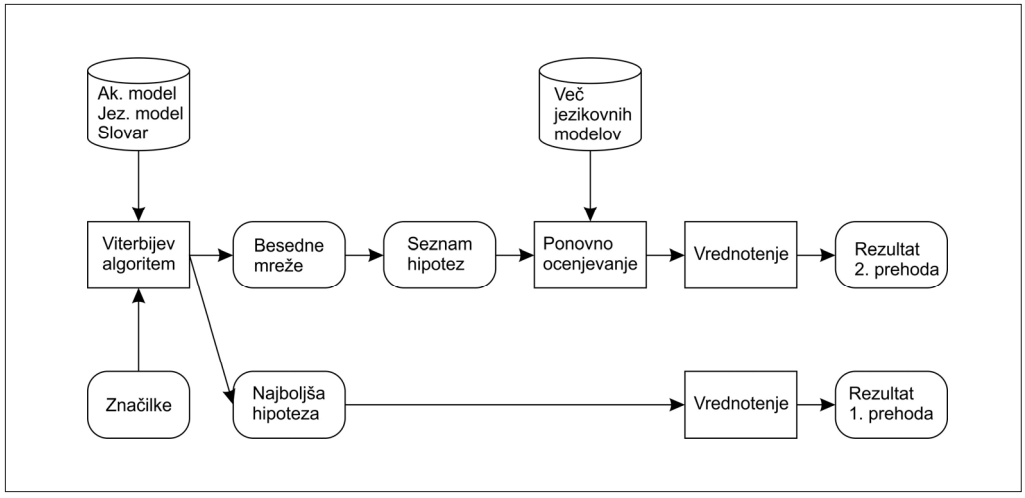
Postopek učenja akustičnih modelov poteka v treh korakih, pri čemer se postopoma izboljšuje kakovost akustičnih modelov. Kot osnovno akustično enoto smo uporabili grafeme, saj so predhodne analize pokazale, da je tako mogoče učiti kakovostne akustične modele (Žgank & Sepesy Maučec, 2010). V nadaljevanju bomo za akustične modele uporabljali poimenovanje fonem in trifon, kljub temu da je bila osnovna akustična enota grafem. V učnem setu smo uporabili 24 oddaj. V prvem koraku izvedemo inicializacijo parametrov akustičnih modelov z globalnimi vrednostmi. Temu sledi več ponovitev učnega Baum-Welchevega algoritma. S tako naučenimi akustičnimi modeli izvedemo prisilno poravnavo transkripcij, s katero se izboljša njihova kakovost. Sledi drugi korak s ponovim učenjem akustičnih modelov od začetka, vendar tokrat z izboljšanimi transkripcijami. Inicializacija vrednosti parametrov prikritih modelov Markova se tokrat izvrši ločeno za vsak fonem posebej.

Akustični modeli, naučeni v drugem koraku, služijo za izhodišče tretjega koraka, v katerem se najprej tvorijo kontekstno odvisni akustični modeli – trifoni, pri katerih upoštevamo predhodni in naslednji fonem. Posledično zelo naraste število prostih parametrov akustičnih modelov, ki jih je treba oceniti med postopkom učenja. Zato uporabimo postopek združevanja z odločitvenim drevesom, pri čemer na podlagi podatkovne metrike združimo stanja oz. celotne modele, ki so med seboj dovolj podobni. Odločitveno drevo zgradimo na podlagi fonetičnih razredov, ki so bili v predstavljenem eksperimentu tvorjeni s podatkovno vodeno metodo na podlagi matrike zamenjav fonemov. Akustični modeli, združeni z odločitvenim drevesom, so bili izhodišče za zadnji korak učenja, v katerem se je število Gaussovih porazdelitvenih funkcij verjetnosti korakoma povečalo do 16 na stanje. Takšni akustični modeli so bili uporabljeni za vrednotenje samodejnega razpoznavalnika govora.

Pred razpoznavanjem govora smo zgradili vrsto jezikovnih modelov, ki smo jih primerjali glede na uspešnost v razpoznavalniku. Tako smo najprej definirali različne velikosti slovarjev od 60.000 (60 k) do 300.000 (300 k) besed. Preizkušali smo dva načina gradnje slovarjev. V prvem načinu (FP) smo slovar gradili tako, da smo mu dodajali besede v vrstnem redu, ki ga je določala njihova pogostost v korpusu FidaPLUS. Ko smo dosegli želeno velikost slovarja, smo v slovar dodali še vse besede, ki so se pojavile z enako frekvenco kot nazadnje dodana beseda. V drugem načinu gradnje slovarja (BNSI+FP) smo najprej v slovar vključili vse besede iz govorne učne množice BNSI, nato smo dodajali besede iz besedilnega korpusa BNSI (iNews) in nazadnje besede iz korpusa FidaPLUS.

Pred gradnjo jezikovnih modelov smo pogledali deleže besed zunaj slovarja, ki se pojavijo na testni množici BNSI glede na oba načina gradnje slovarja. Po pregledu rezultatov smo se odločili, da bomo jezikovne modele gradili le na slovarjih, sestavljenih po prvem načinu (FP).

Nato smo zgradili standardne bigramske, trigramske in štirigramske modele. Pri tem smo uporabljali tako glajenje Good-Turing kot Knesser-Ney. Raziskali smo tudi vpliv velikosti učne množice, zato smo kot učno množico enkrat uporabili celotni korpus FidaPLUS, drugič pa le njegov del – približno devet odstotkov.



Slika 3: **Blokovna shema poteka razpoznavanja**

Splošna shema našega eksperimentalnega sistema je podana na sliki 3. Iskalni algoritem v prvem prehodu je sinhroni Viterbijev algoritem s snopovnim omejevanjem, ki je implementiran v orodju HDecode. Za vsak vhodni akustični segment nam algoritem vrne najboljšo hipotezo in besedno mrežo, ki pomeni iskalni prostor algoritma ob koncu segmenta. Najboljšo hipotezo določimo po uteženem razmerju med verjetnostima, dobljenima z akustičnim in jezikovnim modelom. Za določitev optimalnih vrednosti uteži smo uporabili rezultate razpoznavanja na razvojni množici BNSI.

Kadar neposredno vrednotimo uspešnost razpoznavanja na najboljši hipotezi, dobimo rezultate prvega prehoda. Na podlagi teh rezultatov smo se odločili, katere sisteme prvega prehoda (glede na različne jezikovne modele) bomo uporabili v dvoprehodnem algoritmu.

Pred drugim prehodom razpoznavanja besedne mreže pretvorimo v sezname sto najboljših hipotez, ki jih lahko razberemo iz njih. V nekaterih segmentih je to število tudi manjše, ker ni mogoče tvoriti takšnega števila hipotez. Hipoteze nato oblikoskladenjsko označimo z označevalnikom Obeliks. V naslednjem koraku oznake poenostavimo tako, da vsebujejo le podatek o besedni vrsti, spolu, sklonu, številu in osebi razpoznane besede.

V drugem prehodu hipoteze ponovno ovrednotimo z novimi jezikovnimi modeli. Teh modelov je sedaj lahko tudi več. Podobno kot pri prvem prehodu utežimo verjetnosti, dobljene s posameznimi modeli. Pri tem je treba ponovno uporabiti razvojno množico za iskanje optimalnih vrednosti uteži. Kot končni rezultat algoritem vrne hipotezo, ki ima po drugem prehodu največjo verjetnost.

Za vrednotenje označenih hipotez v drugem prehodu smo zgradili modele oznak MSD. Kot učno množico smo uporabili korpus ssj500k, v katerem smo oznake poenostavili na enak način kot v označenih hipotezah razpoznavalnika.

5 REZULTATI

V Donaj & Kačič (2012) smo že predstavili vpliv velikosti slovarja na delež besed OOV na testni množici BNSI. Tam uporabljeni slovarji so bili grajeni le glede na korpus FidaPLUS. Tabela 2 podaja k temu še rezultate OOV, kadar gradimo slovarje enakih velikosti po drugem načinu (BNSI + FP).

Tabela 2: **Delež besed OOV glede na način gradnje slovarja in njegovo velikost**

Velikost slovarja	Prvi način (FP)	Drugi način (BNSI + FP)
60 k	6,94	5,09
100 k	3,44	3,23
200 k	1,64	2,08
300 k	1,02	1,44

Iz rezultatov vidimo, je pri manjših velikostih slovarja bolj ugodno upoštevati najprej tekstovni korpus BNSI, pri večjih slovarjih pa je položaj ravno nasproten. Manjši delež besed zunaj slovarja dobimo, ko uporabljamo samo korpus FidaPLUS. Vzrok za to vidimo v dejstvu, da se pri drugem načinu gradnje v slovar vključijo besede, ki se v učni množici in v

besedilnem delu BNSI pojavijo zelo redko, medtem ko se ne vključijo besede iz korpusa FidaPLUS, ki se v testni množici pojavijo pogosteje.

V tabeli 3 so predstavljeni rezultati razpoznavanja prvega prehoda pri različnih velikostih učne množice, različnih velikostih slovarja in pri uporabi bigramskih (2 g) in trigramskih (3 g) jezikovnih modelov. V tabeli 4 so podani tudi faktorji realnega časa, s katerimi je potekalo razpoznavanje v teh primerih.

Tabela 3: **Uspešnost razpoznavanja glede na velikost učne množice in jezikovni model**

Slovar	Red modela	9 % Fidaplus	100 % Fidaplus
60 k	2 g	64,05	66,09
60 k	3 g	65,80	69,23
300 k	2 g	68,11	70,77
300 k	3 g	69,90	74,33

Tabela 4: **Faktorji realnega časa pri razpoznavanju glede velikost učne množice in jezikovni model**

Slovar	Red modela	9 % Fidaplus	100 % Fidaplus
60 k	2 g	6,04	6,30
60 k	3 g	8,58	18,46
300 k	2 g	13,35	12,66
300 k	3 g	19,16	37,09

Iz podatkov v tabeli 3 lahko vidimo, da se pri povečanju učne množice, povečanju slovarja in povečanju reda modela opazno izboljša uspešnost razpoznavanja. Izboljšanje uspešnosti ob povečanju velikosti slovarja je v vseh primerih približno 4 do 5 odstotkov, kar je v velikostnem redu zmanjšanja besed OOV pri spremembi velikosti slovarja. Spremembe v uspešnosti ob povečanju reda modela iz bigramskega na trigramskega so odvisne od velikosti učne množice. Medtem ko sta pri uporabi manjše učne množice spremembi 1,75 in 1,79 odstotka, sta pri uporabi večje učne množice spremembi 3,14 in 3,56 odstotka. Iz podatkov v tabeli 4 je razvidno, da tako povečanje slovarja kot tudi zvišanje reda modela poveča časovno zahtevnost razpoznavanja govora. Pri povečanju slovarja se faktor realnega časa poveča za približno 2. Pri zvišanju reda modela pa je ta faktor različen glede na velikost učnega korpusa. V primeru uporabe celotnega korpusa se velikost faktorja poveča približno za 3. Pri uporabi manjšega korpusa je povečanje veliko manjše.

Na podlagi teh podatkov lahko sklepamo, da bi dodatno povečanje učne množice (npr. z uporabo korpusa Gidafida) še dodatno povečalo uspešnost razpoznavanja, ki bo bolj izrazito pri uporabi trigramskega modela.

V tabeli 5 so prikazani rezultati uspešnost razpoznavanja pri uporabi modelov z modificiranim glajenjem Knesser-Ney, ki sta ga predstavila Chen & Goodman (1999) in razlika v uspešnosti glede na ustrezni model z glajenjem Good-Turing.

Tabela 5: **Uspešnost razpoznavanja z modificiranim glajenjem Knesser-Ney**

Slovar	Red modela	Acc (KN)	Acc (KN) – Acc (GT)
60 k	2 g	66,15	+ 0,06
60 k	3 g	69,04	+ 0,19
300 k	2 g	70,71	– 0,06
300 k	3 g	74,12	– 0,21

Iz rezultatov vidimo, da so modeli z modificiranim glajenjem Knesser-Ney uspešnejši le pri manjših slovarjih, medtem ko so pri večjih slovarjih uspešnejši modeli z glajenjem Good-Turing. V obeh primerih so razlike le majhne.

V vseh poskusih smo dobili besedne mreže, s katerimi bi lahko nadaljevali razpoznavanje v drugem prehodu, vendar smo se omejili le na rezultate, ki smo jih dobili pri slovarju 300 k in glajenjem GT. Prva različica tega algoritma je bila predstavljena v Donaj & Kačič (2012). Pokazano je bilo, da lahko z uporabo dvoprehodnega dosežemo primerljive uspešnosti ob bistveno krajšem času razpoznavanja. Prav tako je bilo pokazano, da uporabi trigramskih in štirigramskih modelov v drugem prehodu dajeta enake rezultate.

Za vrednotenje hipotez v drugem prehodu smo uporabili dva jezikovna modela. Prvi je standardni besedni trigramski model, drugi pa je trigramski model oznak MSD. V tabeli 6 so predstavljeni rezultati dvoprehodnega algoritma za istočasno vrednotenje z trigramskim modelom besed in trigramskim modelom oznak MSD.

Tabela 6: **Rezultati v dvoprehodnem algoritmu**

Prvi prehod	74,33 %
Drugi prehod	74,85 %
Sprememba	0,52 %

Iz podatkov vidimo, da smo lahko s pomočjo preprostega modela oznak MSD izboljšali uspešnost razpoznavanja za 0,52 odstotka.

6 SKLEP

V prispevku smo predstavili osnovne pojme s področja samodejnega razpoznavanja govora in govorne ter jezikovne vire za slovenščino, ki jih uporabljamo na tem področju. Razpoznavanji tekočega in spontanega govora sta nalogi z veliko prostora za vpeljevanje izboljšav tako v akustičnem kot v jezikovnem modeliranju. Predstavljeni rezultati kažejo na pomembnost ustreznih jezikovnih virov. Tukaj sta pomembna tako obseg virov kot tudi njihova dodatno obogatena vsebina, kot sta lematizacija in oblikoskladenjsko označevanje besedila.

Prestavljeni rezultati uporabe oblikoskladenjskih oznak v jezikovnem modeliranju pomenijo le začetek dela na tem področju. Zaradi svoje kompleksnosti v kombinaciji z uveljavljenimi jezikovnimi modeli ponujajo ti modeli veliko možnosti za teoretične in praktične raziskave.

Naše nadaljnje raziskave na področju ASR bodo usmerjene tudi v uporabo novih virov za izdelavo modelov, kot sta npr. korpusa Gigafida in GOS, kot tudi na izboljšano uporabo razpoložljivih informacij v korpusih.

Medtem ko je samodejno razpoznavanje govora že uporabno v omejenih domenah z majhnimi slovarji besed, pa trenutni rezultati razpoznavanja tekočega govora z velikim slovarjem besed še niso zadovoljivi za praktične aplikacije. Zato bodo še potrebne raziskave, ki bodo usmerjene tako v izboljšanje uspešnosti kot tudi hitrosti razpoznavanja govora. Zaradi težavnosti razpoznavanja slovenskega govora bo potrebno tudi nadaljnje delo na področju izdelave jezikovnih virov slovenščine. Le s takšnim celovitim pristopom bomo lahko zagotovili stik našega jezika s sodobnimi trendi v informacijsko-komunikacijskih tehnologijah.

LITERATURA

[1] Arhar, Š. & Gorjanc, V. (2007). Korpus FidaPLUS: nova generacija slovenskega referenčnega korpusa. *Jezik in slovstvo*, 52(2), 95–110.

[2] Arhar, Š. (2009). Učni korpus SSJ in leksikon besednih oblik za slovenščino. *Jezik in slovstvo*, 54(3–4), 43–56.

[3] Arhar Holdt, Š., Kosem, I. & Logar Berginc, N. (2012). Izdelava korpusa Gigafida in njegovega spletnega vmesnika. *Zbornik Osme konference Jezikovne tehnologije*, Ljubljana, Slovenija, 16–21.

[4] Aubert, X. L. (2002). An overview of decoding techniques for large vocabulary continuous speech recognition. *Computer speech & language*, 16(1), 89–114.

[5] Chen, S. F. & Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer speech & language*, 13(4), 359–393.

[6] Dobrišek, S., Vesnice, B., Žganec Gros, J. & Mihelič, F. (2006). Uporaba kanoničega govornega akustičnega modela za prilagajanje prostora govornih akustičnih značilk. *Jezikovne tehnologije: zbornik 9. mednarodne multikonference Informacijska družba*, Ljubljana, Slovenija, 89–92.

[7] Donaj, G. & Kačič, Z. (2012). Širjenje slovarja in dvoprehodni algoritem v razpoznavalniku tekočega govora UMB Broadcast News. *Zbornik Osme konference Jezikovne tehnologije*, Ljubljana, Slovenija, 48–51.

[8] Erjavec, T. & Krek, S. (2008). Oblikoskladenjske specifikacije in označeni korpusi JOS. *Zbornik Šeste konference Jezikovne tehnologije*, Ljubljana, Slovenija, 49–53.

[9] Good, I. J. (1953). The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3–4), 237–264.

[10] Grčar, M., Krek, S. & Dobrovoljc, K. (2012). Obeliks: statistični oblikoskladenjski označevalnik in lematizator za slovenski jezik. *Zbornik Osme konference Jezikovne tehnologije*, Ljubljana, Slovenija, 89–94.

[11] Kačič, Z. & Kaiser, J. (1998). Development of Slovenian SpeechDat database. *First International Conference on Language Resources and Evaluation, Workshop on speech database development for Central and Eastern European languages,* Granada, Spain.

[12] Kačič, Z. (2002). Pomen združevanja raziskovalnih potencialov pri preseganju jezikovnih pregrad v okviru jezikovnih tehnologij naslednjih generacij. *Jezikovne tehnologije: zbornik konference*, Ljubljana, Slovenija, 111–115.

[13] Katz, S. M. (1987). Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on acoustics, speech and signal processing*, 35(3), 400– 401.

[14] Lee, A., Kawahara, T. & Doshita, S. (1998). An efficient two--pass search algorithm using word trellis index. *Proceeding of the 5th International Conference on Spoken Language Processing*, Sydney, Australia.

[15] Mihelič, F., Žganec Gros, J., Dobrišek, S., Žibert, J. & Pavešić, N. (2003). Spoken language resources at LUKS of the University of Ljubljana. *International journal of speech technology*, 6(3), 221–232.

[16] Rotovnik, T., Sepesy Maučec, M. & Kačič, Z. (2007). Large vocabulary continuous speech recognition of an Inflected language using stems and endings. *Speech communication*, 49(6), 437–452.

[17] Sepesy Maučec, M., Rotovnik, T., Kačič, Z. & Brest, J. (2009). Using data-driven subword units in language model of highly inflective Slovenian language. *International journal of pattern recognition artificial intelligence*, 23(2), 287–312.

[18] Stolcke, A., Zheng, J., Wang, W. & Abrash, V. (2011). SRILM at sixteen: Update and outlook. *Proceedings IEEE Automatic Speech Recognition and Understanding Workshop*.

[19] Young, S., Jansen, J., Odell, J., Ollason, D. & Woodland, P. (1996). *The HTK book*. Cambridge University.

[20] Žganec Gros, J., Mihelič, F. & Dobrišek, S. (2003). Govorne tehnologije: pridobivanje in pregled govornih zbirk za slovenski jezik. *Jezik in slovstvo*, 48(3–4), 47–59.

[21] Žgank, A., Kačič, Z. & Horvat, B. (2002). Preliminary evaluation of Slovenian mobile database PoliDat. *Third international conference on language resources and evaluation*, Las Palmas de Grand Canaria, Spain, 564–568.
[22] Žgank, A., Rotovnik, T., Grašič, M., Kos, M., Vlaj, D. & Kačič, Z. (2006). SloParl – Slovenian parlamentary speech and text corpus for large vocabulary continuous speech recognition. *Ninth international conference on spoken language processing*, Pittsburgh, PA, USA, 197–200.
[23] Žgank, A., Rotovnik, T. & Sepesy Maučec, M. (2008). Slovenian spontaneous speech recognition and acoustic modeling of filled pauses and onomatopoeas. *WSEAS transactions on signal processing,* 4(7), 388–39.
[24] Žgank, A., Sepesy Maučec, M. (2010). Razpoznavalnik tekočega govora UMB Broadcast News 2010: nadgradnja akustičnih in jezikovnih modelov. *Zbornik Sedme konference Jezikovne tehnologije*, Ljubljana, Slovenija, 28–31.
[25] Žgank, A., Verdonik, D., Zögling Markuš, A. & Kačič, Z. (2005). BNSI Slovenian broadcast news database – speech and text corpus. *9th European conference on speech communication and technology*, Lisbon, Portugal, 1537–1540.
[26] Žibert, J. & Mihelič, F. (2004). Development of Slovenian broadcast news speech database. *Fourth International Conference on Language Resources and Evaluation*, Lisbon, Portugal, 2095–2098.

▪

Gregor Donaj je diplomiral iz elektrotehnike na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru in iz matematike na Fakulteti za naravoslovje in matematiko Univerze v Mariboru. Trenutno je doktorski študent in zaposlen kot mladi raziskovalec na Fakulteti za elektrotehniko, računalništvo in informatiko. Raziskovalno se ukvarja z jezikovnim modeliranjem za avtomatsko razpoznavanje govora.

▪

Andrej Žgank je leta 2003 doktoriral na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Na tej fakulteti je tudi zaposlen kot izredni profesor za področje telekomunikacije. Njegovo raziskovalno področje obsega večjezičnost, križnojezično razpoznavanje govora, akustično modeliranje pri razpoznavalniku govora z velikim slovarjem in gradnja jezikovnih virov.

▪

Mirjam Sepesy Maučec je izredna profesorica za področje telekomunikacije na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Ob pedagoškem delu je raziskovalno aktivna v številnih nacionalnih in mednarodnih projektih s področja jezikovnih tehnologij. Njeno raziskovalno področje obsega statistično jezikovno modeliranje in strojno prevajanje.

Sodobne prevajalske tehnologije in prihodnost prevajalskega poklica

Špela Vintar, Univerza v Ljubljani, Filozofska fakulteta, Oddelek za prevajalstvo
spela.vintar@ff.uni-lj.si

Izvleček

Prispevek pregledno predstavlja področje sodobnih prevajalskih tehnologij in njihovega vpliva na profesionalno prevajanje. V zadnjih letih se to namreč korenito spreminja pod vplivom vse cenejših in dostopnejših strojnih prevajalnikov, ki jih uporabljamo za izdelavo grobega prevoda, vloga prevajalca pa je »le« poprava takega prevoda. V prispevku predstavimo strojno prevajanje danes z najpomembnejšimi tehnologijami in sistemi, nato pa se posvetimo načinom, kako prevajalniki spreminjajo klasične delovne procese, poklicne profile, pojem kakovosti in cenovno politiko v prevajalstvu. V sklepnih odstavkih razmišljamo o prihodnosti prevajalskega poklica in predlagamo nekaj ukrepov, s katerimi bi se bilo dobro odzivati na razvojne trende.

Ključne besede: strojno prevajanje, popravljanje strojnih prevodov, prevajalske tehnologije, pomnilnik prevodov, profil prevajalca.

Abstract

Recent Trends in Translation Technologies and the Future of Professional Translation

The paper gives an overview of the field of translation technologies and their impact on professional translation services. Through the past decade this field has witnessed profound changes due to better and cheaper machine translation systems used to produce a raw translation, while the role of the human translator is reduced to postediting. The paper presents the state-of-the-art in machine translation technologies and systems, and then describes ways in which machine translation affects traditional workflows, professional profiles, the notion of quality and the pricing policy in translation services. We conclude with a discussion of the future of the translation business and suggest certain measures to meet the new challenges and react to technological trends.

Key words: machine translation (MT), post-editing machine translation, translation technologies, translation memory, translator profile.

1 UVOD

Razvoj spletnih prevajalnikov, kot sta Google Translate in Bing Translator, je dodobra posegel v razmišljanje in vedenje uporabnikov spleta. Kljub šalam na račun slabih prevodov so nam namreč danes s preprostim klikom na gumb prevedi dostopne vsebine, ki jih prej nismo mogli prebirati in ki jih po vsej verjetnosti tudi nikoli ne bi poslali v uradni človeški prevod. A do nedavnega je veljalo, da so — z nekaj redkimi izjemami — strojni prevajalniki namenjeni običajnih uporabnikom, ki z njihovo pomočjo dostopajo do v njim nerazumljivih jezikih napisanih informacij, svet računalniških orodij za prevajalce pa se je vrtel okrog pomnilnikov prevodov.

V zadnjih letih sta se tradicionalno ločeni veji strojnega in računalniško podprtega prevajanja močno zbližali in celo prepletli, kar je korenito spremenilo prevajalski proces, s tem pa tudi poklicni profil prevajalca.[1] Obenem so se spremenili tudi prevajalski projekti, saj vse pogosteje prevajalci nimajo več opravka z besedili, temveč z jezikovnimi nizi, seznami besed in besednih zvez brez sobesedila, ki se v okviru ciljne aplikacije dinamično sestavljajo v vsebine.

V prispevku najprej opišemo stanje prevajalskih tehnologij in njihovo razširjenost v praksi, pri čemer začnemo s strojnim prevajanjem in nadaljujemo s prevajalskimi namizji, ki klasični pomnilnik prevodov kombinirajo s strojnim prevajalnikom. Nato spregovorimo o popravljanju strojnih prevodov kot novemu tipu prevajalske naloge in predstavimo nekaj raziskav, ki se ukvarjajo z vprašanji učinkovitosti in kakovosti pri takem tipu prevajanja. Nazadnje prispevek poda vizijo nadaljnjega razvoja področja prevajalskih tehnologij in prevajalskega poklica.

2 STROJNO PREVAJANJE DANES

Če bi želeli celovito ponoviti zgodovino razvoja strojnih prevajalnikov od petdesetih let prejšnjega stoletja

[1] V prispevku dosledno uporabljamo moško obliko poklica prevajalec, pri čemer mislimo na prevajalke in prevajalce.

do danes, bi krepko presegli okvire tega članka. Ponovimo le, da so pri razvoju do konca osemdesetih let prejšnjega stoletja prevladovali na pravilih temelječi pristopi, med katere kronološko po vrsti prištevamo neposredni pristop, vmesni jezik (interlingua) in transferni pristop, nato pa je konec osemdesetih let raziskovalna skupina IBM razvila statistični algoritem, ki je iz vzporednega korpusa črpal prav vse podatke in se torej ni opiral na slovarje in slovnice (Ney, 2005; Hutchins, 2007).

2.1 Statistično strojno prevajanje

V drobovju statističnega prevajalnika sta prevodni in jezikovni model. Za prevodni model algoritem potrebuje vzporedna besedila v obeh jezikih, se pravi izvirnike in prevode, ki jih je mogoče samodejno stavčno poravnati. Iz takšnega vzporednega korpusa besedil je za vsako besedo izvirnega jezika mogoče izluščiti niz najverjetnejših prevodnih ustreznic, in to brez da bi kar koli vedeli o obeh jezikih. Če si predstavljamo, da se v angleško-slovenskem vzporednem korpusu evropskih besedil v izvirniku nekajkrat pojavi beseda *fishing*, in če smo že v fazi predobdelave vsakemu angleškemu stavku določili njegov slovenski prevod, lahko domnevamo, da se bo v naboru teh slovenskih stavkov dosledno pojavljala beseda, ki je prevod za *fishing*, denimo *ribolov*. Vsa takšna sopojavljanja se zapišejo v prevodni model kot verjetnosti, da se bo določena beseda prevedla s ciljno besedo, algoritem pa na podoben način obdela tudi verjetnosti večbesednih enot (Och & Ney, 2004).

Google je v svojih rosnih letih kot vir vzporednih besedil uporabil dokumente Združenih narodov, kmalu pa so njihovi pajki v svoje mreže potegnili tudi dokumente drugih večjezičnih tvorb, kot je EU, različne obstoječe vzporedne korpuse in tudi večjezična spletišča, za katera zna pajek hitro ugotoviti, ali gre resnično za prevod ali zgolj za tujejezično prirejeno različico.

Če bi imel prevajalnik na voljo le prevodni model, bi se posamezne besede in besedne zveze sicer prevedle pravilno, a bi bila struktura ciljnega stavka še vedno tako rekoč identična izvirniku. Prav tako se prevajalnik samo na podlagi prevodnega modela težko odloča med različnimi oblikoslovnimi možnostmi prevoda: naj se *red* prevede kot *rdeč, rdeča, rdečim, rdečimi* …? Da bi bil torej ciljni stavek kar najbolj podoben običajnim slovnično pravilnim stavkom ciljnega jezika, ima prevajalnik na voljo še jezikovni model. Tudi ta se zgradi iz ogromnih količin besedil, le da je tu na voljo še bistveno več virov, saj zanj teoretično lahko uporabimo kar vse spletne strani v ustreznem jeziku. Jezikovni model beleži verjetnosti pojavitve besednih nizov, dolgih navadno dve do pet besed, in tako lahko prevajalnik hitro ugotovi, da je v slovenščini kombinacija *rdečih zastava* bistveno manj verjetna kot *rdeča zastava*.

Ob tem velja poudariti, da je statistično prevajanje natanko toliko dobro, kolikor sta dobra prevodni in jezikovni model. Pri tem ni pomembna le količina besedil, temveč tudi njihove kakovost, strokovnost, slog in terminološka doslednost; posledično lahko iz manjše količine visoko specializiranih besedil za določeno področje zgradimo boljši prevajalnik kot iz velike količine splošnih besedil.

2.2 Hibridni modeli

Tako na pravilih temelječe prevajanje (RBMT) kot statistično strojno prevajanje (SMT) imata svoje pomanjkljivosti, ki jih je težko rešiti v okviru posamezne od obeh razvojnih vej. Tako raziskovalci ugotavljajo (Uszkoreit, 2009), da so običajne težave sistemov RBMT predvsem:

- nezadovoljivo razdvoumljanje, izbira besedišča, slogovna in zvrstna ustreznost,
- nezadovoljivo ravnanje v primeru vrzeli v besedišču in slovničnih pravilih.

Razvoj sistemov SMT je sicer bistveno cenejši, vendar so tudi ti s samo statističnimi algoritmi prišli do težko premostljivih ovir:

- nezadovoljiva obravnava vseh slovničnih pojavov, ki presegajo okvir posamezne fraze: svobodni besedni red, oddaljene slovnične odvisnosti, elipse, kompleksne slovnične strukture itd.,
- nezadovoljivo reševanje vrzeli v učnih podatkih.

Tako ni presenetljiva misel, da bi bili sistemi RBMT boljši, če bi upoštevali verjetnost posameznih jezikovnih enot, in sistemi SMT boljši, če bi poleg verjetnostnih modelov uporabljali še slovnična pravila. Hibridni sistemi se danes razvijajo na oba omenjena načina, se pravi izhajajoč iz RBMT z dodajanjem statistike in izhajajoč iz SMT z dodajanjem pravil, številne raziskave pa so bile opravljene v okviru evropskih projektov Euromatrix in Euromatrix Plus (Eisele idr., 2008).[2]

Morda je z vidika profesionalnega prevajanja še najpomembnejša novost, da so hibridne sisteme

[2] http://www.euromatrix.net in http://www.euromatrixplus.net.

začeli ponujati tudi številni komercialni ponudniki strojnega prevajanja, denimo Asia Online, Systran, LinguaSys, ti pa uporabnikom obenem ponujajo tudi prilagajanje sistema njihovim potrebam in besedilom. To je nadvse pomembno, saj prevajalska agencija za svoje delo večinoma ne more in ne sme uporabljati Googlovega prevajalnika. Tudi če bi jo namreč zadovoljila kakovost Googlovih prevodov, si spletni prevajalnik shranjuje vsa besedila, kar za večino naročnikov pomeni kršitev varovanja osebnih podatkov in poslovnih skrivnosti.

Po drugi strani komercialni ponudniki uporabnikom zagotavljajo, da bodo prevajalnik »naučili« na njihovih besedilih; tako nastali sistem se ob dovolj veliki količini učnih podatkov dobro odreže pri prevajanju strokovne terminologije in ustaljenih fraz, ker pa ima za osnovo še vedno statistični sistem, rado prihaja do napak pri daljših in kompleksnejših povedih.

3 INTEGRACIJA STROJNIH PREVAJALNIKOV V ORODJA ZA RAČUNALNIŠKO PODPRTO PREVAJANJE

Orodja za računalniško podprto prevajanje (Computer-Aided Translation, CAT) so se razširila v devetdesetih letih in so danes osnovna programska oprema vsakega poklicnega prevajalca, ki se redno srečuje s tehničnimi prevodi. Glavna komponenta takšnega programa – pravimo jim tudi prevajalska namizja – je pomnilnik prevodov, ki prevajalcu omogoča shranjevanje že prevedenih enot in njihovo ponovno uporabo pri nadaljnjih prevajalskih projektih. Gre za podatkovno zbirko prevodnih enot, navadno povedi ali krajših delov besedila, ki so v izvirniku in prevodu shranjeni v pomnilnik in so ob morebitni ponovitvi enakega ali zelo podobnega dela besedila na razpolago za ponovno uporabo.

Po ocenah zadnje večje raziskave o rabi prevajalskih tehnologij, v kateri je sodelovalo prek 500 prevajalcev iz 52 držav sveta (Torres Dominguez, 2012), jih okrog 70 odstotkov uporablja prevajalska namizja, z njimi pa prevedejo med 75 in 99 odstotki vseh prevajalskih projektov. Najbolj razširjena orodja so SDL Trados, MemoQ, Wordfast, DejaVu, OmegaT in SDLX. Prevajalsko namizje je nepogrešljivo predvsem pri prevajanju ponovljivih in formaliziranih besedil, kot so navodila za uporabo, tehnična dokumentacija proizvodov, pravna besedila, vmesniki programske opreme različnih elektronskih naprav ipd. Pri tovrstnih besedilih se namreč pojavljajo tipične strukture (*Če želite vključiti X, pritisnite tipko Y*), ki se v enaki ali podobni obliki ponavljajo bodisi v okviru istega besedila bodisi v naslednjem sorodnem projektu.

Zaradi prihranka časa, ki ga prinaša opisana reciklaža prevodov, se je spremenilo tudi obračunavanje prevajalskih storitev, pri katerih se uporablja pomnilnik prevodov. Splošno razširjeno pravilo je, da se za besedilne segmente, za katere je program v bazi našel identičen že prevedeni segment, zaračuna 30 odstotkov celotne cene, za delne oz. meglene zadetke, ki so izvirniku podobni od 70 do 95 odstotkov, se zaračuna 70 odstotkov cene, za dele, pri katerih v bazi ni uporabnega zadetka, pa naročnik plača polno ceno prevoda.

V nekaterih primerih se uporablja še bolj podrobno razdeljena tarifna shema, včasih pa naročniki prevajalcem izrecno prepovedo spreminjanje stoodstotnih zadetkov iz baze. To še posebej velja takrat, kadar pomnilnik prevodov vsebuje uradno potrjene in pregledane prevode, ki zagotavljajo terminološko in slogovno doslednost. Prevajalska namizja so opremljena s funkcijo, ki primerja novo besedilo z obstoječo bazo in pomaga pri izdelavi predračuna za prevod.

Skoraj vsa od prej omenjenih prevajalskih namizij danes omogočajo vključitev strojnega prevajalnika v namizje, tako da prevajalec lahko uporablja tako pomnilnik prevodov kot strojni prevajalnik v skupnem okolju. Številna orodja omogočajo integracijo plačljivega vtičnika za Google Translate API, podpirajo pa tudi uporabo drugih, ne nujno spletnih, prevajalnikov.

Takšno rešitev od letošnjega leta uporablja tudi največja prevajalska služba na svetu, Generalni direktorat za prevajanje Evropske komisije (DGT), ki zaposluje okrog 2.500 prevajalcev in letno prevede prek osem milijonov strani. Delovno okolje tamkajšnjih prevajalcev je SDL Trados Studio, prek katerega prevajalci dostopajo do zadetkov iz skupnega pomnilnika prevodov Euramis. Kadar niti Euramis niti druge interne baze Evropske komisije ne vsebujejo enakega ali podobnega segmenta, se ta prevede s prav za potrebe DGT razvitim statističnim strojnim prevajalnikom MT@EC. Da prevajalec ve, da ima pred seboj strojni prevod, je ta v okolju SDL Studio označen s sivo barvo. Ko prevajalec pregleda, popravi in potrdi strojni prevod, se ta shrani v pomnilnik prevodov skupaj z drugimi (človeškimi) prevodi.

Ker kakovost strojnega prevajanja za različne jezikovne pare zelo niha, so na DGT-ju pred kratkim med prevajalci izvedli raziskavo o vtisih pri delu s strojnim prevajalnikom (Leal Fontes, 2013). Ta je pokazala, da strojni prevajalnik uporablja že skoraj tri četrtine prevajalcev, od tega pa jih slaba polovica meni, da je strojni prevod v 75 odstotkih primerov zelo uporaben z manjšimi popravki. Slovenski prevajalci so za jezikovni par angleščina – slovenščina podali nekoliko manj navdušene, a še vedno zadovoljne odzive: strojni prevod se jim je zdel uporaben z manjšimi popravki v približno 50 odstotkih primerov. Najslabše se MT@EC odreže pri aglutinirajočih jezikih, kot je madžarščina, ter pri jezikih baltskih držav.

4 POPRAVLJANJE STROJNIH PREVODOV

Če smo v prejšnjem razdelku opisali kombinacijo »klasičnih« prevajalskih namizij in strojnega prevajanja, je naslednji korak pričakovan: v mnogih prevajalskih okoljih postopoma prehajajo na način prevajanja, pri katerem besedilo najprej prevedemo strojno, nato pa prevajalec besedilo popravi do želene stopnje kakovosti. Za to delo v angleščini uporabljajo kratico PEMT (Post-Editing Machine Translation), gre pa za opravilo, ki se v marsičem zelo razlikuje od tradicionalnega prevajanja.

Pravzaprav ideja ni nova, saj so v vseh okoljih, v katerih že dolgo prevajajo računalniki, morali rezultat pregledati in izboljšati prevajalci ali tehnični pisci. Prav tako je v okoljih, v katerih strojno prevajanje uporabljajo že dlje, pogosta uporaba t. i. nadzorovanega jezika, ki naj bi s pomočjo omejenega nabora slovničnih struktur in besedišča zagotavljal boljši strojni prevod. Novost pomeni dejstvo, da so postali v zadnjih nekaj letih prevajalniki na eni strani dovolj dostopni, na drugi pa dovolj kakovostni, da je njihova uporaba smiselna za vse širši krog uporabnikov.

Kljub temu da kakovost prevodov močno niha glede na uporabljeni prevajalnik in glede na jezikovni par, pa številne raziskave (Guerberof, 2009; Specia, 2011) kažejo povečanje produktivnosti prav za vse jezikovne pare, in sicer se to giblje od 42 za kitajščino do kar 130 odstotkov za francoščino. Za slovenščino še ni primerljivih rezultatov, so pa v teku raziskave, ki se ukvarjajo tako z vprašanjem produktivnosti kot kakovosti.

Odzivi prevajalcev na novo obliko dela, ki pravzaprav ni več prevajanje, so različni, a v glavnem negativni. Na forumu prevajalskega portala ProZ.com se je nedolgo tega odvijala razprava,[3] v kateri so bila prevladujoča stališča v zvezi s popravljanjem strojnih prevodov zelo odklonilna in so vsebovala izjave:

> »Osebno zavračam popravljanje strojnih prevodov.«
>
> »Enako. Nikakor ne nameravam učiti stroja, kako naj me nadomesti.«
>
> »To ni delo, ki bi bilo primerno za mojstra, zato takšna naročila vselej z gnusom zavrnem. Dobro bi bilo, ko bi tudi drugi prevajalci ustrezno spoštovali svoj poklic.«

Tradicionalna podoba prevajalskega poklica vsebuje ustvarjalnost kot pomembno, če ne že kar najpomembnejšo sestavino poklicnega profila. Razumljivo je torej, da so – še posebno starejši – prevajalci ogorčeni, ko od naročnika dobijo strojno prevedeno besedilo, polno napak in nerodnih besednih zvez, skupaj s pričakovanjem, da bodo za majhen denar iz njega pričarali kakovostno in za objavo primerno besedilo. A tehnološki razvoj gre svojo pot in danes so posebej zanje razviti strojni prevajalniki prisotni že tudi pri slovenskih prevajalskih agencijah.

Previdnost je potrebna pri obračunavanju tovrstnega dela, saj je ena od zgodnejših raziskav popravljanje strojnega prevoda primerjala s kakovostnimi 80- ali 90-odstotnimi meglenimi zadetki (O'Brien, 2007). To je seveda zelo optimistična ocena, ki utegne biti zavajajoča tudi za naročnike; ti potem pričakujejo, da bodo za popravljeni prevod plačali le okrog 40 odstotkov polne cene. Prevajalci, ki imajo s popravljanjem strojnih prevodov izkušnje, svetujejo, da pred začetkom dela, še bolje pa pred dogovorom o prevzemu naročila, izvedemo preskus, delo pa nato obračunavamo po urni postavki.

V uvodnem odstavku tega razdelka smo popravljanje strojnih prevodov opredelili kot dejavnost, pri kateri prevajalec strojno prevedeno besedilo popravi do želene kakovosti. Pojem kakovosti namreč v razponu med popolnoma avtomatiziranim strojnim prevodom na eni strani in slogovno ter strokovno pregledanim profesionalnim človeškim prevodom na drugi strani postane gibljiv: Koliko kakovosti potrebuje naročnik in koliko kakovosti je pripravljen plačati?

Za lažje razumevanje gibljive kakovosti spomnimo, da s strojnimi prevajalniki danes pogosto preva-

[3] http://www.proz.com/forum/money_matters/215371-rates_for_post_editing_machine_translation_texts-page2.html; izbrane izjave prevedla avtorica članka.

jajo besedila, ki jih prej verjetno sploh ne bi prevajali. Tako si pri brskanju po spletu ogledujemo nerodno prevedene spletne strani, a nam grobi prevod zadošča za razumevanje in verjetno nikoli ne bi najeli prevajalca oziroma popravljalca, naj ga izboljša. Za druga besedila, denimo obsežna tehnična navodila, ki niso namenjena širši publiki, ampak le izbranemu krogu strokovnih uporabnikov, je morda dovolj površna poprava (*light post-editing*), ki zagotovi razumljivost in odpravi hujše slovnične napake. Besedila, namenjena objavi ali širši publiki, zahtevajo polno popravo (*full post-editing*), pri kateri prevajalec zagotovi kakovost, ki po jezikovni, slogovni, terminološki, oblikovni in tehnični plati v ničemer ne odstopa od človeškega prevoda.

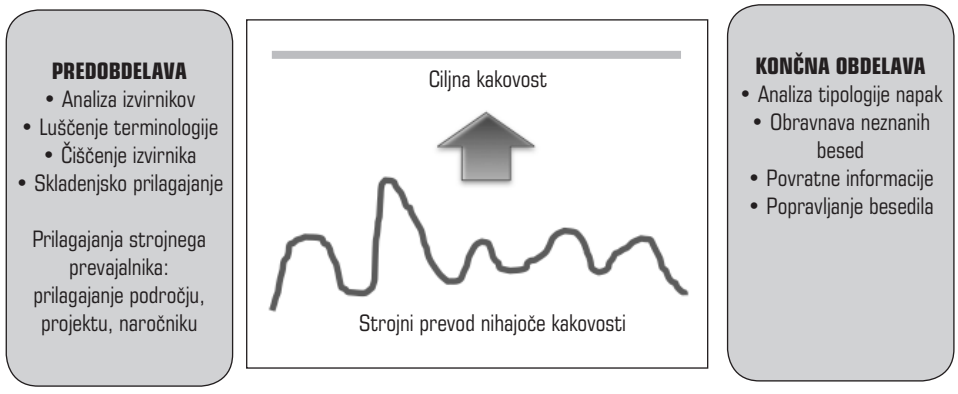
Za popravljanje strojnih prevodov je v okviru Googlovega prevajalnika na voljo okolje Translator Toolkit; Aziz idr. (2012) so razvili tudi orodje PET, sicer pa je za to mogoče uporabiti prevajalsko namizje, kot je denimo SDL Trados Studio ali memoQ.

5 NOVI MODEL PREVAJALSKEGA PROCESA

V tradicionalnem toku prevajalskega procesa igra od trenutka, ko izvirno besedilo zapusti naročnika, pa do trenutka, ko naročnik prejme ciljno besedilo in storitev obračunamo, glavno vlogo prevajalec. Ne glede na to, da ta proces glede na vrsto prevajanja pogosto zajema druge akterje (lektorje, terminologe, urednike, pravne redaktorje idr.), je v jedru prevajalske storitve še vedno prevajalec.

Tudi programi s pomnilnikom prevodov, ki so pred dobrim desetletjem zavzeli trg profesionalnega prevajanja in krepko spremenili način dela, niso bistveno posegli v obseg človekove vloge pri nastajanju ciljnega besedila – zadetke iz pomnilnika prevodov je prav tako nekoč moral nekdo prevesti. Morda je zanimivo, da so bile tedanje reakcije prevajalcev na pojav orodij, katerih glavni namen je bil recikliranje starih prevodov, prav tako odklonilne in čustvene kot današnje na strojno prevajanje.

Vsekakor se z vse boljšimi prevajalniki širi njihova uporaba v profesionalnem prevajanju, s tem pa se spreminja tudi vloga prevajalca. V novem modelu prevajalskega procesa, ki ga ponazarja slika 1, je v središču prevajalnik, vijugasta črta pa poudarja dejstvo, da je kakovost strojnega prevoda odvisna od besedilnega tipa, sloga in slovničnih lastnosti izvirnika. Še preden besedilo predamo prevajalniku, se izvedejo različni postopki predobdelave, ki skušajo besedilo čim bolje pripraviti na računalniško obdelavo. Tako je – vsaj pri večjih prevajalskih projektih – smiselno vnaprej izluščiti terminologijo in izdelati projektni glosar, iz besedila odstraniti elemente, ki niso jezikovni ali ki bi utegnili otežiti prevajanje (imena, simbole, formule itd.), včasih besedila tudi skladenjsko prilagodimo v smislu poenostavljanja stavčnih struktur, krajšanja povedi, izogibanja dvoumnim slovničnim oblikam ipd. Prav tako je mogoče prilagoditi prevajalnik: pri statističnih sistemih, ki gradijo prevodni model iz vzporednih besedil, lahko uporabimo pomnilnike prevodov določenega naročnika ali področja, številni prevajalniki pa omogočajo tudi vnos področnih glosarjev in terminoloških baz. Prav tako lahko prilagodimo obravnavo neznanih besed, imen in drugih specifičnih elementov.



Slika 1: **Model prevajalskega procesa** (prir. po Vashee, 2011)

Vložek na strani vhoda je pomemben in lahko bistveno vpliva na rezultat. Kaj se zgodi po samem prevajanju, je odvisno od želene oziroma dogovorjene ravni kakovosti, a v vsakem primeru bi morala slediti analiza napak, saj se jim v prihodnje morda lahko izognemo bodisi z izboljšavami prevajalnika bodisi z boljšo predpripravo besedila. V skrajnem primeru z analizo napak ugotovimo tudi, da se za določeni tip besedila strojno prevajanje s popravljanjem ne splača in da je zanj bolje uporabiti klasični način prevajanja.

Ob razmišljanju o prihodnosti prevajalskega poklica se neizogibno postavlja vprašanje, ali bodo prevajalci čez čas sploh še potrebni. V prizadevanju za zniževanje stroškov namreč naročniki polagajo velike upe v strojne prevajalnike, za popravljanje samodejno prevedenih besedil pa ne uporabljajo nujno prevajalcev, temveč tudi druge (cenejše) osebe z znanjem ciljnega jezika. Da tako ne moremo pričakovati prevodov, ki bi bili ne le jezikovno dovršeni, ampak tudi kulturno in slogovno ustrezni za ciljno publiko, najbrž ni treba posebej poudarjati.

Po drugi strani pa je za mnoge naročnike in za določene tipe besedil kakovost še kako pomembna, poleg tega s tehnološkim razvojem postajajo tehnična oziroma dokumentacijska besedila (s tem mislimo na navodila za uporabo v najširšem smislu) vse bolj kompleksna. Za vsebinsko, kulturno in strokovno funkcionalen prevod lahko poskrbi le visoko usposobljen prevajalec, ki si pomaga z ustreznimi računalniškimi pripomočki.

Pojavlja pa se še en – vse bolj zaželen – poklicni profil: prevajalec tehnolog je strokovnjak, ki ima poleg prevajalskih kompetenc še široko računalniško in jezikovnotehnološko znanje. Vanj spadajo ustvarjanje in upravljanje jezikovnih virov, kot so pomnilniki prevodov, korpusi, terminološke baze in leksikoni, testiranje in prilagajanje strojnih prevajalnikov, luščenje terminologije, pretvarjanje formatov, integracija različnih virov in orodij v enotno okolje, upravljanje strežniških in oblačnih programskih rešitev, v prihodnosti pa zagotovo še kaj. Tako prihodnost prevajalskega poklica zaradi strojnih prevajalnikov ni nujno črna, nedvomno pa bo vse bolj zaznamovana s tehnologijami.

6 SKLEP

V prispevku smo predstavili pregled sodobnih prevajalskih tehnologij, ki korenito spreminjajo delovne procese in razmerja v svetu profesionalnega prevajanja, vplivajo pa tudi na pojem kakovosti in cene teh storitev. V luči opisanih razvojnih tendenc se kaže več potreb: na eni strani bi bilo dobro posodobiti mehanizme, ki skušajo regulirati trg prevajalskih storitev. Edini tudi pri nas veljavni standard za zagotavljanje kakovosti prevajalskih storitev EN 15038 namreč nikjer ne omenja popravljanja strojnih prevodov kot ene od morebitnih kompetenc prevajalca, prav tako je še veliko nejasnosti pri pravičnem obračunavanju novih delovnih nalog. V razmerju naročnika in ponudnika storitev bi sčasoma pričakovali stratifikacijo prevajalskih storitev, pri čemer bi naročnik lahko izbiral med različnimi načini prevoda, seveda tudi različno ovrednotenimi.

Na drugi strani bi se na spreminjanje poklicnega profila morale ustrezno odzvati izobraževalne ustanove in v visokošolske programe prevajalstva še intenzivneje vključiti tehnološke vsebine. Na tretji strani pa opisani trendi pomenijo tudi grožnjo za kakovost prevodov, še posebno če tehnologije uporabljamo le kot sredstvo za zmanjševanje stroškov in brez razumevanja njihovih omejitev. S tega vidika je potrebno ozaveščanje vseh akterjev prevajalskega procesa in sistematično evalviranje tehnologij z analizami učinkovitosti in kakovosti.

7 VIRI IN LITERATURA

[1] Aziz, W., Castilho, S. & Specia, L. (2012). PET: a Tool for Post-editing and Assessing Machine Translation. In LREC, str. 3982–3987.

[2] Eisele, A., Federmann, C., Uszkoreit, H., Saint-Amand, H., Kay, M., Jellinghaus, M., Hunsicker, S., Herrmann, T., Chen, Y. (2008). Hybrid Machine Translation Architectures within and beyond the EuroMatrix project. 12th EAMT conference, 22–23 September 2008, Hamburg, Germany.

[3] Guerberof, A. (2009). Productivity and quality in MT post-editing. Dostopno na http://www.mt-archive.info/MTS-2009-Guerberof.pdf, 10. 6. 2013.

[4] Hutchins, J. (2007). Example-based machine translation: a review and commentary. Machine Translation vol.19, str. 197–211.

[5] Leal Fontes, H. (2013). Evaluating Machine Translation: preliminary findings from the first DGT-wide translators' survey. Dostopno na http://ec.europa.eu/dgs/translation/publications/magazines/languagestranslation/documents/issue_06_en.pdf, 13. 7. 2013.

[6] Ney, H. (2005). One Decade of Statistical Machine Translation: 1996–2005. Machine Translation Summit (MT Summit), str. i–12–i–17, Phuket, Thailand.

[7] O'Brien, S. (2007). An Empirical Investigation of Temporal and Technical Post-Editing Effort. Translation And Interpreting Studies (tis), II, I.

[8] Och, F.-J., Ney, H. (2004) The Alignment Template Approach to Statistical Machine Translation. Computational Linguistics, vol. 30, str. 417–449.

[9] Specia, L. (2011). Exploiting objective annotations for measuring translation post-editing effort. Proceedings of the 15th Conference of the European Association for Machine Translation, Leuven, str. 73–80.
[10] Torres Dominguez, R. (2012). Translation technologies survey results 2012. Dostopno na http://mozgorilla.com/en/texnologii-en-en/translation-technologies-survey-results/, 10. 7. 2013.
[11] Uszkoreit, H., Federmann, C., Chen, Y., Eisele, A., Theison, S. & Hunsicker, S. (2009). Hybrid Machine Translation. Translingual Eur.
[12] Vashee, K. (2011). Spletni dnevnik na temo Post-Editing MT, 15. 2. 2011, dostopno na http://kv-emptypages.blogspot.com/2011/02/exploration-of-post-editing-mt-part-i.html.

▪

Špela Vintar je izredna profesorica na Oddelku za prevajalstvo Filozofske fakultete Univerze v Ljubljani, kjer poučuje računalniško podprto prevajanje, lokalizacijo, prevajalske tehnologije in terminologijo. Raziskovalno se ukvarja z razvojem sistemov za samodejno luščenje znanja (terminov, definicij in semantičnih relacij) iz eno- in večjezičnih besedil, z empiričnim raziskovanjem značilnosti prevodov in z razvojem slovenskega znakovnega jezika. Sodelovala je v več kot desetih nacionalnih in mednarodnih raziskovalnih projektih s področja korpusnega jezikoslovja in jezikovnih tehnologij. Letos je organizirala prvo mednarodno poletno šolo s področja prevajalskih tehnologij TransTech13 na Reki, Hrvaška. Je članica Evropske zveze za računalniško jezikoslovje (EACL) in predseduje Slovenskemu društvu za jezikovne tehnologije.

Na poti do Islovarja 3.0

[1]Katarina Puc, [2]Tomaž Turk
[1]Slovensko društvo INFORMATIKA; [2]Univerza v Ljubljani, Ekonomska fakulteta
puckatarina@gmail.com; tomaz.turk@ef.uni-lj.si

Izvleček

V članku predstavljamo razvoj spletnega terminološkega slovarja informatike Islovar. To je dolgoročen projekt s ciljem skrb za slovenski strokovni jezik informatike. Prva leta dela so bila učna doba, ko nihče od sodelujočih ni imel izkušenj s podobnim načinom dela. Slovar je bil prosto dostopen in je sproti nastajal tudi s prispevki uporabnikov. Oblikovala se je skupina sodelavcev, ki je poleg strokovnjakov informatikov vključevala jezikoslovce in sčasoma tudi leksikografe. Opredelili smo obliko urejenih slovarskih sestavkov in uredniški postopek. Razvoj Islovarja poteka še danes. Uporabniki dodajajo nove izraze, uredniki posodabljajo in urejajo vsebino. Podpora okolja, število uporabnikov in obiskanost potrjujejo pravilnost rezultatov dela. V pripravi je nova programska rešitev, ki naj bi izkoristila sodobnejša programska orodja, ponudila učinkovitejši uredniški vmesnik, hkrati pa razširila uporabo Islovarja na druge vsebine. Glede na pričakovani razvoj informacijske tehnologije bo Islovar ostal na spletu, kar se ujema z osnovnima usmeritvama: odprtost za zajem vsebine in prosta dostopnost.
Ključne besede: spletni slovar, terminološki slovar, prosti dostop, informatika, računalništvo.

Abstract

Towards Islovar 3.0

In this paper the evolvement of the Islovar on-line dictionary is presented. Islovar is a long-term project with the mission to create and support Slovene terminology in the field of information management and technology. The early years of Islovar were in fact a learning period because the people involved in editorial activities had little experience with lexicography in modern technological environments. The dictionary was free for public access and open for contributions of new entries. The editorial team consisted of informatics experts, linguists and later lexicographers, too. The exact form and structure of dictionary entries and editing procedures were gradually established. Today, the development of Islovar is still an ongoing effort. Users can add new terms while the editors enrich and update the content. The support of the academic environment, the number of users and frequency of queries confirm the appropriateness of the envisaged goals. A new software solution is under development in order to exploit the possibilities of state-of-the-art web technologies, to enable efficient editing and at the same time expand the usability of Islovar in other fields. Considering the expected development of information technology in the future, Islovar will remain on the web, which is consistent with its main values: openness for content provision and free access.
Key words: online dictionary, terminological dictionary, free acccess, informatics, computing.

1 UVOD

Z razvojem informacijskih tehnologij se je v zadnjih letih razvilo spletno založništvo, ki ponuja knjige in revije kar na spletu. Dostopnost do številnih znanstvenih in strokovnih besedil se je s tem izredno povečala. Ogromni so prihranki pri stroških izdajanja, predvsem pa pri času dostopanja. Najbolj očitne so koristi pri izdajanju in uporabi spletnih slovarjev in enciklopedij. Te publikacije so zdaj zvečine brezplačno dostopne, njihovo posodabljanje je preprosto. Številne se vzdržujejo z donacijami ali oglasi, ki jih objavljajo na straneh slovarja. To omogoča, da lahko tako rekoč vsakdo objavi svoj slovarček. Ugotavljamo, da nastaja nekakšna popularizacija objavljanja besedil; mnogo je tudi plevela, mnogo pa je žlahtnih rastlin, ki prej niso bile dostopne.

Tako najdemo na spletu slovarje in slovarčke za množico jezikov. Samo portal Onelook.com ponuja 1060 angleških slovarjev, specificiranih po področjih. Svetovno znana Encyclopedia Britannica se je preselila na splet v novi preobleki v več jezikih. Verjetno najbolj uporabljana med enciklopedijami je Wikipedia, ki izhaja v 285 jezikih.

V Sloveniji od leta 1995 obstaja portal Spletni slovarji, ki ponuja 900 slovarjev za več kot 40 jezikov, ločeno po področjih (Željko, 2013). Za področje računalništvo in informatika je dostopnih 14 slovarjev, med njimi tudi Islovar. V zadnjih letih se razvija portal Termania, ki zdaj vsebuje 36 splošnih, terminoloških in posebnih slovarjev (Amebis, 2013).

Pri urejanju spletnih slovarjev obstaja bistvena razlika. Najbolj pogosto na splet kar prenesejo knjižni slovar, ki ga potem posodabljajo. Ali pa slovar urejajo neposredno na spletu, tako da uporabniki vidijo tudi

prispevke, ki še niso dokončno urejeni. Taka primera sta Wikipedia, pa tudi spletni terminološki slovar informatike Islovar. V članku bomo na kratko opisali nastajanje Islovarja, njegovo uporabo in izkušnje, ki so jih pri tem pridobili uredniki.

2 OD ZAMISLI DO PROJEKTA

2.1 Razvoj spletnega terminološkega slovarja

Leto 2000 je rojstno leto Islovarja. Začelo se je s pobudo za ustanovitev sekcije, ki naj bi sistematično skrbela za strokovni jezik v okviru Slovenskega društva Informatika (v nadaljevanju društvo), sledilo je vabilo članom društva, naj se pridružijo sekciji (Batagelj, 2001). Junija 2000 je društvo ustanovilo jezikovno sekcijo, ki naj bi razvijala strokovni jezik informatike pod motom »Ni strokovne odličnosti brez odličnega strokovnega jezika«. Udeleženci prvih sestankov še niso imeli jasnega načrta, kaj bi delali in kako. Šele po nekaj mesecih razprav so prišli do projekta terminološkega slovarja in se zedinili, da bo v njem uporabljena sodobna informacijska tehnologija, da bo slovar odprt in za uporabnike brezplačen.

Na posvetovanju Dnevi slovenske informatike v Portorožu aprila 2001 je jezikovna sekcija predstavila terminološki slovar, ki je deloval na spletu (Turk, Jaklič, 2001). V slovarju je bilo tedaj okoli tristo izrazov z angleškimi ustreznicami. Razlag ni bilo. Izraze so prispevali člani sekcije, največ jih je prišlo iz tedanjega računalniškega spletnega slovarčka.

Na začetku je bilo delo s slovarjem nerodno. Vedno bolj se je kazalo, da bi morali vzpostaviti metodo uredniškega dela. To se je uresničilo, ko sta se skupini urednikov pridružila sodelavca, ki sta imela večletne izkušnje s pisanjem terminoloških slovarjev. Tedaj je sekcija oblikovala strategijo razvoja, ki je v devetnajstih točkah opredelila glavne značilnosti Islovarja.

Leto 2004 je bilo pomemben mejnik v razvoju. Slovar je po razpravi urednikov dobil ime Islovar. Na posvetovanju Dnevi slovenske informatike je izšel poskusni snopič slovarja, ki je vseboval okrog dvesto izrazov z razlagami, naglasi in zapisom izgovora. Nova programska rešitev je upoštevala izkušnje in želje urednikov. Temeljila je na dogovorjenem uredniškem postopku (Turk, Puc, 2006). Uvedla je oznako zanesljivosti sestavkov: predlog, pregledano, strokovno pregledano, urejeno. Bila je lepo in ergonomsko oblikovana. Rešitev smo javno predstavili na Ekonomski fakulteti v Ljubljani.

2.2 Povezovanje z drugimi

Jezikovna sekcija se je povezovala z institucijami, na katerih so bili zaposleni uredniki Islovarja: z Ekonomsko fakulteto Univerze v Ljubljani (UL; gostovanje na strežniku, programiranje in vzdrževanje), s Filozofsko fakulteto UL, Oddelkom za prevajalstvo (vključevanje študentov), s Fakulteto za elektrotehniko UL, Laboratorijem za telekomunikacije (povezava z njihovim spletnim slovarjem) in Laboratorijem za umetno zaznavanje, sisteme in kibernetiko (zvočni zapis izrazov). Od leta 2010 uredništvo Islovarja pri vnašanju zvočnega zapisa sodeluje s podjetjem Alpineon, razvoj in raziskave. Vse te povezave so obogatile vsebino slovarja.

Leta 2006 se je društvo obrnilo na vse slovenske visokošolske ustanove, ki so imele v svojih izobraževalnih programih informatiko, s priporočilom, naj uporabljajo Islovar, prispevajo nove izraze in za lažjo dostopnost do strokovnih besedil objavljajo diplomska in magistrska dela ter druge dokumente na spletu. To je brez dvoma vplivalo na obiskanost Islovarja, pa tudi spodbudilo k objavljanju besedil na spletu, kar je postalo pomemben vir pri urejanju slovarja.

Društvo je podpisalo sporazum o sodelovanju s Fakulteto za upravo UL, ki sekciji omogoča delo v računalniških učilnicah fakultete.

3 UPORABA IN UREJANJE

3.1 Izkaznica

V Islovarju je bilo na dan 20. 6. 2013 6.406 iztočnic. Vsako leto je dodanih okoli štiristo novih izrazov, nekaj takih, ki po presoji urednikov ne spadajo v Islovar, je tudi izbrisanih. Slovar odlikujejo dostopnost, odprtost, prijaznost, ažurnost, preglednost in zanesljivost. Statistika kaže okrog 20.000 iskanj mesečno.

Islovar je dostopen štiriindvajset ur na dan. Uporabniki lahko iščejo po slovenskem ali po angleškem izrazu. Uporabniški vmesnik je prijazen. Uporabnikom sproti ponuja krajša navodila za delo. Uporabniki lahko vnašajo nove izraze, popravljajo lastne izraze, jim dodajajo razlago, komentirajo slovarske sestavke ali se oglašajo z vprašanji v forumu. Lahko si ogledajo nove izraze, lahko se sprehajajo po Islovarju, ko iščejo naključne izraze ali izraz dneva.

Islovar vsebuje obširen opis, povezave na druge spletne slovarje, navedeni so uporabljeni viri, zgodovina razvoja in vsi sodelavci, ki so v preteklosti prispevali k razvoju. Islovar pri iskanju ponudi izraz

in besedne zveze z njim. Če v Islovarju ni enakega izraza, ponudi podobne izraze, kar je koristno zlasti, kadar se uporabnik zmoti pri zapisu izraza. Urejen slovarski sestavek je opremljen z razlago, povezavo na sinonime in podobne izraze ter z zvočnim zapisom izgovora. Vsaka iztočnica je opremljena z značko zanesljivosti (predlog, pregledano, strokovno pregledano, urejeno), ki označuje stopnjo obravnave v uredniškem postopku. Urejeni izrazi imajo najvišjo stopnjo zanesljivosti.

digitalna kompetenca -e -e [digita:lna kOmpEte:ntsa] ž *(angl. digital competence)*

urejeno

znanje, spretnosti in motivi, ki so potrebni za učinkovito izvršitev določene naloge ali rešitev problema z uporabo informacijske tehnologije; sin. e-kompetenca

Slika 1: **Primer urejenega sestavka**

Iztočnici sledi v oglatem oklepaju zapis v MRPA,[1] ki je podlaga za zvočni zapis in ga uporabnik lahko sliši s pritiskom na gumb »zvočnik«. V okroglem oklepaju sledi angleška ustreznica.

Obvezni del urejenega slovarskega sestavka je razlaga. Slovarska razlaga je kratka, razumljiva, kolikor mogoče poljudna. Razširjena je z navedbo sinonima ali podobnega izraza. Uporabnikom omogoča, da spoznajo pomen pojma, ki so ga iskali v slovarju. Urednikom pomaga pri natančni opredelitvi pojma in po potrebi oblikovanje novega izraza v slovenščini.

Islovar naj bi zajemal področje informatike in računalništva. Vendar na vprašanje, kaj danes spada v terminološki slovar informatike, ni jasnega odgovora. Uporaba informacijske tehnologije se širi tako hitro, da so meje za zdaj nejasne. Glede na praktično neomejenost prostora na spletu uredništvo Islovarja vključuje tudi izraze z mejnih področij. Primer takega področja je e-izobraževanje, ki je v zadnjem času zelo aktualno.

3.2 Uporabniki

Uporabniki Islovarja so vsi, ki se pri učenju, študiju ali delu srečujejo z informatiko in so vešči uporabe spleta – dijaki, študenti, profesorji, prevajalci, informatiki. V slovarju je trenutno registriranih 1.455 uporabnikov. Njihovo število je brez dvoma veliko večje, ker se uporabnikom ni treba registrirati, če samo iščejo izraze. Tako je v obdobju od 1. 5. 2012 do 1. 5. 2013 Islovar obiskalo 26.856 uporabnikov, ki so slovar skupno uporabili 72.412-krat. Največ obiskov je bilo iz Slovenije (okr. 90 %), Velike Britanije (1,7 %) in Belgije (1,3 %), sledijo pa Nemčija, Avstrija, Hrvaška, Italija, Luksemburg in Srbija s po 0,5 % obiskov. Predvidevamo, da gre za Slovence po svetu, ki delujejo kot pisci besedil, prevajalci in podobno. Pri obiskih iz Slovenije je nekaj čez polovico obiskovalcev z območja Ljubljane (55,8 %), iz Maribora 14,4 % in iz Celja 4,3 %, z manjšimi deleži sledijo drugi kraji.

Večinoma gre za enostavne poizvedbe – v prej navedenem obdobju je samo dva odstotka iskanj izvedenih z naprednim iskalnikom.

Zlasti mladi nimajo težav pri komuniciranju prek spleta, veliko lažje iščejo po spletu kot po knjigah. Zato lahko ocenjujemo, da je večina uporabnikov mladih.

3.3 Uredniki in uredniško delo

Uredniški vmesnik je drugačen od uporabniškega. V Islovarju je obsežno navodilo za delo urednikov. Uredniki imajo možnost naprednega iskanja, ki omogoča iskanje po raznih kriterijih. Slovar pokaže podrobnosti slovarskega zapisa, npr. ime avtorja, datum vnosa, vso zgodovino sprememb, uporabljeni vir, če ga je avtor zapisal.

Uredniški postopek ločuje tri glavne faze urejanja:

- vnos izrazov v Islovar,
- strokovno urejanje in
- slovaropisno urejanje.

Pri vnosu sodelujejo razen urednikov tudi uporabniki, ki izrazom dodajajo razlage in lahko tudi popravljajo svoje sestavke. Enako vnašajo izraze tudi uredniki. Taki sestavki prejmejo značko »predlog« in jih uredništvo praviloma pregleda, po potrebi dopolni ali se posvetuje glede vsebine sestavka. Značka »pregledano« v takem sestavku pomeni, da je izraz vsebinsko primeren za Islovar, vendar sledi še na-

[1] Machine Readable Alphabet – računalniško berljivi zapis.

tančno urejanje, zelo verjetno spreminjanje razlage, pogosto tudi samega izraza.

Strokovno urejanje je zahtevnejši postopek, pri katerem sodelujejo uredniki Islovarja v skupinah. V strokovni skupini sodeluje tri do pet urednikov. Na sestankih razpravljajo o vsebinsko zaokroženi zbirki, ki jo pred tem pripravi eden od članov skupine. Pregledujejo pravilnost slovenskih izrazov, pa tudi vsebino razlag. Pri urejanju strokovne skupine pregledujejo vse dostopne tiskane vire, od slovarjev do člankov v strokovnih revijah in učbenikov. Vendar so tiskani viri za področje informatike pogosto že zastareli. Zato uredniki upoštevajo predvsem spletne informacijske vire, kot so spletišča najdi.si, Evroterm, Google, Wikipedija, korpus DSI.[2] Dostopni so številni članki in druga strokovna besedila; avtorji in institucije, ki delujejo na področju informatike, pogosto objavljajo prispevke na svetovnem spletu. Pri vrednotenju izrazov in njihovih razlag uredniki upoštevajo pogostost objav, pa tudi kdo so njihovi avtorji. Omejevanje virov samo na to, kar najdemo na spletu, je morda sicer praktičen, toda enostranski pristop (Puc, 2009). Izrazi prejmejo značko »strokovno pregledano«.

Sledi slovaropisno urejanje izrazov. Slovaropisna skupina je sestavljena interdisciplinarno, dva člana imata večletne izkušnje pri slovaropisju in urejanju terminoloških slovarjev. Skupina pregleduje zbirke, ki so bile že strokovno pregledane, uredi zaglavja sestavkov – iztočnico, obrazilo rodilnika, spol, oznake za besedno vrsto, naglas, doda oceno primernosti izraza, uredi sinonime in povezave na sorodne izraze.

Za naglase vsebuje Islovar posebno rešitev. Nabor črkovnih znakov na spletu za zdaj ne omogoča preprostega zapisa naglasov in nekaterih znakov, kot je npr. naglašeni polglasnik. Prav ta pa se pojavlja pri izgovoru kratic, ki so v informatiki pogoste. Zato Islovar vsebuje posebno rešitev naglasnih znamenj, ki se pri urejenih izrazih spremenijo v MRPO in zvočni zapis.

Pred dokončno ureditvijo gre zbirka v razpravo vsem urednikom, ki lahko prispevajo pripombe in predloge. Te nato obravnava slovaropisna skupina, ponovno pregleda vse sestavke in jih označi s »pregledano«. Spreminjanje teh sestavkov nato brez soglasja slovaropisne skupine ni več mogoče.

Odprtost Islovarja bi lahko ogrožala kakovost njegove vsebine. To uredništvo rešuje z večkratnim pregledovanjem sestavkov, pogosto celo s ponovnim pregledovanjem in dopolnjevanjem že urejenih sestavkov. Vsak izraz je pregledan vsaj štirikrat, razlaga vsaj trikrat. Ta skrb za kakovost hkrati povzroča, da postopek urejanja poteka počasi in je v Islovarju dokončno urejena samo nekaj več kot tretjina vseh izrazov.

4 NAČRTI

Spletni slovar za razliko od knjižne izdaje poleg lažje dostopnosti prinaša tudi dinamičnost in možnost izmenjave mnenj in izkušenj. Ker je urejanje vsebine terminološkega slovarja dolgoročen projekt, zlasti če je vanj vključenih več sodelavcev, je koordinacija med njimi ključnega pomena. Poleg tega je jezik živ, se spreminja in raste iz dneva v dan, še posebno na področju informatike in računalništva. To pri dolgoročni naravi projekta pomeni, da moramo slovarske sestavke, ki bi jih klasično že uvrstili v knjižno izdajo, redno spreminjati in dopolnjevati.

Zato je Islovar poleg same vsebine tudi razmeroma zapleten spletni program, ki omogoča različne funkcionalnosti, ki jih pri svojem delu uporabljajo uredniki in uporabniki slovarja. Spletni program je namenjen podpori uredniških postopkov, ki smo jih razvijali iz leta v leto, na podlagi pridobljenih izkušenj in tudi ob pomoči strokovnjakov s področja slovaropisja, ki so med stalnimi sodelavci Islovarja.

Postopki urejanja so že dosegli stopnjo zrelosti, tako da je način urejanja že ustaljen in usklajen in veliko sprememb na tem področju v prihodnje ne gre pričakovati. Napredek je še mogoč pri posameznih opravilih, ki pa so odvisna tudi od programske rešitve.

Spletno zasnovo Islovarja kot orodja za uporabnike ter programsko opremo kot orodja za urednike bomo ohranili, ker poleg samega dela podpira osnovno usmeritev jezikovne sekcije Slovenskega društva INFORMATIKA, to je odprtost in prosta dostopnost. Trenutno načrtujemo prenovo spletne aplikacije, ki bo zasnovana nekoliko drugače kot obstoječa in pri kateri bomo uporabili nekaj novejših tehnologij za gradnjo spletnih programov (npr. AJAX), ki pri zasnovi obstoječe rešitve še niso bile razvite. To bo omogočilo bolj učinkovito uporabo in izvedbo določenih opravil pri urejanju slovarja. Uporabili bomo minimalistični pristop pri gradnji uporabniškega vmesnika in sodobne prijeme pri obravnavi in shranjevanju podatkov, osnovni cilj pri tem pa je olajšanje nekaterih opravil pri urejanju in uporabi Islovarja. To

[2] Korpus informatike vsebuje besede iz zbornikov posvetovanja DSI 2003–2012 in člankov revije Uporabna informatika 2010–2012.

bo že tretja popolnoma prenovljena verzija programske opreme od ustanovitve jezikovne sekcije. Zaradi boljše vidnosti načrtujemo tudi vključitev spletne izdaje Islovarja v socialna omrežja.

Poleg programske opreme je pomembna tudi vsebina – pred uredništvom Islovarja je velik izziv, saj se informatika in računalništvo zelo hitro širita in razvijata. Nova področja prinašajo s sabo tudi nove izraze, ki jih je treba evidentirati, zlasti pa pravilno urediti z vidika slovenistike in z vsebinskega vidika.

5 SKLEP

Delo pri Islovarju je skupinsko. Področje informatike postaja vedno bolj razvejano, kompleksno, zato danes ni več mogoče, da bi terminološki slovar sestavil en sam avtor. Zelo pomembna je interdisciplinarnost v delovnih skupinah, sodelovanje strokovnjakov, informatikov in jezikoslovcev. Za uspešno delo so potrebni toleranca, spoštovanje mnenja drugih, včasih tudi kompromis. Nič manj pomembna ni trajna, prijateljska vez, ki se po večletnem sodelovanju ustvarja v skupinah.

V Islovarju je uskladiščeno znanje, ki so ga v letih dela prispevali uporabniki in uredniki in je zdaj na voljo javnosti. To znanje je zlasti razvidno iz razlag, ki jih vsebujejo slovarski sestavki. Urejanje terminološkega slovarja pomeni ne samo posredovanje lastnega znanja, temveč tudi učenje, pridobivanje novega znanja od drugih sodelujočih v skupini.

Islovar je informativen, pa tudi normativen slovar. Namen Islovarja je spodbujati uporabo pravilnih slovenskih strokovnih izrazov. Pri končnem urejanju slovaropisna skupina oceni primernost posameznega izraza in ga tudi ustrezno označi, če ni sprejemljiv. Namesto njega ponudi drug, dober slovenski izraz. S tem se povečuje uporaba pravilnega, lepega strokovnega jezika.

Islovar je prav gotovo dolgoročen projekt zaradi svoje narave in zasnove – ker je strokovni jezik živ in se nenehno spreminja, je treba slediti tem spremembam. Glavni cilji in smernice pri urejanju Islovarja ostajajo isti že vrsto let, lahko bi celo rekli, da je organizacija uredniškega dela zrela; vprašanje, ki nam ob tem pride na misel, pa je, ali je že zrela vsebina Islovarja. V tem trenutku bi lahko rekli, da bi lahko slovar že nekajkrat izdali v knjižni obliki, vendar to ni skladno z njegovo naravo in dinamiko.

Sodelavci Islovarja opravljajo uredniško delo in druga opravila s tem v zvezi pretežno ljubiteljsko, v osnovi je slovar odprt in prosto dostopen, in to se kaže tudi v medsebojni nesebični izmenjavi znanja in izkušenj. Še zlasti je vedno prisotno zavedanje o koristnosti prispevka stroki, jeziku in slovenski kulturi nasploh.

VIRI IN LITERATURA

[1] Batagelj, V. (2001). Razvoj slovenskega računalniškega izrazja. *Uporabna informatika, št. 2*, str. 95–99.

[2] Puc, K. (2009).*Urejanje spletnega terminološkega slovarja Islovar*, Terminologija in sodobna terminografija, ur. N. Ledinek, M. Žagar Karer, M. Humer, Založba ZRC SAZU, Ljubljana.

[3] Puc, K., Erjavec, T. (2006). Uporaba korpusa pri urejanju spletnega terminološkega slovarja. *Zbornik 5. slovenske in 1. mednarodne konference Jezikovne tehnologije IS.LTC*, Ljubljana.

[4] Slovensko društvo INFORMATIKA (2013). *Islovar*. http://www.islovar.org/.

[5] Turk, T., Jaklič, J. (2001). Terminološki spletni slovar. *Zbornik posvetovanja Dnevi slovenske informatike*.

[6] Turk, T., Puc, K. (2006). *Islovar kot model spletnega terminološkega slovarja*. Obdobja 24, Razvoj slovenskega strokovnega jezika. Univerza v Ljubljani, Filozofska fakulteta.

[7] Turk, T., Puc, K. (2013). Islovar včeraj, danes, jutri. *Zbornik posvetovanja Dnevi slovenske informatike.*

[8] Željko, M. (2013). Spletni slovarji. http://evroterm.gov.si/slovar/.

[9] Amebis (2013). *Termania*. http://www.termania.net.

▪

Katarina Puc je diplomirala na Filozofski fakulteti Univerze v Ljubljani iz predmetov francoski jezik s književnostjo in angleški jezik s književnostjo ter magistrirala iz poslovne politike in organizacije na Ekonomski fakulteti Univerze v Ljubljani z delom Ekonomski in organizacijski kriteriji za odločanje o uporabi tehnologije v pisarniških sistemih. Pomembnejše delovne izkušnje: izobraževanje, uredništvo in tehnično uredništvo knjig, revij, zbornikov, prevajanje književnih del in strokovnih besedil iz angleščine, francoščine in nemščine v slovenščino, lektoriranje strokovnih besedil. Pobudnica ustanovitve jezikovne sekcije pri Slovenskem društvu Informatika. Urednica spletnega terminološkega slovarja informatike Islovar.

▪

Tomaž Turk je izredni profesor na Ekonomski fakulteti Univerze v Ljubljani. Poleg pedagoškega dela sodeluje pri mnogih raziskovalnih in svetovalnih projektih. Raziskovalno se ukvarja s problematiko privzemanja informacijske in komunikacijske tehnologije, z ekonomiko informatike in telekomunikacij, menedžmentom informacijskih tehnologij in telekomunikacijskih storitev ter razvojem programskih rešitev. Objavil je več kot petdeset raziskovalnih člankov in delov monografij. Je član upravnega odbora Ekonomske fakultete Univerze v Ljubljani ter član več raziskovalnih in strokovnih združenj (Zveza ekonomistov Slovenije, Internet Society, European Distance Education Network ter International Association for Computer Information Systems). Pobudnik ustanovitve jezikovne sekcije ter njen predsednik od leta 2010.

Iz Islovarja

V tej številki revije objavljamo zbirko izrazov, ki se uporabljajo pri pisavi in v tiskarstvu. Izraze lahko komentirate tako, da se prijavite v poglavju *Nov uporabnik*, poiščete izraz, ki ga želite komentirati, in zapišete svoj komentar ali predlog spremembe. V Islovar lahko kot uporabnik dodajate tudi nove izraze in tako pripomorete k bogatenju vsebine. Islovar najdete na naslovu *http://www.islovar.org*.

abecédni nabòr znákov -ega -ôra -- m (*angl. alphabetic character set*)
nabor znakov, ki obsega velike in male latinične črke

álfanumêrični nabòr znákov -ega -ôra -- m (*angl. alphanumeric character set*)
nabor znakov, ki obsega črke in številke

berljívost -i ž (*angl. 1. legibility, 2. readability*)
1. lastnost zapisanega, da ga je mogoče prebrati, npr. berljivost zapisa[1]; sin. čitljivost
2. lastnost, ki določa napor, s katerim je mogoče prebrati besedilo, npr. berljivost besedila, berljivost podatkov

bítni znák -ega -a m (*angl. bitmap character*)
gl. rastrski znak

čitljívost -i ž (*angl. legibility*)
lastnost zapisanega, da ga je mogoče prebrati, npr. čitljivost zapisa; sin. berljivost (1)

dólarski znák -ega -a m (*angl. dollar sign*)
vidni znak $, ki označuje podatkovni tip spremenljivke, konec vrstice v regularnem izrazu, pozivnik

eksponènt -ênta m (*angl. superscript, superior*)
pomanjšan vidni znak, dvignjen ob drugem znaku, npr. v matematičnih, kemijskih besedilih

generátor znákov -ja -- m (*angl. character generator*)
bralni pomnilnik, v katerem so shranjene znakovne matrike rastrske pisave

gráfični načín -ega -a m (*angl. graphics mode*)
prikaz podatkov na zaslonu, pri katerem je mogoče spreminjati vrednost posamezni slikovni piki (1); prim. znakovni način

kodírani nabòr znákov -ega -ôra -- m (*angl. coded character set, code page, codepage*)
standardiziran nabor znakov, pri katerem je vsakemu znaku (2) dodeljena številka; sin. kodna stran, kodni nabor

kódna strán -e -í ž (*angl. coded character set, code page, codepage*)
standardiziran nabor znakov, pri katerem je vsakemu znaku (2) dodeljena številka; sin. kodirani nabor znakov, kodni nabor

kódni nabòr -ega -ôra m (*angl. coded character set, code page, codepage*)
standardiziran nabor znakov, pri katerem je vsakemu znaku (2) dodeljena številka; sin. kodirani nabor znakov, kodna stran

kontrólni znák -ega -a m (*angl. control character*)
gl. krmilni znak

krépko prisl. (*angl. bold*)
izraža slog pisave, pri katerem so črke odebeljene

krmílni znák -ega -a m (*angl. control character*)
nevidni znak, s katerim se krmili program, izhodna naprava; sin. kontrolni znak

kurzíva -e ž (*angl. italic*)
gl. ležeča pisava

ležéča pisáva -- ž prisl. (*angl. italic*)
pisava s postrani oblikovanimi črkami; sin. kurziva

ločílo -a s (*angl. punctuation*)
vidni znak za členitev pisanega besedila, npr. pika, vejica

ločítveni znák -ega -a m (*angl. separator symbol, delimiter*)
znak, več znakov, ki se uporabljajo za označitev začetka ali konca polja, podpolja, zapisa (3), npr. ločilec podpolja; sin. razmejevalec, separator

maskírni znák -ega -a m (*angl. wildcard, wild character*)
gl. nadomestni znak

MICR MICR-ja [micərə] m krat. (*angl. magnetic-ink character recognition*)
računalniško prepoznavanje znakov (2), zapisanih z magnetnim črnilom, ki se uporablja pretežno v bančništvu za poslovanje s čeki

mínus -a m (*angl. minus sign*)
1. znak (-), ki označuje operacijo odštevanja
2. računski znak (-) za označevanje negativnih vrednosti; prim. plus (2)

nabòr znákov -ôra -- m (*angl. character set, charset*)
dogovorjena množica znakov (2), ki omogoča zapisovanje in izmenjavo podatkov, npr. abecedni nabor znakov, numerični nabor znakov, alfanumerični nabor znakov

nadoméstni znák -ega -a m (*angl. wildcard, wild character*)
znak (2), ki nadomešča en znak (2) ali več znakov (2) pri poizvedovanju, npr. zvezdico (*), vprašaj (?); sin. maskirni znak

natisljívi znák -ega -a m (*angl. printable character, printing character*)
gl. vidni znak

nèdovóljeni znák -ega -a m (*angl. illegal character*)
znak (2), ki v danem kontekstu ni dovoljen

nènatisljívi znák -ega -a m (*angl. non-printable character, non-printing character*)
gl. nevidni znak

nèvídni znák -ega -a m (*angl. non-printable character, non-printing character*)
vsak od znakov (2) brez pripadajoče grafične podobe, npr. krmilni znak; sin. nenatisljivi znak; prim. vidni znak

níz znákov -a -- m (*angl. character string, alphanumeric string*)
zaporedje znakov (2), ki se obravnava kot celota; sin. niz; prim. spojitev

numêrični nabòr znákov -ega -ôra -- m (*angl. numeric character set*)
nabor znakov, ki obsega števke

obrísni znák -ega -a m (*angl. outline character*)
znak (2), opisan z matematično krivuljo

OCR OCR-a [oceèr, -êra] m krat. (*angl. optical character recognition*)
gl. optično prepoznavanje znakov

óptično prepoznávanje znákov -ega -a -- s (*angl. optical character recognition*, krat. *OCR*)
postopek pretvorbe bitne slike besedila v besedilo, ki ga je mogoče obdelovati v urejevalniku besedil

paginácija -e ž (*angl. page numbering, pagination*)
označitev strani z zaporednimi številkami, črkami; sin. straničenje (1)

podčŕtano -am prisl. (*angl. underline*)
tako, da je uporabljen slog pisave, pri katerem so pod črkami narejene črte

pôlkrêpko prisl. (*angl. semibold*)
tako, da je uporabljen slog pisave, pri katerem so črke nekoliko odebeljene, vendar manj kot pri krepki pisavi

polnílni znák -ega -a [ọu̯n] m (*angl. fill character*)
vsak od znakov (2), s katerimi se niz znakov (2) zapolni do predpisane dolžine

posébni znák -ega -a m (*angl. special character*)
1. vsak od znakov, ki jih uporabljajo nekateri jeziki, vendar v naboru ASCII niso predvideni, npr. č, š, ž
2. znak (2), ki ima v danem kontekstu posebno vlogo, npr. @ v elektronskem naslovu

prázni znák -ega -a m (*angl. whitespace, white space*)
presledek med dvema vidnima znakoma, ki je lahko v besedilu ali sliki

prečŕtano prisl. (*angl. strikethrough*)
tako, da je uporabljen slog pisave, pri katerem so po sredini črk narejene vodoravne črte, ki navadno označujejo besedilo, ki bo izpuščeno

predstavítev znáka -tve -- ž (*angl. character representation*)
enoznačna prireditev številčne kode znaku (2) znotraj kodiranega nabora znakov

prográmska pisáva -e -e ž (*angl. soft font*)
pisava, ki se preslika z računalniškega diska v tiskalniški pomnilnik; prim. vgrajena pisava

računski znák -ega -a m (*angl. operation character*)
vsak od znakov (2), za označevanje v aritmetičnih operacijah, npr. plus (+), minus (-)

rástrski znák -ega -a m (*angl. bitmap character*)
znak (2), predstavljen kot polje^2 slikovnih pik (1); sin. bitni znak

razmejeválec -lca m (*angl. separator symbol, delimiter*)
gl. ločitveni znak

separátor -rja m (*angl. separator symbol, delimiter*)
gl. ločitveni znak

sredínska píka -e -e ž (*angl. midpoint, small bullet*)
vidni znak (•), ki se uporablja kot računski znak za množenje ali pri naštevanju, odstavčnem razporejanju besedila

straníčenje -a s (*angl. page numbering, pagination*)
1. označitev strani z zaporednimi številkami, črkami; sin. paginacija
2. delitev dokumenta na strani

ubéžni znák -ega -a m (*angl. escape*)
krmilni znak, ki na vhodu označuje prekinitev delovanja ali vnosa podatkov, na izhodu pa začenja zaporedje za krmiljenje izhodnih naprav

vídni znák -ega -a m (*angl. printable character, printing character*)
vsak od znakov (2) s pripadajočo grafično podobo, npr. črka, številka, ločilo, matematični znak, oznaka valute; sin. natisljivi znak; prim. nevidni znak

znák -a m (*angl. 1.sign, 2.character, 3.signal*)
1. dogovorjen lik, ki ima določen pomen
2. vsak od elementov besedila, ki v določeni sestavi oblikuje pomen, npr. črka, številka, ločilo
3. gib, zvok, s katerim se kaj sporoča ali na kaj opozarja

znák na pálec -a -- -- m (*angl. characters per inch*, krat. *cpi*) t.d.
enota za gostoto izpisa, ki podaja število odtisnjenih znakov (2) na širini enega palca; sin. znak/palec

znák na sekúndo -a -- -- m (*angl. characters per second*, krat. *cps*) t.d.
enota za hitrost prenosa podatkov, ki podaja število znakov (2), prenesenih v eni sekundi; sin. znak/s

znák za évro -a -- -- m (*angl. euro sign*)
vidni znak €, povzet po oznaki valute Evropske unije

znák za fúnt -a -- -- m (*angl. pound sign*)
vidni znak £, povzet po oznaki britanske valute

znákovna kóda -e -e ž (*angl. character code*)
enolično določena številka, ki je v danem kodiranem naboru znakov dodeljena danemu znaku

znákovna matríka -e -e ž (*angl. character matrix*)
dvorazsežno polje[2] dvojiških vrednosti, ki se izriše kot rastrski znak

znákovni náčin -ega -a m (*angl. character mode, text mode*)
prikaz podatkov na zaslonu, pri katerem je zaslon dvorazsežno polje[2] znakov (2) in je mogoče spreminjati le cel znak, ne pa vrednosti posamezne slikovne pike (1); prim. grafični način

znákovni podátkovni típ -ega -ega -a m (*angl. character type, character, char*)
podatkovni tip, katerega vrednosti so kodirani alfanumerični znaki; sin. znakovni tip

znákovni típ -ega -a m (*angl. character type, character, char*)
podatkovni tip, katerega vrednosti so kodirani alfanumerični znaki; sin. znakovni podatkovni tip

znákovno usmérjen -- -a -- -o prid. (*angl. character-oriented, character-based*)
1. ki se nanaša na znakovni način, npr. znakovno usmerjen uporabniški vmesnik
2. ki se nanaša na znakovni podatkovni tip, npr. znakovno usmerjen prenos

zrcálo -a s (*angl. layout*)
potiskani del strani brez paginacije in podaljškov, ki segajo čez pravokotni ali kvadratni lik potiskane ploskve

Izbor pripravlja in ureja Katarina Puc s sodelavci Islovarja

Koledar prireditev

21. konferenca Dnevi slovenske informatike: Informatika – neizkoriščeni dejavnik razvoja	14.–16. april 2014	Portorož, Slovenija	http://www.dsi2014.si/
22nd European Conference on Information Systems (ECIS 2014)	9.–14. junij 2014	Tel Aviv, Izrael	http://ecis2014.eu/

Pomembni spletni naslovi

→ IFIP News: **http://www.ifip.org/images/stories/ifip/public/Newsletter/news ali www.ifip.org → Newsletter**

→ IT Star Newsletter: **www.itstar.eu**

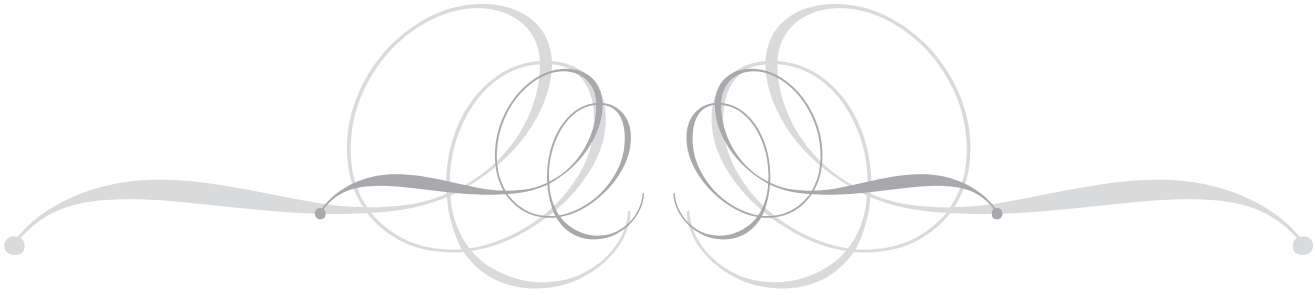
→ ECDL: **www.ecdl.com**

→ CEPIS: **www.cepis.com**

Dostop do dveh tujih strokovnih revij

→ Revija **Upgrade** (CEPIS) v angleščini (ISSN 1684-5285) je dostopna na spletnem naslovu: **http://www.upgrade-cepis.org/issues/2008/4/upgrade-vol-IX-4.html**.

→ Revija **Novática** (CEPIS) v španščini (ISSN 0211-2124) je dostopna na spletnem naslovu: **http://www.ati.es/novatica/**.



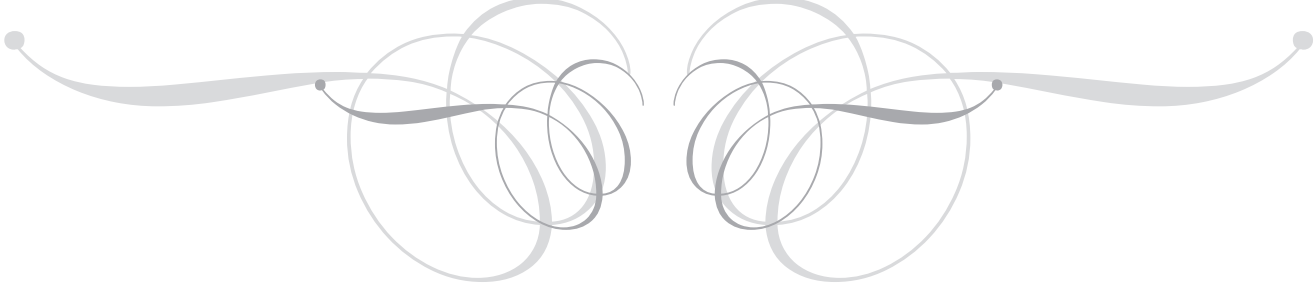
Bralcem in sodelavcem revije

Uporabna informatika

želimo uspešno in ustvarjalno

novo leto 2014

Uredništvo



Včlanite se v Slovensko društvo INFORMATIKA

Pristopna izjava

za članstvo v Slovenskem društvu INFORMATIKA

Pravne osebe izpolnijo samo drugi del razpredelnice

Ime in priimek	
Datum rojstva	
Stopnja izobrazbe	srednja, višja, visoka
Naziv	prof., doc., spec., mag., dr.
Domači naslov	
Poštna št. in kraj	
Ulica in hišna številka	
Telefon (stacionarni/mobilni)	

Zaposlitev člana oz. člana - pravna oseba	
Podjetje, organizacija	
Kontaktna oseba	
Davčna številka	
Poštna št. in kraj	
Ulica in hišna številka**	
Telefon	
Faks	
E-pošta	

Zanimajo me naslednja področja/sekcije*

- ☐ jezik
- ☐ informacijski sistemi
- ☐ operacijske raziskave
- ☐ seniorji
- ☐ zgodovina informatike
- ☐ poslovna informatika
- ☐ poslovne storitve
- ☐ informacijske storitve
- ☐ komunikacije in omrežja
- ☐ softver
- ☐ hardver
- ☐ upravna informatika
- ☐ geoinformatika
- ☐ izobraževanje

podpis

kraj, datum

Pošto društva želim prejemati na domači naslov / v službo.

Članarina znaša: 18,00 € - redna
7,20 € - za dodiplomske študente in seniorje (ob predložitvi dokazila o statusu)
120,00 € - za pravne osebe

Članarino, ki vključuje glasilo društva – revijo **Uporabna informatika**, bom poravnal sam / jo bo poravnal delodajalec.

DDV je vključen v članarino.

Naročilnica na revijo UPORABNA INFORMATIKA

Naročnina znaša: 35,00 € za fizične osebe
85,00 € za pravne osebe – prvi izvod
60,00 € za pravne osebe – vsak naslednji izvod
15,00 € za študente in seniorje (ob predložitvi dokazila o statusu)

DDV je vključen v naročnino.

ime in priimek ali naziv pravne osebe in ime kontaktne osebe

davčna številka, transakcijski račun

naslov plačnika

naslov, na katerega želite prejemati revijo (če je drugačen od naslova plačnika)

telefon/telefaks

elektronska pošta

Podpis

Datum

Znanstveni prispevki

Strokovni prispevki

Informacije

ISSN 1318-1882