НАУЧНИ РАД УДК 004.912:811.163.41’322

ПРОГРАМИ ЗА ЕТИКЕТИРАЊЕ ТЕКСТА НА СРПСКОМ ЈЕЗИКУ*

Зоран Поповић**
Hemofarm, STADA

Апстракт: овај текст даје упоредни преглед постојећих језичких алата, односно програма за етикетирање, заснованих пре свега на методама машинског учења, уз конкретне тестове и резултате различитих програма над текстом на српском језику. У ту сврху су коришћени већ припремљени етикетирани корпуси и десетострука унакрсна провера (10-fold cross-validation), и посебно развијен поступак аутоматизованог тестирања реализованог unix скриптовима (bash, perl, awk) – TnT је показао најбоље перформансе, док су се Tree Tagger и SVMTool показали успешнијим у неким специјалним случајевима. Могућност упаривања различитих метода и програма за етикетирање, као и интеграција са другим окружењима за ОПЈ отварају могућност даљих испитивања оваквих решења.

Кључне речи: етикетирање, tagger, PoS, машинско учење, NLP, рачунска лингвистика

*Овај рад представља резултате приказане у оквиру дипломског (мастер) рада који је одбрањен на Математичком факултету 2009. године

**shoom013@gmail.com

1. Увод – две парадигме рачунске лингвистике

Обрада природног језика (ОПЈ, тј. NLP, Natural Language Processing) као једне од области рачунске лингвистике (Computational Linguistics) подразумева најчешће веома сложене поступке у смислу времена извршавања. Обрада природног језика се дели у фазе, које не морају у свакој апликацији бити примењене, као што су лексичка анализа (сегментација и токенизација улазног говора или текста, где се најпре одређује почетак и крај реченица и речи, као и основне лексичке класе – будуће лексеме, или токени: бројеви, интерпункција, речи, HTML етикете, и слично), морфо-синтаксна анализа (структура речи, реченица и текста), и на крају семантичка анализа или чак анализа прагматике. Парсери, програми који се баве синтаксном анализом, имају нимало лак задатак да обухвате сва правила и особине неког природног језика. Традиционалан приступ је од „врха на доле", где се сложеним анализама у обједињеном поступку добијају и једноставније особине као што су лексичке. Циљ и основни резултат ових анализа су структуре и правила по угледу на формалне граматике Ноама Чомског (синтаксно дрво реченице). Ово одговара и неоствареном циљу потпуног описивања природног језика неком одговарајућом теоријом предикатског рачуна првог реда.

Други квантитативни приступ овим проблемима који више одговара обратном поступку (од дна ка врху) и у којем поступак почиње ефикасним и брзим алгоритмима статистичке природе који нису егзактни у традиционалном смислу, дао је неочекивано добре резултате за једноставније лексичке обраде и битно допринео новим решењима ОПЈ. Једна класа таквих програма који се називају програми за етикетирање (енгл. taggers) бави се откривањем категорије речи у реченици. Ови програми, поједностављено говорећи, свакој речи у тексту према морфолошкој, синтаксној и другим улогама додељује одговарајућу етикету, која описује њену припадност подкласи одговарајуће лексичке класе: именица, заменица, глагола, предлога, итд. Затворене класе етикета се не мењају (нпр. бројеви и заменице), а код отворених се очекују промене мењањем улазног корпуса. Број тих подкласа и њихових категорија може бити веома велик ако се узму у обзир и друге особине (род, број, лице, падеж и слично), и одређен је сложеношћу модела језика који се на тај начин гради, али и самим језиком. Негде 80-тих година први практични успеси европских истраживача с програмом CLAWS (Leech и др. 1987), који се делом заснива на скривеним ланцима Маркова (СЛМ), покренули су даље интересовање за ове и сличне проблеме, а посебно за програме за етикетирање и даљи развој статистичке ОПЈ, као и примену машинског учења у ОПЈ.

Ова два приступа представљају две парадигме које су се могле препознати и у одговарајућим сродним областима рачунарства, и у различитим временским периодима развијале су се, а понекад и сукобљавале: симболичка наспрам квантитативне парадигме, може се пратити и до семиотичких сукоба рационалиста и емпириста у оквирима ОПЈ или бихејвиористичких расправа. С једне стране су интуитивна анализа и често "ручно" креирана симболичка правила, док се с друге стране подешавањем нумеричких параметара статистичког модела долази до аутоматског грађења правила. Прва поседује дубље и често интуитивно јасније лингвистичко знање о природном језику, док је друга способнија да се у пракси избори са неисправним или непознатим улазом и боље генерализује иако има плиће језичко знање. У домену текстуелног инжењерства (појам који се среће примера ради код комерцијалних примена машинског превођења и других система ОПЈ) често је практичније користити статистички генерисана

правила која можда нису толико прегледна и језички интуитивна, али се помоћу њих често са мање уложених човек дана може доћи до програма који дају и добре перформансе. Наравно, специфичне области примене нису крајњи домети ових парадигми и истраживања у њиховим оквирима настављају свој прилично комплементаран развој. Познат је IBM-ов тим под вођством Фредерика Јелинека који је развио систем за препознавање говора средином 70-тих заснован на статистичком приступу. Овај тим је постигао успех у време кризе истраживања у области вештачке интелигенције и симболичке парадигме уопште, након Лајтхиловог извештаја 1973. године (Dreyfus 1974) који је добрим делом био инспирисан слабим резултатима машинског превођења и који је поткопао финансирање таквих истраживања барем за читаву деценију. Током периода 1990-1994. у оквиру DARPA MTEval иницијативе, Јелинеков систем за машинско превођење који је био статистички оријентисан и коме језичка структура није играла битну улогу, CANDIDE, није се показао бољим ни од старијег симболичког SYSTRAN-а, нити од у том смислу „опонентског" Pangloss-а[4], што се може објаснити недостатком неопходног синтаксног и семантичког знања тог система. На крају су и CANDIDE и Pangloss постали хибридна решења да би могла да буду довољно робусна и напредна решења. Не постоји уопштени систем машинског учења који је једнако добар у свим областима[3] (Schaffer 1994) и по перформансама значајан у односу на специфична решења. Зато је код машинског учења важно имплицитно предзнање (енгл. bias) које има нека симболичка теорија, односно концептуално знање описано добро одабраним граматичким правилима у ОПЈ. Од половине 90-тих су програми за етикетирање и статистичко парсирање добили додатно на значају простим растом сирове снаге рачунарских ресурса.

2. Језички ресурси и квантитативни приступ

Оно што је заједничко овим парадигмама јесу језички ресурси које користе као што су корпус и лексикон – основни алати рачунарске лингвистике. Дужина корпуса је одређена дужином ниске речи у низу који чини његов текст, а величина његовог лексикона одређује се укупним бројем различитих речи од којих је састављен. N-грам је подниска дужине N неке ниске речи, и може се поновити више пута исти N-грам у тој нисци речи, што представља учестаност тог N-грама у датом тексту као нисци речи. Предмет почетне језичке обраде могу бити елементи документа као уопштеног текста, на пример знаци интерпункције или HTML етикете, односно токени као уопштење појма речи. На сложеност етикетираних корпуса, код којих је свакој речи додељена одговарајућа етикета, посебно утиче број елемената класе етикета, који је обично одређен њеном структуром, што је лепо илустровано у (Džeroski и др. 2000). Структура класе етикета одређује изражајно богатство корпуса, али и отежава само етикетирање, како ручно, тако и програмима за етикетирање. За примену програма за етикетирање и рачунску лингвистику уопште веома је важан дигитални облик корпуса. Пример обичајеног тока развоја дигиталног корпуса се може наћи у документима MULEXT-East[14] пројекта. Етикетирани корпус може да служи као „златни стандард» којим се обучава програм за етикетирање и мери његова тачност, односно мере се његове перформансе у смислу машинског учења. Тренутно се постиже и преко 95% тачности аутоматским морфо-синтаксним етикетирањем.

Поменуте парадигме се разликују у начину описивања језичког модела: код симболичке парадигме су дата експлицитна граматичка, морфолошка и друга правила, док су код квантитативног приступа дата имплицитно задава-

њем примера исправних језичких конструкција (нпр. код етикетираног корпуса) и њиховог значења, као и нумеричких параметара одговарајућег статистичког модела. Код квантитативне парадигме лексикон учествује и као његов директан део, нпр. уз дате учестаности N-грама (такође, за статистичке парсере (примери: [12], [19], [20]) су посебно интересантни TreeBank[18] корпуси где се етикетирају читаве реченице корпуса синтаксним стаблима). Статистички приступ датира још од основа информационе теорије, са најкраћом и највероватнијом хипотезом (Shannon Claude E., Weaver Warren, 1949), и чувеним примером Шенонове игре препознавања следеће речи у реченици (1951, циљ је да буде што мања ентропија речи). Касније, 60-тих година Соломоноф (Solomonoff 1964), Колмогоров, Чаитин, Голд (Gold 1967) и други постављају појмове статистичке индукције, стохастичких контекстно-слободних граматика и идентификације језика. Ноам Чомски је тада ставио примедбу да ниједан познат појам вероватноће реченице нема разумног смисла. Тек је каснији напредак статистичке теорије машинског учења ((Vapnik 1999), (Mitchell 1997)) утицао на боље дефинисање ових појмова и боље практичне резултате. Ако се занемаре грешке у улазу, природни језик обилује проблемима за овакав приступ - пре свега проблемима који потичу од семантичке вишезначности (пример полисемије, „Видео сам девојку са повишеном температуром") или синтаксне вишезначности: нпр. у синтагми „с времена на време" програм може различито етикетирати „на време" као сложену форму или атрибут, или сваку поједину реч као предлог или именицу, или целу синтагму као сложени токен.

У овом тексту се користе већ припремљени корпуси о којима ће у даљем тексту бити више детаља. Познати примери етикетираних корпуса за енглески језик су Браунов (дужине око 1,000,000 тј. 1M речи, настао 60-тих) и Susanne корпус који је настао издвајањем дела Брауновог корпуса, National British Corpus (BNC), WSJ и PTB (пре свега Wall Street Journal чланци од којих је састављен PTB – Penn Tree Bank). Етикетирани корпус може настати (полу)ручним етикетирањем за дати прописани скуп етикета као што је то рађено у оквиру MULTEXT-East[14], где се користио и помоћни програм CLOG којим је припремана лематизација аутоматским учењем за сваку етикету посебно (Erjavec 2005). Етикетирање корпуса се може вршити и над аутоматски генерисаним скупом етикета (у смислу ненадгледаног учења). Неколико примера према подацима из Wortschatz[5] пројекта датих у табели 1 илуструје неке основне особине етикетираних корпуса добијених ручним етикетирањем (M = милиона):

Корпус	**BNC**	**CLEF**	**Wortschatz**
Језик	Енглески	Холандски	Немачки
Дужина	100 M	70 M	755 M
Величина лексикона	25706	21863	74398
Величина класе етикета	344	418	511

Табела 1 – неколико примера корпуса

3. Различити програми за етикетирање и методе машинског учења

У програмима који који користе етикетиране корпусе језички модел је одређен (током фазе обучавања) пре свега лексиконом и класама етикета, а имплицитно и корпусом обучавања и моделом учења који зависи од самог програма и његовог алгоритма. То могу бити датотеке N-грама као код TnT-а, правила преписивања (енгл. rewriting rules – тачније, правила којима се мењају тренутно додељене етикете у односу на етикетирани текст као контекст, слично формалним граматикама) као код Бриловог програма, и друго. Тако схваћен језички модел се разликује од традиционалног скупа формалних граматичких и других

правила која и у овом моделу постоје али су донекле у другачијем, квантитативном облику. N-грами као подниске дужине *N* (узастопних) речи или етикета у корпусу са укупним учестаностима појављивања у референтном корпусу, садрже имплицитну информацију о исправним или бар статистички очекиваним конструкцијама и везама међу речима, али с друге стране, замагљују синтаксну структуру реченице, што за проблем етикетирања до извесне границе није пресудно важно.

Прве успешне имплементације програма за етикетирање су засноване на статистичким методама, и то посебно на методи скривених ланаца Маркова. Неки од најпопуларнијих и најефикаснијих решења као што је TnT и данас користе тај исти метод уз нека побољшања (видети одељак 4.1). Многа побољшања се тичу осетљивости програма за етикетирање на непознате речи (оне које се не налазе у лексикону) које увек узрокују велики пад тачности (и до 50%) уколико у решење нису уграђене додатне модификације (једна од последица плитког квантитативног приступа). Неки програми користе препознавање дела речи или суфикса и префикса (са задатим максималним бројем слова) у лексикону, или прате додатне особине лексема, које могу бити и специфичне за одређени језички модел или језик, или као у Бриловом случају, користе посебна лексичка правила којима се одређује највероватнија етикета непознате речи. Пошто се етикетирање може свести на проблем методе машинског учења (Mitchell 1997), (Nillson 2005) и класификације, временом су настале имплементације засноване на методама које нису статистичке природе. Референтни пример је Брилов програм заснован на учењу правила презаписивања којима се повећава тачност етикетирања. Овај програм је увео правила којима се обухватају поред етикета и речи, док стандардни СЛМ користи само етикете (видети одељак 4.3). Врхунске резултате[1] постигли су програми као што је SVMTool заснован на SVM класификацији (видети одељак 4.5), Станфордов PoS Tagger-a (Stanford NLP Group[12]) који се заснива на Бајесовим мрежама и варијанти методе процене највероватнијег исхода (енгл. Maximum Likelihood Estimation, MLE), и LTAG[21] PoS Tagger-a који је заснован на варијанти вештачких неуронских мрежа. Уместо традиционалног читања „с лева на десно», сви ови програми током учења користе двосмерно учење.

Постоје многе варијанте статистичког учења које нису засноване на СЛМ и сличним моделима. Такав је, на пример, програм MXPOST који користи модел максималне ентропије (видети одељак 4.4). Статистички метод је донекле присутан и у Бриловом програму, али је његова суштина ипак у учењу правила презаписивања етикета. Међу методама које нису статистички оријентисане издваја се учење дрветом одлуке (енгл. Decision Tree Learning) које користи програм Tree Tagger (видети одељак 4.2), и учење меморијом (користи га познати Memory-Based Tagger - MBT[10]) које се своди на (лењо) учење примерима (где није потребно обучавање), поред осталих.

Нова решења и реализације програма за етикетирање се често појављују. Организују се такмичења и конференције[2] где се упоређују на десетине решења, а тренутно се води „рат" готово за сваки промил тачности. Поред тачности, брзина рада програма и његовог обучавања, као и једноставност његове употребе многима значе више него неколико промила тачности. У даљем тексту ће бити кратко описано неколико најпопуларнијих програма за етикетирање базираних на методама надзираног машинског учења.

3.1 Перформансе програма за етикетирање

Различити критеријуми могу бити интересантни приликом оцене рада неког од програма за етикетирање, али је свакако најбитнија оцена перформансе тачност етикетирања. У

овом раду ће у односу на дужину текста грешка етикетирања за дати текст бити дефинисана као број погрешно етикетираних речи, а тачност као број исправно етикетираних речи. Поред ове основне особине, за ове програме може да буде важно и потребно време за учење и етикетирање, али се то у овом тексту не разматра детаљно. Важне су и додатне могућности, опције учења модела и једноставност употребе.

Тестирање перформанси се ради десетоструким унакрсним тестом (енгл. 10-cross validation test) који се често користи и за финије одређивање и учење оптималних вредности параметара модела. Састоји се у томе да се корпус за обучавање издели на 10 делова, и да се онда обука програма врши над 9/10 корпуса, а тестирање над преосталих 1/10. Поступак се понавља за сваку комбинацију партиција „9/10+1/10" за обуку и тестирање чиме се добија 10 вредности грешака етикетирања чија просечна вредност и варијанса одговарају оцени перформанси програма. Финија анализа перформанси би узимала у обзир посебно статистике за речи препознате у лексикону, а посебно за оне које су непознате програму. У овом другом случају може доћи до пада тачности и до 30-50%. Међутим, врхунски резултати за енглески језик су тренутно око 96-97% (State-Of-The-Art[1]) где се при њиховом исказивању не води рачуна посебно о непознатим речима којих има увек око 10% у корпусу за тестирање описаном методом. Сви програми су овде у том смислу равноправно тестирани истим методом евалуације, истим корпусима за учење и помоћу истих партиција десетоструког тестирања. Статистике до којих се долази у овом тексту служе пре свега као показатељ односа различитих програма, а не као мера удаљености од врхунских резултата. Понашање програма са непознатим речима је изражено процентуалним уделом непознатих речи у односу на исправно етикетиране и у односу на све непознате речи.

4. Одабрана решења и програми за етикетирање

Евалуација неколико различитих врста програма за етикетирање се врши над текстовима на српском језику. Програми су изабрани као референтни академски и некомерцијални програми за етикетирање, а додатан разлог за њихов избор је и њихова доступност на вебу. Улазни етикетирани корпус за обучавање као и резултат етикетирања за ове програме може бити вертикалног формата где су свака реч и њена етикета раздвојене белим размаком у својој посебној линији:

```
Bio         Vmps-sman-n---p
je          Va-p3s-an-y---p
vedar       Afpmsnn
i           C-s
hladan      Afpmsnn
aprilski    Aopmpn
dan         Ncmsn—n
.           SENT
```

Пример 1а – вертикалан текстуелни формат етикетираног корпуса

и где неки програми могу имати поред етикете речи наведену и лему у тој истој линији. Текст може бити и хоризонталног формата, где је свака реч сепаратором повезана са својом етикетом, свака реченица се налази у својој посебној линији, а поједине речи су раздвојене белим размаком уместо засебним линијама (Брилов програм користи „/" као сепаратор, док MXPOST користи „_"):

```
Bio/Vmps-sman-n---p  je/Va-
p3s-an-y---p  vedar/Afpmsnn
i/C-s  hladan/Afpmsnn
aprilski/Aopmpn  dan/
Ncmsn—n
./SENT
```

Пример 1б – хоризонталан текстуелни формат етикетираног корпуса

Сваки од ових програма има и свој одговарајући алат за учење којим се за дати корпус за обучавање гради језички модел који је

представљен обично скупом посебних датотека уз расположиве опције учења.

4.1 TnT – Trigrams'n'Tags

TnT[7] (енгл. Trigrams'N'Tags) програм чији је аутор Thorsten Brants, Saarland универзитет у Немачкој, при одељењу за рачунску лингвистику и фонетику, настао је у периоду од 1993-2000. године (Thorsten Brants 1999, 2000). Овај програм користи машинско учење засновано на СЛМ које је случај Бајесовог учења и које представља пример класе алгоритама машинског учења који користе оцену вероватноће добијену ацикличним усмереним графом рачунања (или Бајесовом мрежом) који је дефинисан односом условне зависности међу случајним променљивама као чворовима. Проблем етикетирања се онда своди на проблем класификације таквом мрежом рачунања вероватноће где се тражи највероватнији низ етикета за дати низ речи. СЛМ је Бајесова мрежа којој је скуп чворова подељен на две партиције: скуп скривених догађаја или стања, и скуп посматрања или симбола. Сваком стању одговарају различита посматрања која зависе само од њега у датом низу стања што се описује вероватноћама емитовања симбола, док скривено стање може зависити и од свих претходних скривених стања у низу. Ред модела СЛМ је одређен бројем претходних скривених стања у низу од којих зависи тренутно (на пример, код СЛМ другог реда свако скривено стање зависи само од два суседна претходна). Скупу скривених стања одговарају етикете, а посматрањима речи код проблема етикетирања. Модел СЛМ је одређен расподелама вероватноћа почетних стања и преласка из стања у стање. Основна питања у овом случају су: одредити највероватнији низ скривених стања за дати низ посматрања (проблем декодирања), и како научити параметре модела тако да се максимизује вероватноћа низа посматрања за дати скуп обучавања (проблем учења). TnT је пример успешног статистички оријентисаног (стохастичког) програма за етикетирање, заснованог пре свега на Витербијевим алгоритмом за декодирање СЛМ другог реда, и користи N-граме и интерполацију (енгл. Smoothing) код проблема нормализације током учења Баум-Велчовим алгоритмом, (Rabiner 1989), (Welch 2003).

TnT је пример програма за етикетирање који је једноставан за употребу, брзо учи и изузетно брзо етикетира. За учење му је довољан улазни етикетирани корпус вертикалног формата чији назив датотеке без екстензије се касније користи као назив модела. Свака реченица се мора завршити токеном са етикетом SENT, осим ако се не зада друга етикета опцијом „-st". Овај систем користи неколико врста датотека генерисаних учењем које се могу додатно мењати. Ту је лексикон као датотека вертикалног формата која се састоји од најмање 4 колоне где је свака колона раздвојена белим размаком: у првој је реч (токен), у другој је учестаност у корпусу за обучавање (број понављања), у трећој је етикета, а у четвртој је фреквенција те етикете. Етикета може бити више за задату реч, и иза сваке следи њена учестаност, а збир учестаности етикета једнак је учестаности те речи. Поред ове постоји датотека са N-грамима, списак расположивих етикета и посебна датотека пресликавања етикета која се користи да би се променили називи етикета у излазној датотеци у односу на раније називе у полазном језичком моделу.

Програм не подржава лематизацију. Расположивост и лиценцирање програма се разликује од уобичајеног приступа код програма отвореног кôда. Директним обраћањем аутору електронском поштом се може једино добити приступ за преузимање програма за учење и етикетирање (а можда и изворном коду), а програм је начелно расположив бесплатно за академску

и некомерцијалну употребу. Прављен је и тестиран пре свега за Posix (Unix, Linux) платформе.

4.2 Tree Tagger

Аутор Tree Tagger (TT) програма је Helmut Schmid[6] (Schmidt 1994), Институт за рачунску лингвистику, Stuttgart универзитет у Немачкој. Програм је настао у периоду од 1994-1996. године у оквиру TC пројекта. Tree Tagger није прави представник стохастичких програма за етикетирање у односу на уобичајене програме који користе СЛМ јер се разликује у самом начину учења, док се његово етикетирање такође врши применом Витербијевог алгоритма. За обучавање се користи алгоритам учења дрветом одлуке (какви су ID3 и C4.5) уз употребу модела триграма. Дрво одлуке се након изградње одсеца према правилу информационе добити, што је важно у решавању проблема презасићења учењем (енгл. overfitting) и тиме се обезбеђују оптималне перформансе алгоритма.

TT лексикон се састоји из 3 дела: лексикона пуног облика, лексикона суфикса дужине до 5 карактера, и од претпостављене вредности која се додељује ако реч није нађена у претходним деловима. Ако током етикетирања реч није нађена у лексикону пуног облика ни након промене у мала слова, претражује се лексикон суфикса дрветом одлучивања суфикса. Дрво одлучивања за суфиксе се гради током учења на сличан начин као и претходно описано дрво одлучивања за етикете: за сваку реч која је означена етикетом из класе отворених етикета, карактер по карактер суфикса. Сваки родитељ дрвета има грану претпостављене вредности обележену вероватноћом која сабрана са вероватнаћама осталих грана тог родитеља даје 1, а брише се ако је то једина преостала грана. Листови су означени векторима вероватноће етикета са датим суфиксом.

За етикетирање је довољна улазна датотека и параметарска датотека генерисана током обучавања. Датотека отворене класе је листа могућих етикета за речи које су непознате. Етикетирање текста се може вршити и без лема на пример тако што се у лексикону свим речима задаје иста вредност лексеме, као што је „-”. Учење и генерисање модела је једноставно, мада лексикон и датотеку отворене класе треба посебно припремити пре тога (нпр. посебним скриптом[22], видети одељак 5) поред улазног етикетираног корпуса за обучавање. Модел као резултат обучавања је смештен у бинарну параметарску датотеку. Улазна датотека је вертикалног формата као и излазна датотека. Део једног лексикона је дат у примеру 1:

```
prisloni            Vmia3s-an-
n---e prisloniti
talenta  Ncmsg--n
talenat
gutljaju            Ncmsl--n
gutljaj
nagnut   Vmp--smpn-n---e
nagnuti
Vinstonovo          Aspnsn
Vinstonov           Aspnsa
Vinstonov
```

Пример 1 – део TT лексикона

Tree Tagger подржава и лематизацију за разлику од осталих програма који су овде наведени. Лиценцирање је као и код TnT-а бесплатно за непрофитне (академске) сврхе, и такође није отвореног кôда (осим ако се са аутором не договори другачије), а сам програм је расположив за преузимање без посебног ограничења. Програм је расположив како за Posix, тако и за Windows оперативни систем.

4.3 Brill – Rule Based Tagger (RBT)

Ерик Брил[8] (Brill 1992) је један од пионира учења трансформацијама (Transformation-Based Learning) и творац програма за етикетирање RBT (Rule Based Tagger) заснованим на

таквом моделу учења. Развијен је у периоду од 1992-1994. године и поставио је својевремено многе стандарде. Једним делом је реализован Perl скриптовима јер Брил то сматра идеалним програмским језиком за оваква лингвистичка истраживања због богатства рада са нискама и регуларним изразима, али то је уједно и главни узрок велике спорости учења. Овај систем представља један од основних примера програма за етикетирање који нису засновани на статистичким моделима већ пре свега на инкременталном учењу правила контекстних трансформација. Користе се правила преписивања којима се даје резултат са највећом оценом тачности, односно са најмањом грешком етикетирања. У суштини, етикетирање се врши у две фазе: најпре се иницира почетно стање корпуса који се етикетира према лексикону, уграђеним и лексичким правилима модела. Да би се превазишла ограничења СЛМ где се користе само N-грами над етикетама, Брил уводи правила која се односе и на речи. Онда се у другој фази повећава тачност етикетирања применом контекстних правила модела. Ова правила су слична лексичким, али садрже и другу етикету која замењује („преписује/презаписује“) први - сва правила у суштини мењају етикету и окидају се ако су испуњени неки од типова услова. Учење се састоји из фазе учења лексичких правила, и фазе учења контекстних правила вођеног најмањом грешком етикетирања. У другој фази се итеративно оцењује грешка за свако правило кандидата пре и после трансформације, и онда се најбоље оцењено додаје у скуп правила све док не буде ни једног примењивог правила са оценом грешке изнад задатог прага. Постоји „уграђено“ лексичко правило за речи које нису у лексикону а које почињу великим словом: сматрају се властитим именима (иначе се узимају као обичне именице), након чега се примењује потпрограм за непознате речи. Укратко, суфикси и префикси дужине до 4 карактера се узимају у обзир и траже подречи у лексикону и уче се нова правила ако су нађене.

Подаци који чине научен модел смештају се у текстуелне датотеке. Улазни етикетирани корпус за обучавање се очекује у хоризонталном формату, и генерисани излаз је такав. Учење се одвија у неколико фаза: токенизација (дајући вертикалан формат према Penn Tree Bank стандарду, на пример: „Мачка/именица седи/глагол на/предлог тепиху/именица ./.”), подела етикетираног корпуса на део за учење лексикона и лексичких правила за речи изван лексикона, и други део за учење контекстних правила, и коначно покретање учења лексичких, а потом и контекстних правила. Једно од ограничења овог програма је споро учење које се може контролисати прагом грешке. Треба имати на уму да је програм намењен инкременталном учењу увећавањем и мењањем скупа правила додавањем нових мањих корпуса. Учење се прекида када грешка достигне задати праг који је задат у самом Perl коду, и цео поступак учења је прилично сложен и захтеван у односу на остале алате. Детаљан опис и постпуци за овај и друге алате расположиви су на[22].

Овај програм не подржава лематизацију. RBT са изворним кôдом је расположив бесплатно без посебног ограничења, али уз тип лиценцирања који одговара лиценцама отвореног кôда. Иако постоје имплементације за већину постојећих платформи, укључујући и Java платформу, учење у основном облику није подржано на Windows оперативном систему.

4.4 MXPOST

Adwait Ratnaparkhi, тренутно запослен у Yahoo! предузећу, аутор је MXPOST[11] (скраћено MX) програма који је настао у периоду од 1996-1999. године. Овај програм користи статистички концепт учења и метод модела мак-

сималне ентропије (енгл. Maximum Entropy Principle, скраћено MEP, (Ratnaparkhi 1996)) који под одређеним условима може бити дуалан раније поменутом MLE. Овим концептом учења се максимизује условна ентропија модела учења уместо оцене вероватноће исхода. Већа ентропија модела учења значи да је већа количина информација кодирана у параметре модела. Овај програм (MX) је ближи Бриловом и SVMTool по спорости учења које у њиховом случају ипак за ред величине траје дуже. Програм ради 100 итерација без обзира на перформансе током учења.

Датотека корпуса за обучавање има сличну хоризонталну реченичну структуру као и код Бриловог програма за етикетирање, само што користи знак „_" као сепаратор токена и његове етикете. Излазни формат је истог облика. Научени подаци о моделу се смештају у текстуелне датотеке у посебном директоријуму за тај модел.

Овај програм не подржава лематизацију. Оригинална верзија програма је реализована као Java програм, а расположива је и даље у донекле пробном облику. Бесплатно је на располагању за непрофитне сврхе, слободно је расположив за преузимање са нета и није отвореног кôда.

4.5 SVMTool

Аутори, Jesus Gimenez и Lluis Marquez, направили су SVMTool[9] у периоду од 2000-2004. године у оквиру TALP истраживачког центра за ОПЈ при универзитету Каталоније. Ово је један од програма којима је достигнут „state-of-the-art" резултат на WSJ корпусу, и свакако програм са најбогатијим опсегом опција за учење и утицање на рад самог програма. Подржава поред осталог: двосмерно учење и етикетирање, унакрсну валидацију резултата, посебне језичке изузетке и поправке речника, различите моделе учења, и друго. Ово води уједно и ка главном недостатку овог програма, а то је прилично неефикасно учење. Укратко, идеја је да се секвенцијално кодирају различите особине текста у прозору дате ширине мерене токенима, центрирано око токена који се посматра и претпостављене ширине 7. Ту су како особине речи (да ли садрже велико слово, специјалне знаке, бројеве и слично), тако и N-грами, етикете и друго. Вектор таквих особина се класификује SVM алгоритмом, где свакој класи одговара једна етикета. Вапник је за SVM методу класификације (енгл. Support Vector Macihne, (Vapnik 1999)) још 1963. поставио теоретске основе. Ова метода класификације је врста методе максимално граничне хиперравни јер се тражи хипер раван која раздваја хиперпростор примера у две партиције (полупростора) које одговарају двема категоријама класификације, тако да је удаљеност те хиперравни од најближих тачака података максимална. SVM додатно минимизује грешку класификације уз стандардно нумеричко решавање проблема квадратног програмирања. SVMTool користи једну од најефикасних и најједноставних имплементација, SMO (енгл. Sequential Minimal Optimization[9]), (Platt 2000). SVM методе се лако преводе у нелинеарне употребом нелинеарних језгара (Аизерманов „трик језгром") и представљају једне од најефикаснијих познатих метода класификације.

Улазни корпус за обучавање и резултат су у вертикалном формату, са обичним размаком уместо белог размака. За учење је потребно подесити у текстуелној конфигурационој датотеци за коју су на располагању краћа варијанта шаблона и дужа. Више од два сата корпусу са 2500 речи није било довољно да конвергира са дужим шаблоном, а учење са краћим шаблоном траје тек нешто краће него код Бриловог програма. Напреднија опција употребе подразумева откривање SVM параметара валидацијом над скупом обучавања, али главно питање је да ли је оправдано због

неколико промила неколико пута дуже радити обучавање модела.

Овај програм не подржава лематизацију. Аутори нуде програм под LGPL и FSF лиценцом отвореног кôда у Perl-у који није довољно тестиран под Windows оперативним системом. Слободно је расположив за преузимање са нета, али се програм ослања на SVMLight чији је аутор Thorsten Joachim који је такође отвореног кôда, и за чију се комерцијалну употребу тражи посебна дозвола.

5. Евалуација одабраних програма

Сам поступак евалуације одабраних решења је рађен аутоматизовано коришћењем скрипта који омогућава обављање евалуације кроз следеће кораке:

• улазна датотека корпуса се трансформише у вертикални формат уз помоћ одговарајуће XSLT трансформације; као резултат се добија датотека out1.txt целог корпуса;

• иницијализују се датотеке stat_*TAG*.txt са статистичким бројачима, где *TAG* припада скупу *{TNT, TT, RBT, MX, SVM}* одабраних програма за етикетирање;

• главна петља, скрипта се извршава у 10 итерација за променљиву *n* од 0 до 9

• помоћу скрипта cross-tab10.sh припремају се корпус за учење learn.txt и корпус за тестирање test.txt као n-те „9/10+1/10” партиције целог корпуса;

• за сваки програм *TAG* за етикетирање се одговарајућим AWK скриптовима припремају потребни трансформисани корпуси, покреће се учење, а затим и етикетирање неетикетираног корпуса за тестирање;

• резултат етикетирања *TAG* програма се смешта у вертикалном формату у датотеку test_*TAG0*.txt, а пошто је раније припремљен етикетирани корпус за тестирање у датотеци test_*TAG*.txt; вертикални формат омогућава лако поређење добијеног етикетирања са полазним;

• за сваки програм *TAG* се помоћним скриптовима unknown.sh и unknown.awk додају бројачи непознатих речи (лоше етикетиране речи се траже у лексикону учења, па ако нису ту, сматрају се непознатим речима);

• помоћни скриптови report1.awk и report2.awk праве агрегиране статистике;

• бришу се датотеке које се формирају учењем, као и разне помоћне датотеке;

Овакво аутоматизовање поступка тестирања и евалуације програма за етикетирање омогућава лако проширивање другим програмима за етикетирање које треба евалуирати и упоређивање са већ испробаним програмима, смањује број грешака у поступку и у рачунању статистичких особина, и пре свега представља погодно радно окружење за испробавање ових програма. Посебно је интересантна могућност интегрисања различитих програма и њихових параметара и модела обуке, па чак и различитих класа етикета и корпуса. На пример, у раду (Jakub Zavrel, Walter Daelemans 2000) методом Bootstrapping се постиже учење нове класе етикета мањим бројем примера, али се такође и постиже тачност етикетирања која надмашује бар мало сваки од појединачних програма. Један сасвим једноставан начин да се побољшају перформансе етикетирање употребом више различитих програма или њихових подешавања је методом гласања – бира се етикета коју већи број програма изабере или оцени најбоље. Даљи приступ би подразумевао избор према неком машински наученом искуству у односу на избор програма, класа етикета, параметара и улаза.

Сви програми су инсталирани без већих проблема према упутствима приложеним уз њих. Укупно време трајања скрипта за обучавање и етикетирање је тестирано на рачунару са процесорском платформом Intel(R) Core™2 Duo T7700 2.40GHz, што је даље поменуто у овом тексту. Изабран је Linux Fedora FC8 као ОС платформа за тестирање, због тога што многи програми (Tree Tagger, RBT) не подржавају учење на не-Posix платформама или имају нека друга ограничења. Ипак, глав-

ни разлог за избор ове платформе је удобан рад са нискама, регуларним изразима, могућност прослеђивања токова команди и бројни згодни алати за рад са текстом као што је пребројавање речи или линија, упоређивање разлика, моћни скрипт језици и друго. Сви развијени скриптови се могу преузети са мрежне странице[22], где се такође могу наћи референтни радови с детаљнијим објашњењима, као и комплетно окружење са скриптовима.

5.1 Корпуси

Основна особина етикетираног корпуса за обучавање је његова дужина, односно број етикетираних речи у низу који чини текст корпуса. Међутим, величина скупа речи које чине лексикон сачињен над корпусом може бити статистички битнија јер се према Зипфовом закону расподела речи не мења битно када се једном постигне довољна дужина текста на природном језику. Веома важна особина корпуса је и број елемената класе етикета над којом се корпус проучава. Класа етикета може бити сложена и велике кардиналности, што највише утиче на перформансе програма за етикетирање. Кардиналност класе етикета зависи како од концепта етикетирања, односно од тога шта се етикетирањем жели добити, тако и од самог језика. Примера ради, српски језик је свакако захтевнији у том смислу у односу на енглески језик ако се, на пример, етикетирају и падежи. Поред тога, нису сви програми за етикетирање на исти начин осетљиви на квалитет информација у самом корпусу. Известан мањи број грешака може битно да утиче на перформансе програма (на пример, погрешно унета етикета у корпусу за обучавање, или знак интерпункције који недостаје на крају реченице у корпусу за обучавање касније могу произвести грешке етикетирања). Неке мање грешке могуће је уочити и ручно исправити ако током провере сам скрипт јави грешку, неке је могуће предвидети и исправити једноставним лексичким обрадама - али многе остају „скривене“. Сви програми су тренирани са претпостављеним (default) вредностима параметара учења, без икаквих измена (и у том смислу равноправни). Наравно, код неких програма је могуће постићи и боље резултате, али уз додатни труд и специфична прилагођавања.

У оквиру ове евалуације коришћена су две врсте датотека за изградњу корпуса за обучавање. Једна врста је у структуираном XML формату, уз коришћење CES[15] стандарда, где је, укратко:

- свака реч дата XML етикетом mw са атрибутима:
  - id који је јединствено идентификује,
  - lex који даје лексему или токен, реч,
  - lemma који даје лему дате речи,
  - tag који даје етикету дате речи.
- свака реченица је дата XML етикетом seg и атрибутом id који је јединствено идентификује
- свака страна је дата XML етикетом p, а одељак XML етикетом div

Пример овакве датотеке и њене структуре дат је испод у примеру 2 као део садржаја датотеке 02HP-SR-Lemma.xml, а оваква датотека се онда одговарајућом XSLT трансформацијом из примера 3 претвори у текстуелни облик потребан за даљи рад:

```
<Annotation type=”morpho”>
  <body>
    <div>
      <head>
        <mw id=”mw__1 “ lex=”ZAKLJUCAK”
lemma=”ZAKLJUCAK” tag=”?”/>
      </head>
      <p>
      <seg id=”n1”>
        <mw id=”mw_1_1 “ lex=”Na” lemma=”na”
tag=”PREP+p4”/>
        <mw id=”mw_1_2 “ lex=”međunarodnom”
lemma=”međunarodni” tag=”A”/>
        <mw id=”mw_1_3 “ lex=”planu” lemma=”plan”
tag=”N”/>
        <mw id=”mw_1_4 “ lex=”poslednjih” lemma=”poslednji”
tag=”A”/>
        <mw id=”mw_1_5 “ lex=”decenija” lemma=”decenija”
```

```
tag="N"/>
      <mw id="mw_1_6 " lex="preduzeti" lemma="preduzeti"
tag="V+Perf+Tr"/>
      <mw id="mw_1_7 " lex="su" lemma="jesam"
tag="V+Imperf+It+Iref"/> ...
    </seg>
    <seg id="n2"> ...
```

Пример 2 – XML структура корпуса 1 и корпуса 2

```
<?xml version="1.0" encoding="UTF8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text" omit-xml-declaration="yes"
indent="no"/>
 <xsl:template match="//seg">
  <xsl:for-each select="mw">
   <xsl:value-of select="@lex"/>~
   <xsl:value-of select="@tag"/>~
   <xsl:value-of select="@lemma"/>~
  </xsl:for-each>*SENT*
 </xsl:template>
</xsl:stylesheet>
```

Пример 3 – data.xsl

Други тип датотека коришћених за евалуацију припремљен је у складу са TEI[16] препорукама. И оне користе XML нотацију, али им је структура нешто другачија: XML етикете и атрибути нису исти, а лексеме нису дате као атрибути већ као садржај XML етикете w. TEI стандард подразумева и обавезно заглавље са библиографским подацима, подацима о кодном распореду, структури датотеке и многим другим мета-подацима, али и многе друге помоћне структуре (нпр. за опис структуре етикета).

```
<TEI.2 id="Osr" lang="sr">
 <teiHeader creator="CK" status="update" ... id="Osr.
teiHeader">
  <fileDesc>
    <titleStmt> ... </fileDesc>
  <encodingDesc>
    <projectDesc> ... </encodingDesc>
  <revisionDesc> ... </revisionDesc>
</teiHeader>
<text lang="sr" id="Osr.">
 <body>
  <div id="Osr.1" type="part" n="1">
  <div id="Osr.1.2" type="chapter" n="1">
   <p id="Osr.1.2.2">
    <s id="Osr.1.2.2.1">
     <w lemma="biti" ana="Vmps-sman-n---p">Bio</w>
     <w lemma="jesam" ana="Va-p3s-an-y---p">je</w>
     <w lemma="vedar" ana="Afpmsnn">vedar</w>
     <w lemma="i" ana="C-s">i</w>
     <w lemma="hladan" ana="Afpmsnn">hladan</w>
     <w lemma="aprilski" ana="Aopmpn">aprilski</w>
     <w lemma="dan" ana="Ncmsn--n">dan</w>
     <c>;</c>
... <!-- pb n=283 -->
    </p>
   </div>
  </body>
</text>  </TEI.2>
```

Пример 4 – XML структура корпуса 3 у складу са TEI

У примеру 4 приказана је оваква датотека као део садржаја датотеке oana-sr.xml. Одговарајућа XSLT трансформација која производи излаз исти као и за претходну врсту датотека користи се на сличан начин. Након тога следе даље обраде awk скриптовима којима се добија коначан вертикални облик етикетираног корпуса за обучавање.

За евалуацију су коришћена три корпуса који ће у тексту бити даље помињани као корпус 1, корпус 2 и корпус 3 (полазне датотеке од којих су направљени су расположиве за преузимање[22]):

- корпус 1 је настао од датотеке `01HP-SR-Lemma.xml` која представља део документа „Хелсиншке свеске бр. 15, националне мањине и право"[0] у CES формату којег имају и датотеке `02HP-SR-Lemma.xml` и `03HP-SR-Lemma.xml`
- корпус 2 је настао конкатенацијом датотека `01HP-SR-Lemma.xml, 02HP-SR-Lemma.xml, 03HP-SR-Lemma.xml`, и додатно датотека `Radiodif-SR-lemma.xml` и `Radionica-SR-lemma.xml` које садрже српски Закон о радиодифузији[0] и материјале са УНДП радионице; све су у CES формату и величине 1-2MB
- корпус 3 је настао од датотеке oana-sr.xml у TEI формату Орвелове „1984", величине око 4.5MB

Семантичка структура класа етикета, за које се често користи и термин MSD од енглеског morpho-syntactic descriptions, а које се користе у корпусима 1 и 2 није иста као и корпусу 3. Наиме, у корпусима 1 и 2 су кодиране у основи само врсте речи и сваком токену је придружена лема, док је корпус 3 кодиран много детаљније[22], у складу са морфосинтаксним описом развијеним у оквиру MULTEXT-East пројекта. Корпус 3 се састоји од текста Орвеловог романа „1984" (Krstev Cvetana и др. 2004) који је као паралелни вишејезични корпус развијен у оквиру европског MULTEXT-East пројекта[13]. По TEI стандарду MSD су задате библиотекама структура својстава (енгл. feature structure). Примера ради, у оквиру поменутог MULTEXT-East пројекта[14] глагол је описан са 15 својстава, од којих су прва два са могућим вредностима дата у примеру 5:

Verb (V)

PoS	Type	VFrm	Tens	Pers	Numb	Gend	Voic	Neg	Def	Cltc	Case	Anim	Clt2	Aspt
****	****	****	****	****	****	****	----	----	----	----	----	----	----	----

P	ATT	VAL	C	EN	RO	SL	CS	BG	ET	HU	HR	SR
1	Type	main	m	x	x	x	x	x	x	x	x	x
		auxiliary	a	x	x	x	x	x	x	x	x	x
		modal	o	x	x	x	x		x		x	x
		copula	c		x	x	x				x	x
		base	b	x								
2	VForm	indicative	i	x	x	x	x	x	x	x	x	x
		subjunctive	s		x							
		imperative	m		x	x	x	x	x	x	x	x
		conditional	c	x		x	x		x	x	x	x
		infinitive	n	x	x	x	x		x	x	x	x
		participle	p	x	x	x	x	x	x		x	x
		gerund	g		x			x	x			
		supine	u			x			x			
		transgressive	t				x					
		quotative	q						x			

...

Пример 5 – MSD структура

У оквиру тог пројекта развијен је и посебан програм за етикетирање ToTaLe[14] (Erjavec 2005), а користи се упоредо и TnT и MBT.

6. Резултати

Резултати добијени претходно описаним поступком за сва три корпуса представљени су табелама 2а, 2б и 3 у којима се ознака * користи за резултате над познатим речима, ознака ** за резултате над непрепознатим речима, док се ознака *** односи на скуп обучавања. Дужина, величина и број лема се односе на одговарајуће особине корпуса изражене бројем различитих речи, где се К означава 1000 речи. Табела 2а даје општи преглед тестирања и корпуса употребљених за све програме, док табела 2б даје оцену перформанси сваког од програма независно од класе речи заједно са односом непознатих речи међу неуспешно етикетираним речима (**, мањи проценат је бољи). Табела 3 боље показује понашање програма са непознатим речима јер се ту даје удео лоше етикетираних непознатих речи у односу на све непознате речи у тест корпусу (**), а за познате речи је дат однос исправно етикетираних речи у односу на остатак корпуса (*, већи проценат је бољи). Оваква табела боље говори о резултатима етикетирања над класама речи у односу на претходну, али зато има мање смисла у општој процени перформанси програма. Корпуси 2 и 3 се незнатно разликују по броју лексема, али зато се значајно разликују по броју етикета.

Програм/корпус		**Корпус 1**	**Корпус 2**	**Корпус 3**
	дужина	7.5К	75К	105К
	величина	2.5К	11К	18К
	бр. лема	1.6К	5К	7.6К
	бр. етикета	79	129	908
Укупно трајање теста		22мин.	9ч : 50мин.	5 дана, 1ч : 29мин.
*** просек	лексема	2290–2378 (2335)	9766–10952 (10368)	16550-17372 (16919)
	етикета	73 – 79 (77)	120 – 129 (126)	840 – 897 (884)

Табела 2а – основне особине корпуса и тестова

Програм / корпус		**Корпус 1**		**Корпус 2**		**Корпус 3**	
		%	**% ****	**%**	**% ****	**%**	**% ****
TT	просек	85.44	64.93	**94.39**	33.30	79.65	35.05
	ст. дев.	3.90	3.87	1.86	20.25	1.92	1.85
SVM	просек	84.93	64.70	94.27	38.02	85.24	34.67
	ст. дев.	3.60	5.51	1.72	22.61	1.87	2.27
TnT	просек	**86.18**	67.65	94.11	37.42	**85.47**	32.26
	ст. дев.	3.60	4.33	1.65	21.85	1.75	2.19
MX	просек	82.69	54.01	92.78	29.43	82.07	28.62
	ст. дев.	3.84	2.49	1.79	16.93	1.79	16.93
RBT	просек	84.96	**82.15**	93.14	**47.24**	85.20	**37.96**
	ст. дев.	4.34	4.32	3.21	26.29	1.95	1.97

Табела 2б

Овде није посебно разматрана брзина етикетирања, али је дато трајање учења заједно са евалуацијом (укупно трајање теста). TnT је свакако шампион и брзине етикетирања и учења, а и његове перформансе у смислу тачности јесу најбоље, као и једноставност употребе.

Програм	Корпус 1		Корпус 2		Корпус 3	
	*** %**	**** %**	*** %**	**** %**	*** %**	**** %**
TT	98.37	56.71	97.53	**71.49**	91.78	36.79
SVM	98.29	55.18	97.69	67.17	93.98	54.60
TnT	**98.54**	**57.50**	97.57	67.17	93.86	**58.36**
MX	97.43	57.01	96.48	69.09	92.06	54.26
RBT	**99.10**	**43.96**	**97.97**	**48.17**	**94.24**	50.33

Табела 3

Скрипт за раздвајање целог корпуса на партиције за учење и тестирање чита корпус редом, у једнаком броју реченица по партицији. То можда није идеално на први поглед, и могло би се побољшати насумичним бирањем реченица као што то ради divide-in-two-rand.prl код RBT-а, или коришћењем корпуса веће дужине. У овде описаним тестовима су неке од партиција за тестирање у другом корпусу практично остале без непознатих речи, што је довело до велике стандардне девијације за непознате речи – с друге стране, то јесте реалнији тест. Торстен[7] даје резултате са стандардном девијацијом 0.13 за Penn Treebank (0.76 за Susanne Corpus на енглеском, 0.29 за NEGRA корпус на немачком), што показује стандардну девијацију упоредиву са овде добијеним резултатима. Наравно, овакво поређење са статистичким резултатима других није потпун начин доказивања исправности закључака у овом тексту, али описује добро основне особине програма за етикетирање. Међу наведеним радовима у овој области постоје и детаљнији докази о особинама програма за етикетирање и њиховим перформансама.

7. Закључак

У (Erjavec 2005) су представљени резултати за два програма за етикетирање дати овде у табели 4 и заснивају се такође на Орвеловој "1984"[13] као MULTEXT-East ресурсу. Ови резултати су упоредиви са овде добијеним резултатима за TnT програм како за непознате речи (добијен је можда и нешто бољи резултат), тако и за познате речи. Слични резултати управо на корпусу 3 имају још више смисла као прилог поређењу програма на овај начин. Иако је већи број етикета нарушио перформансе корпуса 3 у односу на корпус 2 где су достигнути максимални (практично врхунски резултати), овај тест је сигурно реалнији и употребљивији.

	TnT	MBT
Познате	93.55%	93.58%
Непознате	60.77%	44.45%

Табела 4

Судећи према табели 2, програм TnT се показао као најбољи за корпусе 1 и 3, док је програм Tree Tagger „победио" у случају корпуса 2, али за промил. Такође, ако погледамо табелу 3, стиче се утисак да се RBT генерално боље сналази од других над корпусима 1 и 2, а Tree Tagger над корпусом 3, за непознате речи. Из те табеле би се могло

закључити да Брилов RBT има боље резултате него други у етикетирању познатих речи, али то треба узети с резервом као оцену перформанси јер су разлике сувише мале. Можда би неки општи закључак био да се Tree Tagger донекле добро сналази са мањим бројем етикета, али да TnT у супротном односи лаку „победу". На крају, реч је о заиста малим разликама где су и резултати за SVM-Tool такође близу.

Сваки од ових програма се може до извесне границе додатно подешавати да би се добили нешто бољи резултати у чему SVMTool предњачи по могућностима, али чије учење постаје превише споро у зависности од корпуса. BNC корпус је приближно 1000 пута дужи корпус од корпуса 3, а има скоро три пута мање етикета и тек нешто мањи број токена од њега, што је још важније - дакле, на перформансе пресудно утиче број етикета. Корпус, какав је корпус 2, са мањим бројем етикета (или чак мање дужине) је идеалан за истраживања на појединим програмима и методама, као и за упоређивање њихових перформанси, али не и за крајњу експлоатацију програма за етикетирање. Један од неистражених изазова је откривање управо тих крајњих домета перформанси и накнадно поређење ових програма на српском језику.

Оптимална величина скупа обучавања (корпуса) једно је од најбитнијих достигнућа статистичке теорије машинског учења - показује се да зависи само од жељене грешке и вероватноће учења, као и од величине и структуре простора простора хипотеза (детаљи о томе се могу наћи у (Vapnik 1999) и (Nillson 2005)). Већи корпус (типа корпуса 3) би свакако био потребан за прецизнија испитивања и нешто боље перформансе, уз вођење рачуна о презасићењу учења (overfitting, превелик скуп обучавања може нарушити перформансе и способност генерализације) што овде наведени програми имплицитно решавају сваки својим посебним механизмима.

(страницама је приступано од новембра 2008-2010. године)

0 http://www.anem.rs/download/files/cms/attach?id=4

ISBN 86-7208-065-3 http://www.helsinki.org.rs/serbian/doc/sveske15.zip

1 State-of-the-Art results: http://aclweb.org/aclwiki/index.php?title=POS_Tagging_(State_of_the_art)

2 Tagger Competitions and Conferences, Resources:

http://www.aclweb.org http://www.lrec-conf.org

http://alias-i.com/lingpipe/web/competition.html

http://ltrc.iiit.ac.in/nlpai_contest07/cgi-bin/index.cgi

3 http://www.no-free-lunch.org

4 http://www.lti.cs.cmu.edu/Research/Pangloss/

5http://wortschatz.uni-leipzig.de/~cbiemann/pub/2007/BiemannGiulianoGliozzoRANLP07.pdf

6 TT http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html

7 TNT http://www.coli.uni-saarland.de/~thorsten/tnt/

http://coli.uni-sb.de/~thorsten/tnt/

8 Brill http://www.cst.dk/download/tagger/

http://www.tech.plym.ac.uk/soc/staff/guidbugm/software/RULE_BASED_TAGGER_V.1.14.tar.Z

http://www.ling.gu.se/~lager/Home/brilltagger_ui.html

9 SVM: http://www.lsi.upc.edu/~nlp/SVMTool/

http://svmlight.joachims.org/

10 MBT: http://ilk.uvt.nl/mbt/

11 MXPOST: ftp://ftp.cis.upenn.edu/pub/adwait/jmx/

http://www.inf.ed.ac.uk/resources/nlp/local_doc/MXPOST.html

12 Stanford NLP Group – statistical Parser

http://nlp.stanford.edu/software/lex-parser.shtml

13 Orwell http://nl.ijs.si/ME/bib/mte-nlprs01.pdf

14 MULTEXT-East: http://nl.ijs.si/ME/V3/msd/

http://nl.ijs.si/et/

http://nl.ijs.si/et/talks/SFB441/tue-slides/

http://langtech.jrc.it/Documents/LTC-2005_Multilingual-corpus-compilation_Erjavec-et-al.pdf

15 CES: http://www.cs.vassar.edu/CES

16 TEI: http://www.tei-c.org http://bcdlib.tc.ca/tools-standards.html

17 Stanford Tagger http://nlp.stanford.edu/software/tagger.shtml

18 treebanks: PTB http://www.cis.upenn.edu/~treebank/

ICE http://www.comp.leeds.ac.uk/amalgam/tagsets/ice.html

19 Michael Collins 1998. PhD http://people.csail.mit.edu/mcollins/

ftp://ftp.cis.upenn.edu/pub/mcollins/PARSER.tar.gz

20 Dan M. Bikel http://www.cis.upenn.edu/~dbikel/

21 LTAG POS Tagger http://www.cis.upenn.edu/~xtag/spinal/

22 download: http://users.hemo.net/shoom/taggers.tar.gz

http://users.hemo.net/shoom/tag.pdf

Литература

Brill Eric. 1992. A simple rule-based part of speech tagger. Speech and Natural Language: Proceedings of a WorkshopHeld at Harriman, New York.Morgan Kaufmann Publishers, Inc., San Francisco, Californi, pp. 112–116.

Dreyfus L. Hubert, Haugeland John. 1974. An Exchange On Artificial Intelligence.

http://www.nybooks.com/articles/9452

Džeroski Sašo, Erjavec Tomaž, Zavrel Jakub. 2000. Morphosyntactic Tagging of Slovene: Evaluating Taggers and Tagsets. 2nd International Conference on Language Resources & Evaluation (LREC), pp. 1099-1104.

Erjavec Tomaž, Ignat Camelia, Pouliquen Bruno, Steinberger Ralf. 2005. Massive multi lingual corpus compilation: Acquis Communautaire and totale. Proc. of 2nd Language & Technology Conference, pp. 32-36.

Gold E. Mark. 1967. Language identification in the limit, Information and Control, 10:447—474. http://www.isrl.uiuc.edu/~amag/langev/paper/gold67limit.html

Jakub Zavrel, Walter Daelemans. 2000. Bootstraping a Tagged Corpus through Combination of Existing Heterogeneous Taggers. Proceedings of the second international conference on language resources and evaluation (LREC), pp. 17-20.

Krstev Cvetana, Vitas Duško, Erjavec Tomaz. 2004. Morpho-Syntactic Descriptions in MULTEXT-East - the Case of Serbian. In Informatica, No. 28, The Slovene Society Informatika, Ljubljana. pp. 431-436. http://www.matf.bg.ac.rs/~cvetana/biblio/mtesr-inform04.pdf

Leech G. 1987. Garside R. and Sampson G. The Computational Analysis of English: A Corpus-based Approach. London: Longman http://www.comp.lancs.ac.uk/computing/research/ucrel/claws/

Mitchell M. Tom. 1997. Machine Learning. McGraw-Hill. *ISBN 0-8493-1232-9* Nillson J. Nils. 2005. Introduction To Machine Learning. Stanford unpublished textbook draft http://ai.stanford.edu/people/nilsson/mlbook.html

Platt C. John. 2000. Fast Training of Support Vector Machines using Sequential Minimal Optimization. MIT Press, ISBN:0-262-19416-3 pp. 185-208 http://research.microsoft.com/~jplatt/abstracts/SMO.html

Rabiner R. Lawrence 1989. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proc. IEEE 77(2), pp. 257—286.

Ratnaparkhi Adwait. 1996. Maximum Entropy Model for Part-Of-Speech Tagging. Proceedings of the Empirical Methods in Natural Language Processing Conference, University of Pennsylvania, pp. 133--142.

Schaffer Cullen. 1994. A conservation law for generalization performance. International Conference on Machine Learning, pp.295-265.

Schmidt Helmut. 1994. Probabilistic part-of-speech tagging using decision trees. In Proceedings of the International Conference on New Methods in Language Processing, Manchester, UK, pp. 44—49

Shannon Claude E., Weaver Warren. 1949. The Mathematical Theory of Communication. The University of Illinois Press, Urbana, Illinois, ISBN 0-252-72548-4

Solomonoff J. Ray. 1964. A Formal Theory of Inductive Inference. Information and Control, Part I: Vol 7, No. 1, pp. 1–22. http://world.std.com/~rjs/pubs.html

Thorsten Brants. 1999. Cascaded Markov Models. Proceedings of 9th Conference of the European Chapter of the ACL (EACL-99), Bergen, pp. 118-125.

Thorsten Brants, 2000. TnT - A Statistical Part-of-Speech Tagger. 6th Conference on Applied Natural Language Processing, 224-231. Seattle, Washington.

Vapnik N. Vladimir. 1999. The Nature of Statistical Learning Theory, ISBN 978-0-387-98780-4

Welch R. Lloyd. 2003. HMM and the Baum-Welch Algorithm, IEEE IT Society Newsletter. http://www.itsoc.org/publications/newsletters/past-newsletters/itNL1203.pdf/view