T A R T U Ü L I K O O L

MATEMAATIKA-INFORMAATIKATEADUSKOND

Arvutiteaduse instituut

Informaatika eriala

Kadri Kajaste

Eestikeelsete tekstide statistiline morfoloogiline ühestamine TreeTaggeriga

Bakalaureusetöö

Juhendaja: PhD Kaili Müürisep

Autor: …… "….." mai 2006

Juhendaja: …… "….." mai 2006

Professor: …… "….." mai. 2006

TARTU 2006

Sisukord

Sissejuhatus

Teksti morfoloogilisel töötlemisel määratakse igale sõnale ja märgile lauses esmalt kõikvõimalikud sõnaliigi interpretatsioonid. Seejärel valitakse mitmest erinevast variandist välja just see õige, antud kontestis sobiv sõnaliik- sellist protsessi nimetataksegi morfoloogiliseks ühestamiseks. Kuna keeletöötluses on tegemist suurte tekstikorpustega, siis on vaja ka ühestamist arvutiprogrammide abil automatiseerida. Kuid senini eesti keele jaoks kasutatud automaatsed ühestajad (Puolakainen, 2001; Kaalep,Vaino, 1998) ei ole kaugeltki täiuslikud ja töö eesmärgiks oligi katsetada veel ühe meetodi sobivust eesti keelele. Tuleviku nägemuses saaks kasutada mitut meetodit koos, et saavutada täiuslikum tulemus.
Töös vaadeldakse katset rakendada H. Schmid-i poolt 1994 a. väljatöötatud programmi TreeTagger (Schmid, 1994) eestikeelsete tekstide automaatsel töötlemisel. TreeTagger töötab otsustuspuude läbimise meetodit rakendades. Töö põhiosa seisnes vajalike tekstide töötlemises TreeTaggerile sobivale kujule ja siis erinevate katsete teostamises.

Töö koosneb kuuest peatükist. Esimeses peatükis antakse ülevaade erinevatest maailmas kasutatud ühestamismeetoditest ja eesti keelele seni rakendada püütud meetoditest. Teises peatükis tutvustatakse lähemalt otsustuspuude meetodit, mille alusel TreeTagger töötab. Kolmandas peatükis saab selgeks statistiliste meetoditega ühestamiseks vajalike eestikeelsete korpuste olukord. Neljas peatükk tutvustab täpsemalt TreeTaggerit ja tema kasutusvõimalusi. Viiendas peatükis tuleb juttu testide töötlemisest programmile sobivale kujule ja viimases peatükis ühestamise tulemustest.
Lisana on esitatud ka töö erinevate etappide parimad tulemused (cd plaadil).

1. Tekstide morfosüntaktiline analüüs

1.1 Ühestamine

Tekstide automaatsel töötlusel ja analüüsil on mitu erinevat etappi. Esmalt on vaja tekstid töödelda kasutatavale programmile sobivale kujule, eraldada laused ja sõnad. Siis tuleb eelmise etapi tulemus morfoloogiliselt analüüsida. Sellele omakorda järgneb süntaktiline analüüs. Keele morfoloogilisel analüüsimisel leitakse kõikide tekstis esinenud sõnade ja märkide morfoloogiline informatsioon, s.t kas tegemist on nimisõna, tegusõna või mõne muu variandiga. Kõikide loomulike keelte morfoloogilisel märgendamisel tekitavad probleeme sõnade mitmesused: programm ei suuda otsustada, millise sõnavormiga on tegemist, kuna sõna üldpilt, mille alusel otsustusi tehakse, on sama. Kõige lihtsam näide oleks sõna „või“- inimesele on lihtne lause konteksti alusel aru saada, kas tegemist on toiduaine, sidesõna või verbivormiga. Kuid teatavasti programm inimese moodi mõelda ei suuda, samuti ei vaata morfoloogiline analüsaator sõna konteksti, vaid piirdub ainult sõnavormi endaga. Seega leiab morfoloogiline analüsaator kõik võimalikud sõnaliikide variandid, mis antud sõnakujule sobida võiksid (Antud näites siis nimisõna, verb ja sidesõna, nimisõna on omakorda mitmene nimetava ja omastava käände vahel). Morfoloogiline ühestamine seisneb sõna mitmete erinevate morfoloogiliste tõlgenduste vahel õige valimises, arvestades konteksti. On kaks võimalust morfoloogiliste mitmesuste tekkeks: võimalus et mitmesus seisneb kahe üheliigilise sõna erinevas tähenduses või muutevormis (nim *raha* - om *raha* - os *raha*) . Teisel juhul on ühesuguse kirjapildi taga ka lisaks erinevale tähendusele veel erinev sõnaliik. Lihtsa näitena võiks tuua sõna „kuid“, mis võib olla nii sidesõna tähenduses kui ka nimisõna *kuu* mitmuse vorm.

1.2 Ühestamise meetodid

1.2.1 Üldistest ühestamise meetoditest

Maailmas on kõige enam uurimistööd tehtud inglise keele automaatse töötlemise kohta ja seega on ka erinevad meetodid arendatud eelkõige inglise keelele mõeldes. Laialt on kasutuses statistikal põhinevad Markovi mudeli baasil ühestajad. Bigramm ühestaja vaatab treeningetapil vaadeldavast sõnast ühe võrra ees olevat sõna, tõenäosused moodustuvad paarikaupa. Trigramm mudel aga tegeleb kolmikutega, vaadates sõnast kahe võrra ettepoole. Markovi mudeli baasil ühestajatest on eriti edukad just Markovi peitmudelil põhinevad ühestajad, sest need võimaldavad häid tulemusi saavutada ka väikestel treeningkorpustel (El Beze ja Merialdo, 1999).

Vähem on levinud reeglipõhised ühestajad. Selle meetodi puhul moodustatakse inimesele arusaadavad keelereeglid, mida järkjärgult rakendades välistatakse valed märgendid. Tuntuim reeglipõhine ühestamisformalism on kitsenduste grammatika (Voutilainenen, 1999)

Leiduvad ka juhtumi-põhist meetodit (ingl. k *case-based method*) kasutavad ühestajad. Juhtumi-põhine õppimisparadigma põhineb hüpoteesil, et tunnetuslike ülesannete sooritus (antud juhul loomuliku keele töötlus) baseerub uute olukordade analoogial varasemate kogemuste säilitatud esitusega, mitte varasemate juhtumite alusel loodud reeglite baasil. (Daelemans, 1999:291)

Närvivõrkudel baseeruvad ühestajad kasutavad näiteid võrgu treenimiseks. Seda tehakse korduvalt üle kõigi näidete itereerides, võrreldes iga näite puhul võrgu poolt ennustatavat väljundit õige väljundiga ja muutes võrgu sõlmedevahelisi kaalusid vastavalt esitluse kasvule. Samaaegselt hoitakse ühenduste kaalude maatriksit ja unustatakse kasutatud näited.(Daelemans, 1999:300)

1.2.2 Ühestamise meetodid eesti keele jaoks

Eesti keele morfoloogilisel ühestamisel on seni kasutatud kahte põhilist meetodit: reeglipõhist ja statistilist. Esimesel juhul koostatakse lingvistiliste reeglite komplekt sõnade järgnevuse vms alusel sõnavormi määramiseks, eesti keele

morfoloogilisel ühestamisel kasutatakse kitsenduste grammatikat (Puolakainen, 2001).

Kitsenduste grammatika on oma loomult reduktsionistlik, s.o analüüsi alguses lisatakse igale sõnale kõik võimalikud analüüsivariandid ja seejärel hakatakse konteksti mittesobivaid eemaldama. Seetõttu nimetataksegi selles formalismis kasutatavaid reegleid kitsendusteks. Iga kitsendus esitab mõnda spetsiifilist keelereeglilaadset fakti, üldisem grammatikareegel kujuneb alles nende koosmõjust. Grammatikasse on võimalik lisada ka heuristilisi reegleid, mis kirjeldavad pigem keelesüsteemi tendentse kui üheselt tõeseid keelereegleid. (Müürisep, 2000) Kitsenduste grammatikat kasutav ühestaja ühestab küll väga õigesti, kuid jätab paljud sõnad mitmeseks. Ühestaja töö hindamiseks kasutatakse kahte mõistet: saagis ja täpsus. Saagis[1] (*recall*) näitab leitud õigete analüüside suhet võrreldes käsitsi leitud analüüsidega. Täpsus (*precision*) näitab õigete analüüside osakaalu kõigist leitud analüüsidest (Müürisep, 2000). Eesti keele kitsenduste grammatika rakendamisel morfoloogilisele ühestamisele oli saagis ligikaudu 97-98% ja täpsus 83-86% juures.

Teisel juhul püütakse ühestamiseks abi saada teksti statistilisest analüüsist: leitakse sõnade esinemissagedused kontekstides ja peaaegu ei kasutatagi lingvistilisi reegleid. Statistilise ühestamise puhul on üheks enamlevinud meetodiks kujunenud Markovi peitmudel (ingl k *Hidden Markov Model* - HMM). HMM-i rakendamisel eesti keelele kasutati seda tema puhtal klassikalisel kujul. (Kaalep,Vaino, 1998)
Morfoloogilisel ühestamisel HMM-märgendajaga moodustati treeningtekstide alusel kõigepealt sõnaliikide esinemise tõenäosuste tabelid, mida siis seejärel inimese poolt parandati. Nii sai eemaldada lihtsamad eesti keele eripäral tekkinud vead. Väga oluline oli ka märgenditesüsteemi valik. Töös kasutati 88 märgendit, mis olid valitud järgmiselt. Eristatati omadussõnu, põhiarvsõnu, järgarvsõnu, nimisõnu, pärisnimesid, isikulisi asesõnu, muid asesõnu, lühendeid, verbe, alistavaid ja rinnastavaid sidesõnu, hüüdsõnu, ees- ja tagasõnu, määrsõnu, punktuatsioonisümboleid ja tundmatuid sõnu.
Käändsõnade puhul eristati 5 käänet: nimetavat, omastavat , osastavat , lühikest sisseütlevat e. aditiivi ja “kõiki muid”. Isikuliste asesõnade puhul eristati lisaks ka kolme isikut. Ei eristatud ainsust ja mitmust. Verbide puhul eristatati kokku 13 märgendit: “ei”, “ära”, esimene pööre, teine pööre, kolmas pööre, kaudne kõneviis, “pole” ja “polnud”, da-infinitiiv, 0-lõpuline vorm, tingiva kõneviisi vormid, käskiva

1 Kasutatakse ka mõistet korrektsus

kõneviisi vormid, ma-infinitiivi vormid, partitsiibid. Ei eristatud ainsust ja mitmust ega aega. Ühestaja sai tabelite koostamiseks sisendiks morfoloogiliselt analüüsitud, kuid ühestamata teksti, morfoloogiliselt analüüsitud ja ühestatud teksti, lisaks veel ühestamisel kasutatavate märgendite loendi ja teisendustabeli morfoloogilistelt märgenditelt ühestaja märgendidele (see tabel seetõttu et morfoloogilised märgendid ei olnud enamasti samad, mis ühestaja märgendid). Treenimiseks kasutati G. Orwelli romaani „1984“ eestikeelset versiooni, milles on 75 000 sõna, ja testimiseks 2000 sõnalist osa Vello Lattiku raamatust “Mihklipäeval.Mihklikuul” . Erinevusi käsitsi ühestatud tekstiga oli umbes 7,1% (Kaalep,Vaino, 1998).

2. Tõenäosuslik sõnaliikide märgendamine otsustuspuu abil

2.1 Sissejuhatus

Tuntuima statistilise morfoloogilise ühestamise meetodi Markovi peitmudeli (Manning ja Schütze, 1999) alusel loodud ühestajate põhiprobleemiks on, et neil on raskusi väikese ja hajusa treeninghulga alusel õigete hinnangute tegemisel. Selle probleemi lahenduseks on välja pakutud otsutuspuude meetod. Otsustuspuu määrab automaatselt otsustamiseks vajaliku konteksti suuruse. Kontekstiks ei ole enam mitte ainult trigrammid ja bigrammid, vaid ka kontekstid kujul nt tag_{-1} = ADJ ja $tag_{-2} \neq$ ADJ ja $tag_{-2} \neq$ DET (Schmid, 1994) s.t. sõnale eelnev sõna on adjektiiv ja üleeelmine sõna ei ole adjektiiv ja ka mitte artikkel.

Otsustuspuu õppimise meetod baseerub eeldusel, et näidetevahelisi sarnasusi saab kasutada otsustuspuude automaatseks eraldamiseks (Daelemans, 1999: 297).

TreeTaggeril on palju ühist n-gramm ühestajatega: nad mõlemad kasutavad hindamisel märgendatud sõnade järjendit.

$$p(w_1w_2...w_n, t_1t_2...t_n) := p(t_n|t_{n-2}t_{n-1})\, p(w_n|t_n)\, p(w_1w_2...w_{n-1}, t_1t_2...t_{n-1}) \qquad (2.1)$$

Meetodid erinevad vaid siirdetõenäosuste (märgendite vaheliste tõenäosuste) hindamise poolest. N-gramm mudelid kasutavad sageli hindamisel MLE (Maximum likehood estimation) printsiipi - maksimaalse tõepära hinnangut:

$$p(t_n|t_{n-2}t_{n-1}) = F(t_{n-2}t_{n-1}\, t_n) / F(t_{n-2}t_{n-1}) \qquad (2.2)$$

kus $F(t_{n-2}t_{n-1}\ t_n)$ on trigrammi $t_{n-2}t_{n-1}\ t_n$ esinemiste arv korpuses ja $F(t_{n-2}t_{n-1})$ on bigrammi $t_{n-2}t_{n-1}$ esinemiste arv (Schmid, 1994).

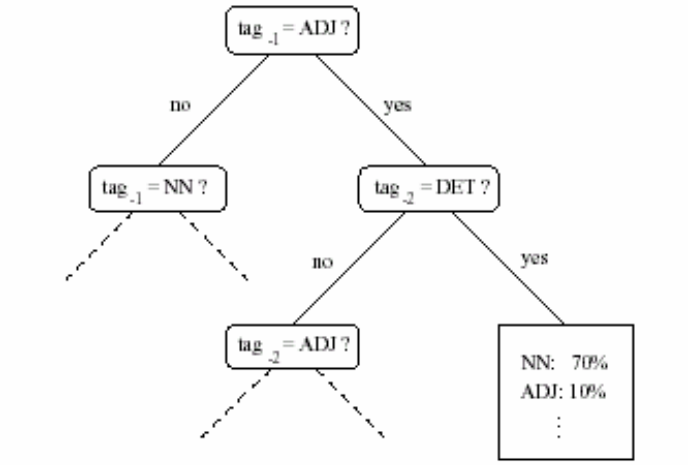
Selline meetod aga toob kaasa probleeme, sest mõnede harva esinevate trigrammide puhul võib olla raske mõista, kas tegemist on sõnade väga haruldase

koosesinemisega või on esinemine lihtsalt grammatiliselt vale. Esimesel juhul oleks tõenäosus võrdne ühega, viimasel juhul nulliga.

2.2 TreeTagger

Vastupidisel kirjeldatud n-gramm-märgendajale kasutab TreeTagger hindamiseks binaarset otsustuspuud. Trigrammi esinemise tõenäosus leitakse mööda otsustuspuud allapoole lehtedeni liikudes (Schmid, 1994) Otsustuspuu näide on toodud joonisel 2.1.

Vaatame näitena olukorda, kus tahame teada, mis sõnaliigi esindaja on sõna, mille ees on artikkel ja omadussõna. Vaatame, kas sellele sõnale eelnev sõna on omadussõna (tag_{-1}=ADJ?), liigume mööda jah-vastuse kaart, kas omadussõnale eelnev sõna on artikkel, jah - jõuame leheni, kus selgub et 70% tõenäosusega on tegemist nimisõnaga.



Joonis 2.1 Otsustuspuu näide

2.2.1 Otsustuspuu moodustamine

Otsustuspuu on andmestruktuur, milles sõlmed esindavad teste ja kaared nende vahel võimalikke vastuseid. Lahendus leitakse mööda otsustuspuud allapoole liikudes ja leheni jõudes. Tee, mida mööda läbi otsustuspuu liikuda sõltub vastustest, mida sõlmedes testile antakse (Daelemans, 1999:297). Otsustuspuu moodustatakse

rekursiivselt treeninghulga trigrammide alusel kasutades parandatud versiooni ID3-algoritmist (Schmid, 1994). Korduvalt jagatakse näidete hulk alamhulkadeks vastavalt sellele, kas vastava alamhulga näidetel on ühiseid väärtuste tunnuseid, kuni kõigil alamhulga näidetel on sama kategooria. Igal rekursiooni sammul luuakse test, mille alusel jaotatakse trigrammide näited kaheks alamhulgaks. Kontrollitakse, kas üks kahest eelnevast märgendist on võrdne märgendiga t.

$tag_{-i} = t;\ i \in \{1, 2\};\qquad t \in T,$ kus T on märgendite hulk.

Igal rekursiooni sammul võrreldakse kõiki teste ja nendest kõige rohkem informatsiooni sisaldav liidetakse vastava sõlme külge. Siis laiendatakse sõlme rekursiivselt igal testi q poolt defineeritud alamhulgal saades tulemuseks jah -ja ei-alampuud. Võrdlemise aluseks on kolmanda märgendi kohta kogutud informatsiooni hulk. Informatsiooni juurdekasvu kasumi maksimiseerimine on samaväärne keskmise informatsioonihulga I_q minimiseerimisega (Schmid, 1994).

2.2.2 Lihtsustatud algoritm

Otsustuspuude genereerimist selgitab järgmine lihtsustatud algoritm (Daelemans, 1999: lk 298)
Olgu antud näidete hulk T
Kui T sisaldab üht või enamat näidet, mis kuuluvad kõik samasse klassi C_j ,
siis on T otsustuspuu leht kategooriaga C_j.

Kui T sisaldab erinevaid klasse, siis

- Vali tunnus ja jaota T alamhulkadesse, mis sisaldavad valitud tunnuse väärtusi. Otsustuspuu sisaldab juhtumi nime sisaldavat sõlme ja alamhulgale viitavat haru.
- Rakenda protseduuri selliselt moodustatud alamhulkadele.

Algoritmi põhiraskus on õige tunnuse valimises. Kogu andmehulga jaoks puude koostamine on NP-täielik ülesanne, seepärast on vajalik heuristiline tunnuse valik.

2.2.3 Otsustuspuu kärpimine

Peale esialgse otsustuspuu moodustamist puu kärbitakse. Kui mõlemad alamsõlmed on lehed ja kaalutud informatsiooni kasum (weighted information gain) sõlmel on alla mingit teatud lävendit, siis eemaldatakse alamsõlmed ja sõlm saab ise leheks. Kaalutud informatsiooni kasum G on defineeritud kui :

$$G=f(C)(I_0-I_q) \tag{2.3}$$

$$I_0= \sum p(t|C)\log_2 p(t|C) \tag{2.4}$$

Kus I_0 on informatsiooni hulk, mis on vajalik vastaval sõlmel ühestamiseks ja I_q on informatsiooni hulk, mis on ikkagi vajalik peale testi q tulemuste teada saamist. On oluline, et informatsiooni kasumi kriteeriumit ei kasutataks puu loomise faasis, vaid alles peale seda. Nagu teisedki tõenäosuslikud ühestajad nii ka TreeTagger selgitab parima märgendite järjendi välja Viterbi algoritmi abil (Schmid, 1994).

2.2.4 Leksikon

Leksikon sisaldab iga sõna erinevaid märgendivõimalusi. Sellel on kolm osa: täisleksikon, sufiksi leksikon ja vaikeväärtus.

Kõigepealt otsitakse sõna TreeTaggeri täisleksikonist, kui sõna leitakse, siis tagastatakse vastav tõenäosuse vektor. Vastasel juhul muudetakse suurtähed väiketähtedeks ja otsitakse uuesti täisleksikonist. Kui ükski eelnevatest tegevustest ei ole olnud edukas, tagastatakse vaikeväärtus.

Kui märgendaja töötleb tundmatut teksti, on vägagi tõenäoline, et tegemist tuleb suure hulga tundmatute sõnadega, isegi kui leksikon on suur. Seega vajab märgendaja strateegiat tundmatute sõnade töötlemiseks. Kõige lihtsam võimalus on siduda iga tundmatu sõna iga sõnaliigi märgendiga võrdse tõenäosusega. Kuid teatud sõnaliigi märgendid (näiteks artiklid, kaassõnad ja asesõnad) võib sellest nimekirjast välja jätta, sest kõik need sõnad on suure tõenäosusega leksikonis esindatud (Schmid, 1995).

Sufiksileksikon on puu kujul, iga puu sõlm (v.a juur) on varustatud tähega. Leht-tippudes on märgendi täenäosuste vektorid. Sufiksipuu läbitakse alustades juurest, igal sammul liigutakse mööda haru, millel on sõna järgmine täht.

Sufiksileksikon moodustatakse automaatselt treeningkorpuse baasil. Sufiksipuu moodustatakse kõigi nimisõna, verbi ja omadussõnana märgendatud vähemalt viietähelise pikkusega sõnade baasil. Lisaks loeti kõigi sufiksite märgendite tõenäosused ja säilitatakse need vastava puu sõlmedes. Siis arvutatakse iga puu sõlmele informatsiooni mõõde (ingl k. information measure) I(S):

$$I(S) = - \sum P(pos| S)\log_2 P(pos| S) \qquad (2.5)$$

S on sufiks, mis vastab vastavale sõlmele ja P(pos| S) on sõna pos tõenäosus, mis vastab sufiksile S. Informatsiooni mõõdet kasutatakse sufiksipuu kärpimiseks. Iga lehe jaoks arvutatakse kaalutud informatsiooni kasum (ingl k. weighted information gain) G(aS) :

$$G(aS)= F(aS)\ (I(S)- I(aS)) \qquad (2.6)$$

Kus S on vanema sõlme sufiks, aS on vastava sõlme sufiks ja F(aS) on sufiksi aS sagedus.

Kui sufiksipuu mõne lehe informatsiooni kasum on alla antud väärtuse, leht eemaldatakse. Kõigi kustutatud lehtede vanemate märgendite sagedused kogutakse vanema vaikesõlme, kui vaikesõlm osutub ainsaks järelolevaks sõlmeks, kustutatakse ka see. Sellisel juhul saab vanem sõlm leheks ja asutakse kontrollima, kas ka seda saaks kustutada.

Kui vaikesõlme ei ole, loetakse otsing leksikonis ebaõnnestunuks ja tagastatakse vaikeväärtus. Vaikeväärtus koostatakse eraldades kõigi kärbitud sufiksipuu lehtede märgendite tõenäosusi juursõlme märgendite sagedustest ja siis normaliseerides tulemuste sagedusi (Schmid, 1994).

3. Ühestamine ja eesti keel

3.1 Hetkeolukord

Hetkel on eestikeelsete morfoloogiliselt ühestatud tekstide olukord on järgmine: on olemas kahe inimese poolt teineteisest sõltumatult ühestatud korpus ca 500 000 sõnaga. (http://www.cl.ut.ee/korpused/morfkorpus/)
Tekstid kuuluvad järgmistesse klassidesse (sõnade hulka ei ole arvestatud kirjavahemärke):

Liik	**sõnade arv**
Ilukirjandus (eesti autorid)	104 000
G. Orwelli "1984"	75 500
Ajakirjandus	111 000
Seadused	121 000
Horisont	98 000
Info-tekstid	4 000
Kokku	513 000

Joonis 3.1 Morfoloogiliselt ühestatud korpuse tekstide jaotus

Ilukirjanduse tekstid on pärit eesti autorite töödest. Ajakirjanduse tekstid on ajalehtedest „Postimees“, „Sõnumileht“, „Eesti Päevaleht“, „Äripäev“ ja „Maaleht“ ning kuuluvad ajavahemikku 1995-1999. Tõlkekirjandusest on esindatud G.Orwelli ulmeromaan „1984“.

```
<s>
Oli    ole+i //_V_ main indic impf ps3 sg ps af //
k&uuml;lm    k&uuml;lm+0 //_A_ pos sg nom //
selge    selge+0 //_A_ pos sg nom //
aprillip&auml;ev    aprilli_p&auml;ev+0 //_S_ com sg nom //
```

```
,    , //_Z_ Com //
kellad    kell+d //_S_ com pl nom //
l&otilde;id    l&ouml;&ouml;+id //_V_ main indic impf ps3 pl ps af //
parajasti    parajasti+0 //_D_ //
kolmteist    kolm_teist+0 //_N_ card sg nom l //
.    . //_Z_ Fst //
</s>
```

Joonis 3.2 Näide eesti keele morfoloogiliselt ühestatud korpusest

Märgendid <s> ja </s> tähistavad vastavalt kas lause algust või lõppu.

Read failis on kujul:

```
sõna     tüvi+lõpp // analüüs //
```

- <sõna> on sõna sellisena, nagu ta algselt esines
- <tüvi> on lemma e. algvormi tüvi: käändsõnadel ainsuse nimetav (kui seda ei ole olemas, siis mitmuse nimetav), pöördsõnadel ma-infinitiivi tüvi ilma (ma-lõputa)
- <lõpp> on sõna lõpp, kusjuures mitmuse tunnus on temaga liitunud (nagu seda on käsitletud ka Ülle Viksi "Väikeses vormisõnastikus"); partikkel GI/KI, kui ta esineb, on lihtsalt lõppu "kleepunud"; ka juhul, kui sõnal ei saagi lõppu olla (nt. hüüdsõnal), pannakse sõnale lõpp - nn. null-lõpp
- <analüüs> on üks variantidest, mis on kõik esitatud morfoloogiliste kategooriate tabelis.

Kui on tegemist liitsõna või tuletisega, siis:

- Tüvi on eristatud eelnevast komponendist '_' märgiga;
- Lõpp on eristatud eelnevast komponendist '+' märgiga; nn. null-lõpp ongi '+0'
- Sufiks on eristatud eelnevast komponendist '=' märgiga. Sufiksite märkimine ei ole järjekindel: märgitakse ainult teatud hulka produktiivseid sufikseid.
- Lemmatüvi leitakse ainult viimase parempoolse komponendi alusel

Mitmesõnalised nimed on kujul:

New Yorgis New York+s //_S_ prop sg in // (EKK)

```
Oli
    ole+i //_V_ s, //
k&uuml;lm
    k&uuml;lm+0 //_S_ sg n, //
selge
    selge+0 //_A_ sg g, sg n, //
aprillip&auml;ev
    aprillip&auml;e=v+0 //_A_ sg n, //
    aprillip&auml;e=v+0 //_S_ sg n, //
,
    , //_Z_ //
kellad
    kell+d //_S_ pl n, //
    kella+d //_V_ d, //
l&otilde;id
    l&otilde;i+d //_S_ pl n, //
    l&otilde;+d //_S_ pl n, //
parajasti
    parajasti+0 //_D_ //
kolmteist
    kolm_teist+0 //_N_ sg n, //
.
    . //_Z_ //
```

Joonis 3.3 Näide samast lausest ühestamata kujul, morfoloogia-analüsaatori väljund

Morfoloogia-analüsaator eristab 17 erinevat sõnaliigi märgendit.
Kasutatakse järgmisi sõnaliigi märgendeid:

- _A_ omadussõna - algvõrre (adjektiiv - positiiv), nii käänduvad kui käändumatud, nt *kallis* või *eht*
- _C_ omadussõna - keskvõrre (adjektiiv - komparatiiv), nt *laiem*
- _D_ määrsõna (adverb), nt *kõrvuti*
- _G_ genitiivatribuut (käändumatu omadussõna), nt *balti*
- _H_ pärisnimi, nt *Edgar*

- _I_ hüüdsõna (interjektsioon), nt *tere*
- _J_ sidesõna (konjunktsioon), nt *ja*
- _K_ kaassõna (pre/postpositsioon), nt *kaudu*
- _N_ põhiarvsõna (kardinaalnumeraal), nt *kaks*
- _O_ järgarvsõna (ordinaalnumeraal), nt *teine*
- _P_ asesõna (pronoomen), nt *see*
- _S_ nimisõna (substantiiv), nt *asi*
- _U_ omadussõna - ülivõrre (adjektiiv - superlatiiv), nt *pikim*
- _V_ tegusõna (verb), nt *lugema*
- _X_ verbi juurde kuuluv sõna, millel eraldi sõnaliigi tähistus puudub, nt *plehku*
- _Y_ lühend, nt *USA*
- _Z_ lausemärk, nt -, /, ...
- _T_tundmatu sõna

(EKK) Teised morfoloogilised märgendid on kirjeldatud põhjalikult Tiina Puolakaineni töös (2001).

4. TreeTagger

Programm TreeTagger[2] on välja töötatud ja programmeeritud Stuttgardi ülikoolis ning tema kasutamine on vaba akadeemilistel eesmärkidel. Seda programmi on edukalt kasutatud inglise, saksa, prantsuse, itaalia, bulgaaria, hispaania, kreeka, portugali ja vana prantsuse keele morfoloogiliseks ühestamiseks.

Programm TreeTagger kasutab töötamiseks otsustuspuude meetodit. Programm ise koosneb kahest osast: treeningosast ja test-osast. Treeningosa moodustab ette antud ühestatud tekstifaili alusel keelele vastava parameeterfaili, mille alusel test-osas omakorda ühestamata teksti on võimalik märgendada.

4.1 Treenimine

Train-tree-tagger on treenimisprogramm, mis vajab käsurea argumenti järgmisel kujul:
train-tree-tagger <lexicon> <open class file> <infile> <outfile>
{-cl <context length>} {-dtg <min. decision tree gain>}
{-ecw <eq. class weight>} {-atg <affix tree gain>} {-st <sent. tag>}
<lexicon>- täielikku leksikoni sisaldava faili nimi, faili igal real on sõnavorm ja märgend-lemma paaride järjend, tegelikult pole lemma informatsiooni vaja ja selle võib asendada nt. „-“ (TreeTagger readme)

```
aasta S aasta
aastasajal  S aasta_sada
aastasaja   S aasta_sada
aastase     A aastane
aastaseks   A aastane
aastaselt   D aastaselt A aastane
aastases    A aastane
aastasest   A aastane
aastas      S aasta
aastast     A aastane   S aasta
```

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

```
aastaste    A aastane
aastatagusel      A aasta_tagune    S aasta_tagune
aastatega   S aasta
aastateks   S aasta
aastatellimuse    S aasta_tellimus
aastatel    S aasta
aastateni   S aasta
aastatepikkust    A aasta     S aasta
aastate     S aasta
aastates    S aasta
aastati     D aastati
aastat      S aasta
```

Joonis 4.1 Näide leksikonist

<open class file>- fail mis sisaldab võimalike tundmatute sõnade märgendite nimekirja. See fail tavaliselt sisaldab adverbide, adjektiividee, nimisõnade, pärisnimede ja ka verbide märgendeid, aga mitte prepositsioone, artikleid, asesõnu või numbreid.
<infile> on sisendfail, igal real on üks sõna, millele järgneb õige sõnaliik (TreeTagger readme)

```
Kuid  J
kas   D
kunagi      D
on    V
olnud V
sellist     P
aega  S
,     Z
mil   P
oldi  V
ka    D
ühel  P
tasemel     S
?     SENT
```

Joonis 4.2 Näide train-tree-taggeri sisendfailist

Sõnaliikide märgendid on samad, mis varemgi, lauselõpumärki tähistab märgend SENT.
<outfile> väljundfailiks ehk train-tree-taggeri töö tulemuseks parameeterfail, mida kasutatatakse ühestamisprogrammi-TreeTaggeri töös
Erinevate lippudega on võimalik määrata konteksti pikkust (-cl <context length>), minimaalset otsustuspuu kasumit (-dtg <min. decision tree gain>), ekvivalentsusklassi kaalu (-ecw <eq. class weight>), afiksipuu kasutegurit (-atg <affix tree gain>) ja milline on lauselõpumärgend (-st <sent. tag>).(TreeTagger readme)

4.2 Märgendamine

Märgendamine toimub tree-tagger programmi abil, mille esimeseks argumendiks on train-tree-tagger-i väljundfail ehk parameeterfail ja teiseks argumendiks sisendfail ning kolmandaks argumendiks on väljundfaili nimi. Sisendfailiks on tekst, mille iga sõna on eraldi real.

```
Mina
olen
tubli
tüdruk
.
```

Joonis 4.3 Näide tree-taggeri sisendfailist

Tree-taggeri standardväljundiks, ehk lõpptulemuseks on sõnaliik iga sõna jaoks eraldi real.

```
P
V
S
A
SENT
```

Joonis 4.4 Näide treetaggeri standardväljundist

Lisaks on mitmeid erinevaid lippe väljundfaili täiustamiseks:

-token: prindib ka analüüsitava sõna

-lemma: prindib ka sõna algvormi, kui analüüsitavat sõna leksikonis pole asendatakse lemma <unknown> märgendiga.

-sgml: ei märgendata SGML kommentaare, st. ridu, mis algavad '<' ja lõpevad '>'-ga

-threshold <p>: printida kõik märgendid, mille tõenäosus on kõrgem kui <p> korda kõige tõenäolisema märgendi väärtus

-prob: prindib märgendite tõenäosused (nõuab lippu -threshold)

-no-unknown: printida <unknown> asemel tundmatute sõnade puhul analüüsitav sõna

-no-heuristics: mitte kasutada leksikoni heuristikaid

-quiet: mitte printida seisundi teateid

-pt-with-lemma: iga märgendi järel peaks olema tühik ja lemma

-pt-with-prob: iga märgendi järel peaks olema tühik ja märgendi tõenäosuse väärtus

-eos-tag <tag>: SGML märgend <tag> tähistab lause lõppu, eeldab lippu -sgml

Lisaks on veel mitmeid nn. eksootilisi lippe, näiteks

-proto: prindib iga sõna kohta leksilise informatsiooni

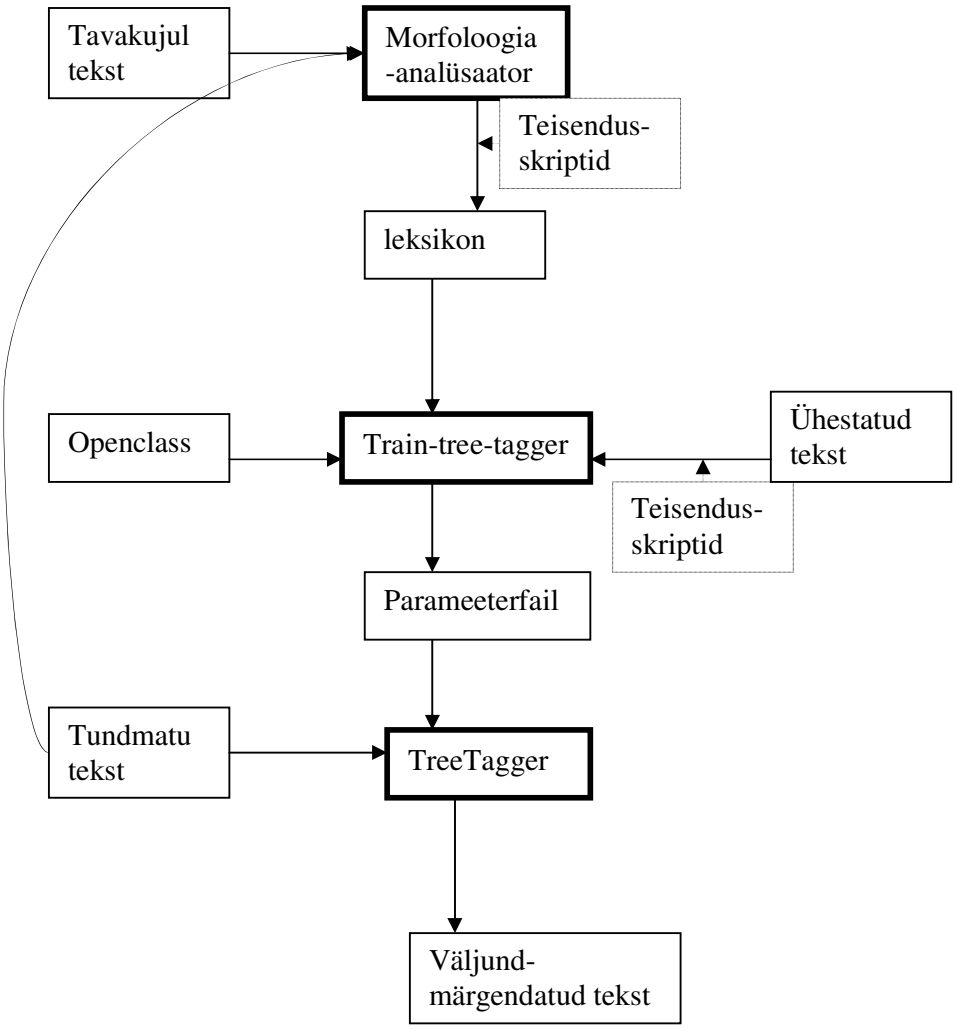
f: kui sõna leiti leksikonist

c: kui sõna väiketähtedega leiti leksikonist

h: sõna sisaldab sidekriipsu ja sidekriipsule järgnev sõna leiti leksikonist, nt öö-elu ei leitud, aga leiti elu (TreeTagger readme)

```
Kuidas D       kuidas  f D 1.000
sa      P       sina    f P 1.000
seda    P       see     f P 1.000
isa-olemist     S       <unknown>     h S 1.000 S 0.000
siis    D       siis    f D 0.764 J 0.236
ette    D       ette    f D 0.752 K 0.248
kujutadV        <unknown>     s V 0.999
?       SENT    ?       f SENT 1.000
```

Joonis 4.5 Näide tree-taggeri väljundist, kasutades lippe –token –lemma –proto



Joonis 4.5 Tree-Taggeri sisendid ja väljundid

5. Programmi rakendamine

5.1 Olukord

Minu uurimuses on kasutatud ainult ajakirjandus-, ilukirjandustekstide ja G. Orwelli romaani „1984“ põhjal koostatud korpust, teised tekstid on liiga spetsiifilised ja võiksid tulemust seega liialt mõjutada. Leksikonis ja sisendtekstis on umbes 300 000 rida ja testimiseks-kontrollimiseks umbes 60 000 rida.

5.2 Töötlus

Algselt tuli käsitsi ühestatud failidest välja valida, millised failid jäävad leksikoni ja treeningfaili koostamiseks ja millised jätta testimiseks - ühestaja töö õigsuse kontrolliks.

Samuti tuli kindlaks määrata esialgne märgendite süsteem. Olemasoleva morfoloogiliselt ühestatud korpuse sõnade morfoloogiline informatsioon koosneb erinevate märgendite kombinatsioonist, nimisõnadel näiteks sõnaliik, arv, kääne, verbidel aga sõnaliik, kõneviis, aeg, isik, arv, kõne, valentsimärgendid. Kui need märgendite kombinatsioonid asendada ühe erineva märgendiga, saaksime üle 800 märgendi. Ükski statistiline ühestaja ei suuda sellise märgendite hulgaga toime tulla. Praktikas kasutatakse teiste keelte puhul 30-150 märgendit, äärmisel juhul kuni 200 märgendit. TreeTaggeri sobivuse hindamiseks eesti keelele otsustasime esialgu piirduda ainult sõnaliigimärgenditega.

TreeTagger ei võimalda kasutada morfoloogiaanalüsaatorit otseselt, vaid vajab leksikonfaili, mis sisaldab sõnavormi ja selle vormi kõiki märgendeid. Selle moodustamiseks tuli ühestatud failid viia tagasi kujule, kus tekst poleks enam üks sõna ühel real, samuti tuli eemaldada morfoloogiline info. Selline puhastatud tekst anti morfoloogilisele analüsaatorile analüüsimiseks. Morfoloogiliseks analüüsiks kasutasin programmi ESTMORF (Kaalep 1999). Morfoloogia-analüsaatori väljundfail tuli omakorda töödelda leksikonfailiks. Selleks tuli morfoloogiliselt analüüsitud

ühestamata tekst teisendada kujule, kus sõna oleks ühel real koos kõigi erinevate tõlgendustega. Kuna morfoloogliselt analüüsitud tekst sisaldas meie jaoks liialt palju morfoloogilist infot - lisaks sõnaliigile ka käärde-või pöördeinfot, tuli see eemaldada. Tekstide töötlemiseks kasutasin Perli skripte.

```
kõrge A kõrge
kõrgeauline A kõrge_auline
kõrged      A kõrge
kõrgeid     A kõrge
kõrgeimaks  A kõrge=im
kõrgeimal   A kõrge=im
kõrgeima    A kõrge=im
kõrgeks     A kõrge
kõrgel      D kõrgel    A kõrge
kõrgele     D kõrgele   A kõrge
kõrgelt     D kõrgelt   A kõrge
kõrgema     A kõrgem    S kõrgem
kõrgemad    A kõrgem    S kõrgem
kõrgemaid   A kõrgem    S kõrgem
kõrgemaks   A kõrgem    S kõrgem
kõrgemal    D kõrgemal  A kõrgem    S kõrgem
kõrgemale   D kõrgemale A kõrgem    S kõrgem
kõrgemalgi  D kõrgemal  A kõrgem    S kõrgem
kõrgemalt   D kõrgemalt A kõrgem    S kõrgem
kõrgemat    A kõrgem    S kõrgem
kõrgemate   A kõrgem    S kõrgem
kõrgematele A kõrgem    S kõrgem
kõrgem      A kõrgem    S kõrgem
```

Joonis 5.1 Katke leksikonfailist

Lõpuks tuli kogu tulemus sorteerida tähestikulisse järjekorda. Lisaks tuli eemaldada ühe sõna mitmekordsed esinemised, selleks tuli ka suure ja väikse tähega kirjed üheks kirjeks töödelda.

Openclass fail on lihtsalt fail, mis sisaldab nimekirja morfoloogilistest märgenditest, antud juhul määrsõna, verbi, omadus-ja nimisõna märgendeid. Nende hulgast valitakse märgend sõnale, mida leksikonis ei leidu.

```
D
A
S
V
```

Joonis 5.2 openclass faili sisu

Sisendfailiks ehk treeningkorpuseks on morfoloogiliselt ühestatud tekst, mis on töödeldud kujule, kus igal real on sõna ja temale järgneb sõnaliiki tähistav täht. Ühestatud failid tuli jälle omakorda töödelda tree-taggerile sobivale kujule. Eemaldada lause algus- ja lõpumärgendid (<s> ja </s>), eemaldada sõna tüvi ja liigne käände- ja pöördeinfo, panna lauselõpumärgendiks SENT ja muuta SGML- kujul olevad tähemärgid tavakujule.

```
<s>
V&otilde;itlus    v&otilde;itlus+0 //_S_ com sg nom //
k&otilde;nnib    k&otilde;ndi+b //_V_ main indic pres ps3 sg ps af //
sellest    see+st //_P_ sg el //
eestlasest    eestlane+st //_S_ com sg el //
m&ouml;&ouml;da    m&ouml;&ouml;da+0 //_D_ //
.    . //_Z_ Fst //
</s>
```

Joonis 5.3 Näide ühestatud töötlemata failist

```
Võitlus          S
kõnnib      V
sellest     P
eestlasest  S
mööda D
.     SENT
```

Joonis 5.4 Näide sama lause töödelduna

Treenimis-etapi tulemusel genereeris train-tree-tagger faili ehk keelemudeli, mida kasutatakse tree-taggeri töös morfoloogiliselt analüüsimata teksti ühestamisel. Tree-taggeri sisendfailiks on tekstifail, milles iga sõna ja kirjavahemärk on eraldi real. Väljundfailina saadakse tekst, millele on lisatud sõnaliigi märgend ja tema tüvi.

```
Mõnesse
oli
juba
asunud
uus
põlvkond
,
ta
armastus
ei
olnud
enam
nii
eluline
,
et
isade
loodu
värske
oleks
hoidnud
.
```

Joonis 5.5 Näide treeningfaili sisendist

Mõnesse	P	Mõnesse	P
oli	V	oli	V
juba	D	juba	D
asunud	V	asunud	V
uus	A	uus	A
põlvkond	S	põlvkond	S
,	Z	,	Z
ta	P	ta	P
armastus	S	armastus	S
ei	V	ei	V
olnud	V	olnud	V
enam	D	enam	D
nii	D	nii	D
eluline	A	eluline	A
,	Z	,	Z
et	J	et	J
isade	S	isade	S
loodu	S	loodu	S
värske	A	värske	A
oleks	V	oleks	V
hoidnud	V	hoidnud	V
.	SENT	.	SENT

Joonis 5.6 Näide käsitsi ühestatuna samast lausest ja võrdluseks sama lause tree-taggeri poolt ühestatuna

Kahe lause, mis on täiesti suvaliselt valitud, võrdlusel näeme, et ühestaja on oma tööga suurepäraselt hakkama saanud.

6. Katsed

6.1 Esimene katse

Kõige esimeseks arvessevõetavaks katseks kasutasin 11219 sõnaga leksikoni, mis oli moodustatud 10 morfoloogiliselt analüüsitud, ühestamata failist. Sisendfailiks oli 408 670 rida. Ühestamise tulemuslikkuse testmiseks kasutasin 5000 reaga infotekstide faile. Õigesti ühestatud ja tree-taggeri poolt ühestatud failidel esines erinevusi 811 kohas, 811/5208*100=16% vigu, 100-16=84% õigeid. Ilmselgelt oli see tingitud asjaolust, et enamus valesti märgendatud sõnu ei esinenud leksikonis ja testimiseks kasutatud tekst oli liiga spetsiifiline - entsüklopeediline ja aianduslik tekst. Juba 63 viga oli seotud lühendi cm - vale analüüsiga. Sai selgeks, et asja parandamiseks tuleb leksikoni märgatavalt suurendada. Ja ka testimistekste mitmekesistada - lisada nn tavalisi tekste.

6.2 Teine katse

Treenimise sisend- ja leksikonfailiks olid ajakirjandus-, ilukirjandusfailid ja „1984“, mida algselt töötlema kujul oli 361 025 rida. Järgnes tavapärane töötlus, mille tulemusena valmisid 41963 reaga leksikon- ja 304576 reaga sisendfail. Lisaks juba varasemal töötlusel asendatud sgml märgenitele ä, ü, ö, õ, said asendatud ka š, ž, &, “ ja ”. Pärast treenimist sai train-tree-taggeri väljund- ehk parameeterfailist tree-taggeri sisendfail. Ühestamiseks kasutasin infotekstide faile, mis alguses sisaldasid 6162 rida, ajakirjandusest oli 10445 rida ja ilukirjandust 56326 rida. Kokku seega 66781 rida. Pärast töötlemist jäi järgi kokku 57053 rida. Tulemust omakorda võrdlesin õigete tõlgendustega failiga. Läbi viisin mitmeid erinevaid katseid, pärast iga katset erinevusi vaadeldes avastasin vigu, mida etapp etapilt parandasin. Üheks tähtsamaks

mitte minu poolt põhjustatud veaks oli, et morfoloogia analüsaator eristab omadussõna liikidest ka ülivõrret- U ja keskvõrret C, mida aga inimese poolt ühestatud tekstides eristatud pole. See tõi kaasa probleemi, kus tree-taggeri poolt ühestatud failis esinesid U ja C, aga võrdlemiseks mõeldud käsitsi ühestatud failis neid polnud. Olukorra lahendamiseks asendasin U ja C ka train-tree-taggerile vajalikes failides A-ga, ehk siis väljendab A omadussõna üldiselt. Pärast seda sain katsel tulemuseks 4628 erinevust, 57053 reaga teksti analüüsimisel, mis moodustab 8%. Seega 92% tree-taggeri poolt määratud tõlgendustest on õiged. Päris palju erinevustest oli tingitud ka sellest, et käsitsi ühestatud tekstides on pärisnimed määratud nimisõnadeks (S), aga morfoloogia-analüsaatori poolt analüüsitud teksti alusel koostatud leksikonfail määrab need märgendiga H, samamoodi on järgarvsõnadega, morfoloogia analüsaator määrab need märgendiga O, aga käsitsi ühestatud tekstides on kõik määratud numbriks (N). Jällegi olukorra lahendamiseks asendasin siis ka leksikonis ja train-tree-taggeri sisendfailis H S-ga ja O N-ga. Lisaks asendasin ka testimisfailides seni asendamata SGML- kujul tähed ja märgid. Tulemuseks oli 3600 erinevust õigesti ühestatud failiga ehk 6,3%. Kraadi märgi ° , km^2 ja km^3 SGML kujult tavakujule asendamine ei toonud loodetud tulemusi, tree-tagger märgendab need endiselt valesti nimisõnaks S, kuigi õige oleks lühend Y. Üheks väga suureks silmatorkavaks veaks on, et enamikel juhtudel määrab tree-tagger valesti kohanimede märgendi G-ks ehk käändumatuks omadussõnaks, kuigi õige oleks nimisõna S. Seda viga on raske seletada, kuna treenimisprogrammi sisendfailis on pärisnimed G-ks määratud mõnes üksikus kohas ja ka leksikonis ei ole midagi, mis tingiks G määramise suurema tõenäosuse. Ilmselt ongi põhjuseks G harv esinemine treeningandmetes.

6.3 Kolmas katse

Kolmanda katse eesmärgiks sai uurida ja katsetada erinevaid train-tree-taggeri lippe. Teha kindlaks, kas oleks võimalik veel ühestaja tulemusi parandada.

1. a) Esimesena muutsin konteksti suuruse võrdseks ühega, vaikeväärtusena on see 2, mis vastab trigrammi kontekstile. Train-tree-taggeri juhendis oli seda soovitatud väikeste korpuste peal töötlemisel. Tulemuseks oli vigade arvu suurenemine 157 juhtumi võrra, ehk siis 3600lt 3757ni.

b) Seejärel sai suurendatud konteksti väärtust kolmeks, mis tõi tulemuseks 3562 viga, ehk 38 viga vähem kui trigramm konteksti puhul.

2. Edasi muutsin lippu –dtg ehk minimaalse otsustuspuu kasumi väärtust.

a) Väärtuse muutmine 0,8-ks (vaikeväärtus oli 0,7) ei muutnud vigade arvu ei rohkemaks ega vähemaks.

b) Vaikeväärtuse 0,1 võrra vähendamine tõi kaasa ühe vea juurde tekkimise.

Seega ei toonud selle lipu muutmine kaasa olulisi muutusi.

3. Kolmanda sammuna modifitseerisin lippu –ecw. ehk suurendasin/vähendasin vaikeväärtust 0.15 . Ekvivalentsusklassi kaal väljendab tõenäosuste hinnnangut.

a) Väärtused 0,14 ja 0,13 ei põhjustanud muutusi vigade arvus.

b) Väärtus 0,12 vähendas vigade aru 1 võrra, samamoodi ka 0,11 ja 0,10

c) Kuid väärtus 0.08 tõi jällegi kaks lisaviga.

d) Väärtus 0,16 tõi kaasa ühe võrra väiksema vigade arvu.

e) Väärtused 0,18 ja 0,20 vähendasid vigu kahe võrra.

f) Väärtus 0,22 tõi vigade arvuks 3602, ehk kaks lisaviga

4. Kõige viimasena muutsin lipu –atg ehk afiksipuu kasutegurit. Vaikeväärtus on 1,2.

a) Väärtus 1,0 tõi vigade arvuks 3580, tervelt 20 viga vähem!

b) 0,9 aga hoopiski 3570 viga

c) 0,8 aga tõstis vigade arvu jällegi 3575 peale

d) 1,3 tõi 3618 viga

5. Lõpuks kasutasin siis kõige enam kasu toonud lippude kombinatsioonide ühist rakendust, seega cl -3 -ecw 0.18 ja -atg 0.9. Tulemuseks oli 3525 viga, seega lõplikuks parimaks tulemuseks võib lugeda vigade arv 6,18%.

6. Katsetasin veel võimalust eemaldada testimisfailidest infotekstide failid, kuna need sisaldasid palju leksikonis mitteesinevaid sõnu ja olid üleüldse väga spetsiifilise sisuga. Tulemuseks sai 2996 erinevust, ridu 51846, 5,78%

6.3.1 Vead

Vigade analüüsimiseks vaatlesin põhjalikumalt 2402 reaga ilukirjandusfaili ilu_0022.kym. Kõige enam vigu - 13 oli nimisõna määramisel verbiks.

Vigade arvult järgnesid 12 veaga omadussõna määramine verbiks, siin tulid üsna selgelt välja sarnasused valesti määratud sõnade vahel, näiteks olid valesti määratud partitsiibid `kurnatud, praetud, vateeritud, ajavad, avatud, jäänud, kaitsnud`.

Võrdselt kümme viga oli põhjustatud nii nimisõna määramisest omadussõnaks, omadussõna määramisest nimisõnaks kui ka määrsõna määramisest kaassõnaks. Viimases joonistus samuti selgemalt välja sõnade üldpilt: `eest, ees, taga, tagant(2), peale, vastu, kallale, kõrvale`.

Verbi määramisest omadussõnaks tekkis seitse viga. Omadussõna ja verbi märgendamisel võis sarnaselt vastupidisele järjekorrale sarnaseid vigu märgata, näiteks `laotatud, õhkunud, lausutud, nuuditatud, kohanud`. Seitse viga tekkis ka nimisõna määramisel verbiks, siin päris selgelt vigade üldpilti välja ei kujunenud, vigu oli nii eitusega koos esinemisel – `(ei) pea, (ei) aeta`, kui ka mineviku vormides `mugis, muutus, viibis`.

Kuus viga tekkis verbi määramisel nimisõnaks ja kaassõna märgendamisel määrsõnaks, `alla, ette (2), enne, läbi (2), üle`.

Viis viga tekkis määrsõna määramisest nimisõnaks ja neli viga vastupidisel määramisel.

Kolm viga terve faili peale tekkis asesõna määramisel arvsõnaks ja nimisõna määramisel kaassõnaks.

Kaks viga omakorda oli kaassõna nimisõnaks, omadussõna määrsõnaks, määrsõna sidesõnaks, määrsõna arvsõnaks, sidesõna määrsõnaks (`siis(2)`) ja nimisõna käändumatuks omadussõnaks (`Tallinna(2)`) määramisel.

Üks viga oli põhjustatud verbi asesõnaks, asesõna omadussõnaks, asesõna hüüdsõnaks, lühend nimisõnaks ja määrsõna nimisõnaks, arvsõna verbiks, määrsõna omadussõnaks, omadussõna ja käändumatu omadussõna nimisõnaks määramisel.

6.4 Neljas katse

Viimase katsena sai läbi viidud eksperiment, kus leksikoni sai lisatud ka testimiseks kasutatava korpuse sõnad. See sai võimalikuks, sest antud uurimuses polnud eesmärgiks katsetada TreeTagegeri võimekust morfoloogia-analüsaatorina, st tundmatute sõnadele analüüsi määramisel, vaid eelkõige ühestajana. Selline muutus tõi tulemuseks vigade vähenemise 2442-ni, ehk vigu leidub 4,71% ja õiged on 95,29% määrangutest. Kasutasin esialgset infotekste sisaldavat korpust.

6.4.1 Vead

Ka pärast testkorpuse sõnade lisamist leksikoni uurisin lähemalt ühestamisel tekkinud vigu, nimelt sama faili, mis eelmine kord, s.t ilukirjandusfaili ilu_22.kym. Seekord oli loomulikult vigu tunduvalt vähem.

Kõige enam vigu (17) tekkis omadussõna määramisel verbiks (`kangestunud, sihitud, avanenud, trellitatud, külmunud, kurnatud, söödavat, praetud, vateeritud, lagunenud, lapitud, ajavad, kohendatud, avatud, jäänud (2), kaitsnud`)

Üksteist viga oli nimisõna määramisel verbiks.

Kümme viga tekkis määrsõna määramisel kaassõnaks, sarnaselt eelmise vea-analüüsiga - `eest, ees, algul, tagant (2), peale, vastu, taga, kallale, kõrvale`.

Üheksa viga tõi omadussõna määramine nimisõnaks - `poisivibalik, läbielatud, allakeeratud, ettesirutatud, ärakaranud, sooja, väljakannatamatu, tuttavana, mahajäetavaile`.

Kaheksa viga põhjustas kaassõna märgendamine määrdsõnaks- `ette (2), alla(2), enne, läbi (2), üle`.

Seitse viga põhjustas nimisõna määramine omadussõnaks, nt. `külm, hingelist, sügavusest, varast(2), hoone, põrandaalust`. Tegelikult oli

siin veel 15 viga, mis olid põhjustatud puhtalt nime Habemik erinevate variantide määramisega omadussõnaks.

Verbi määramisel nimisõnaks tekkis 6 viga. `((ei) pea, näinud, kohanud, nuuditatud, (ei) tunne, pidanud`)

Võrdselt kolm viga tekitasid nimisõna kaassõnaks, sidesõna määrsõnaks, verbi omadussõnaks ja asesõna arvsõnaks `(ühe, teisi, teist)` määramisel.

Kaks viga põhjustasid nimisõna käändumatuks omadussõnaks (Tallinna (2)), määrsõna verbiks (`ei` kaks korda lause alguses), nimisõna määrsõnaks ja arvsõnaks ning omadussõna määrsõnaks ja kaassõna nimisõnaks märgendamisel.

Üksikuid vigu põhjustas määrsõna hüüdsõnaks, kaassõna nimisõnaks, nimisõna määrsõnaks, arvsõna asesõnaks, määrsõna nimisõnaks, verbi ning nimisõna määramisel asesõnaks, määrsõna omadussõnaks, arvsõna verbiks ja määrsõna hüüdsõnaks määramisel.

6.6 Järeldused ja edasiarendused

TreeTaggeri rakendamisel teistele keeltele on saadud samuti häid tulemusi. Loomulikult on neid raske antud katse tulemustega võrrelda, kuna korpuste suurused ja kasutatud märgendite hulk erinevad suuresti.

Inglise keelele rakendades testiti TreeTaggerit Penn-Treebank korpusel, kasutades 2 miljonit sõna treenimiseks ja 100 000 sõna testimiseks, tulemuseks oli 96, 36% õigeid märgendeid. Penn-Treebank korpus sisaldab 36 erinevat märgendit. (Schmid, 1994)

Saksa keele puhul kasutati treenimiseks 20 000 sõna ja testimiseks 5000 sõna. Leksikon koosnes 350 000 sõnast ja parimaks tulemuseks saavutati 97, 53%. (Schmid, 1995)

Rootsi keele puhul katsetati TreeTaggerit kasutades treenimiseks 1, 1 miljoni sõnalist korpust ja testimiseks 60 000 sõnalist osa, parimaks tulemuseks saavutati 95, 1%. Kasutati 150 erinevat märgendit. (Sjöberh, 2003)

TreeTaggerit on kasutatud ka vana prantsuse keele märgendamisel, kus oli kasutada 2,6 miljonilist sõna treenimiseks ja testimiseks 415 000 sõnalist korpust, tulemus on ligi 95% ning kasutati 24 märgentit. (Stein, 2003)

Ilmselge on et praegusel kujul pole antud eksperimendist eesti keele ühestamise-alases arendustöös suurt kasu. Kasutatud on ainult primaareseid sõnaliigi märgendeid, ei eristata käände- ega pöördeinfot. Edasiseks süntaksitöötluseks oleks vaja ikkagi detailsemat morfoloogilist infot.

Siiski võib olla perspektiivne TreeTaggeri kasutamine koostöös reeglipõhise morfoloogilise ühestajaga (Puolakainen, 2001), milles allesjäänud mitmesustest 38.5% on just sõnaliikide vahelised.

Samuti tuleks katsetada võimalusi, et täiustada märgendite süsteemi. Olukordi, kus verbi eituse vormi ja nimisõna mitmesused põhjustasid hulgaliselt vigu, on võimalik vältida, kui märgendada verbi eitus eraldi märgendiga. Praktilistes rakendustes (nt terminite automaatsel tuvastamisel) oleks ilmselt vajalik eristada nimisõna käändeid, eelkõige just kolme esimest. Huvitav oleks korrata Kaalepi ja Vaino (1998) eksperimenti TreeTaggeriga ehk siis sama märgendite süsteemi kasutada teise algoritmiga.

Kokkuvõte

Tekstide automaatse analüüsi algusetapi kõige tähtsamaks osaks on morfoloogiline töötlemine. Töötlus koosneb kahest osast: sõnaliikide määramisest ja ühestamisest. Ühestamiseks on kasutusel mitmeid meetodeid: kasutatakse reeglipõhiseid, statistikal põhinevaid, närvivõrkudel baseeruvaid jt ühestajaid. Käesolevas töös antakse ülevaade katsest rakendada eesti keelele statistikal põhinevat otsustuspuude meetodit kasutavat ühestajat.

1994. aastal töötas H. Schmid Stuttgardi ülikoolis välja otsustuspuude meetodit kasutava ühestaja - TreeTaggeri. Eesti keele jaoks on olemas ligi 500 000 sõnast koosnev morfoloogiliselt ühestatud korpus, mis on vajalik statistiliste morfoloogiliste ühestajate treenimiseks.Töö TreeTaggeriga koosneb kahest etapist: inimese poolt ühestatud tekstikorpusel treenimisest ja uue, morfoloogiliselt mitmese teksti automaatsest ühestamisest.

Antud töös tegeleti morfoloogianalüsaatori ja treeningtekstide teisendamisega TreeTaggerile sobivale kujule, TreeTaggeri treenimisega, selle optimaalse konfiguratsiooni leidmisega ning saadud keelemudeli hindamisega testkorpusel.

Kasutusel oli 15 erinevat sõnaliigi märgendit, parimaks tulemuseks oli vigade protsent 4,71, testimishulgal, mis koosnes 51846 reast ja sisaldas ajakirjandus- ning ilukirjandustekste. Treenimishulgaks oli 304 576 sõnast koosnev korpus.

Part-of-Speech Tagging with TreeTagger

Bachelor thesis

Kadri Kajaste

Abstract

This bachelor thesis presents a new statistical method for Estonian part-of-speech tagging. The method uses decision tree algorithm and the program is called the TreeTagger.

Disambiguation is a very important part of morphological processing. More than half of the word forms are ambiguous after morphological analysis. In disambiguation phase the correct word form is chosen using the context information..

TreeTagger needs a lexicon and disambiguated text for construction of parameter file. A lexicon is a file that contains all kind of word forms with their different potential tags. Morphologically disambiguated corpus of Estonian texts consists of aproximately 500 000 words.

The work on the bachelor project was divided into three phases. First, all texts had to be processed to the TreeTagger demanded form. After that the training and configuration of TreeTagger took place using corpora of 304576 words. Only 15 part-of-speech tags were used in this experiment. For evaluation, separate corpus (51846 words) was used. The TreeTagger performs relatively well: the output contains 4,71% errors compared with manually annotated text. The detailed description of error types is given in the thesis.

Kirjandus

1. W.Daelemans, *Machine Learning Approaches. Rmts Syntactic Wordclass Tagging.* (ed H. van Halteren). Kluwer Academic Publishers. Dordrecht/Boston/London. lk 285-304, 1999.

2. M.El-Beze , M.B. Merialdo. *1999. Hidden Markov Models. Rmts Syntactic Wordclass Tagging.* (ed H. van Halteren). Kluwer Academic Publishers. Dordrecht/Boston/London. lk 263-284, 1999.

3. EKK = Eesti Kirjakeele Korpus
http://www.cl.ut.ee/korpused/morfkorpus/ (28.05.06)

4. H.J.Kaalep *ESTMORF, eesti keele morfoloogiline analüsaator ja süntesaator, (lõplik versioon), 1999*
http://www.eki.ee/keeletehnoloogia/projektid/estmorf/estmorf.html (28.05.06)

5. H.J. Kaalep, T. Vaino *Vale meetodiga õiged tulemused? Eesti keele morfoloogiline ühestamine statistika abil*, Keel ja Kirjandus,nr 1, lk 30-38, 1998.
http://www.cl.ut.ee/yllitised/kk_yhest_1998.pdf (28.05.06)

6. C. D. Manning, H.Schuetze *Foundations of Statistical Natural Language Processing*, Ch. 9. 1999.

7. K. Müürisep *Eesti keele arvutigrammatika: Süntaks,* Dissertationes Matematicae Universitatis Tartuensis, Tartu, 2000.
http://math.ut.ee/~kaili/thesis/ (28.05.06)

8. T.Puolakainen *Eesti keele arvutigrammatika: morfoloogiline ühestamine. Dissertationes Matematicae Universitatis Tartuensis, Tartu,* 2001.

9. H.Schmid *Probabilistic Part-of-Speech Tagging Using Decision Trees,* 1994.
http://www.ims.uni-stuttgart.de/ftp/pub/corpora/tree-tagger1.pdf (28.05.06)

9. H.Schmid *Improvements In Part-of-Speech Tagging With an Application To German*, 1995.
http://www.ims.uni-stuttgart.de/ftp/pub/corpora/tree-tagger2.pdf (28.05.06)

10. J.Sjöberh *Combining POS-taggers for improved accurancy on Swedish text,* NoDaLiDa, Reykjavik, 2003.
www.nada.kth.se/~jsh/publications/**combining**03.pdf (28.05.06)

11. A. Stein *Part of Speech Tagging and Lemmatisation of Old French Texts,* Stuttgart, 2003.
www.uni-stuttgart.de/lingrom/stein/forschung/altfranz/afrlemma.pdf (28.05.06)

12. A.Voutilainen, Hand-Crafted Rules. Rmts Syntactic *Wordclass Tagging. (ed H. van Halteren).* Kluwer Academic Publishers. Dordrecht/Boston/London. lk 217-246, 1999.