TARTU ÜLIKOOL
MATEMAATIKA-
INFORMAATIKATEADUSKOND
Arvutiteaduse instituut
Informaatika eriala

Indrek Klanberg

Eesti-inglise statistilise masintõlke mudeli ümberpööramine inglise-eesti suunale

Bakalaureusetöö (6 EAP)

Juhendajad: Mark Fishel,
Mare Koit

Autor: ..... " ..... " mai 2012
Juhendaja: ..... " ..... " mai 2012
Lubada kaitsmisele
Professor: ..... " ..... " mai 2012

Tartu 2012

Sisukord

1. Sissejuhatus

Käesolev töö on tehtud statistilise masintõlkimise vallas. Enne statistilist lähenemist üritati masintõlkimist lahendada arvutile erinevaid keelereegleid ette andes, mis nõuab palju tööd, sest tihti on keeltes väga palju erandeid. Statistilised meetodid põhinevad sellel, et võimalikult suure olemasoleva paralleelkorpuse peal treenitakse mudelid, mida kasutab funktsioon, mis hindab lähtekeele lause V vastavust tõlgitud lausesse E. Seega genereerides mitmeid erinevaid võimalikke tõlkelauseid, võime hinnata, milline neist on parim. Käesolev töö keskendub aga mudelite treenimisele ning ei süvene lausete genereerimisse.

Praktiline osa tööst käsitleb inglise-eesti masintõlkemudeleid. Praktilises osas treeniti eesti keelemudel, treeniti inglise-eesti fraasimudel ja ümberjärjestusmudel ning pöörati ümber olemasolev erinevatest korpustest kaalutult kokku pandud eesti-inglise fraasimudel ja ümberjärjestusmudel. eesti-inglise fraasimudel oli kaalutult kokku pandud kasutades tarkvara TMcombine (TMcom). Lõpuks võrreldi saaduid mudeleid omavahel ja analüüsiti tulemusi eesti keele käänete kasutuse seisukohast. Kogu töö tehti Moses (www2, Moses) raamistikku kasutades.

Käesoleva töö ülesehitus on järgnev: sissejuhatusele järgnevas 2. peatükis räägime algorütmidest, mida praktilises pooles Moses sisuliselt on kasutanud, 3. peatükis räägime praktilisest teostusest (Mosese kasutamine) ja viimasena 4. peatükis analüüsime antud tulemust.

2. Statistilise masintõlkimise teooriast

Statistilises masintõlkimises on alati olemas funktsioon f, mis hindab lause tõlget *E* saades lisaks ette lähtekeele lause V. Selleks, et f oma hinnangut saaks anda, kasutab ta ühte või mitut mudelit, mis on saadud kahe paralleelse korpuse peal treenides. Näiteks võib mudeliks olla sõnamudel, kus on mõlema keele sõnapaaridele antud tõenäosus $t(e|v)$, mis tähendab, et sõna v tõlgitakse sõnaks e tõenäosusega $t(v|e)$. Näide, kuidas seda on väljendatud reaalses failis:

greatest suurimale 0.0289855
biggest suurimale 0.0652174
highest suurimale 0.1231884

Käesolevas töös kasutasime Mosese raamistikku (www2). Täpsemalt kasutab funktsioon f fraasimudelit, keelemudelit ja ümberpaiknemismudelit (i. k. *reordering model*). Töö käigus said kõik mudelid ka treenitud.

Fraasimudeli eelis sõnamudeli ees on see, et palju lihtsam on käsitleda olukordi, kus mõned sõnad ühes keeles kaovad, või teised tekivad juurde. Nt. fraasi "the book" tõlkimine fraasiks "raamat", sõna "the" kaob ära. Sõnamudelile on välja mõeldud erinevaid võtteid nagu rikastamine, null sõna ja paljud teised, mis simuleeriksid sõnade juurde tekkimist ja kadumist, kuid fraasimudel sisaldab neid omadusi juba enda olemuses ja on sellevõrra palju intuitiivsem. Fraasimudel arvutis väljendub fraasitabeli kaudu, kus igas reas on fraas v tõlgitavas keeles, fraas e eesti keeles ja $t(e|v)$, mis väljendab tõlgitavuse tõenäosust.

Kui tõlkimisel genereerida lauseid, kus on samad fraasid, kuid nad on kokku pandud erinevas järjekorras, siis fraasimudel ei ole võimeline ütlema, kumb neist lausetest on parem tõlge. Sellepärast kasutatakse keelemudelit. Keelemudelit iseloomustab funktsioon k(E), mis väljendab lause E eksisteerimise tõenäosust eesti keeles (antud juhul). Selleks kõige lihtsam moodus on näiteks n-gramm meetod, meie kasutame töös trigrammi. Viimane tähendab, et on antud tõenäosused, et kolm sõna a, b ja c esinevad järjekorras $\overline{abc}$.

Näiteks, lause "I live in a house" jaotatakse 3-ks fraasiks: "I live", "in", "a house" ja vastavad fraasid eesti keeles võetakse "ma elan", "sees", "maja". Kuna fraasimudel ei nõua fraaside esinemist järjest vaid lihtsalt nende vastandavuse olemasolu, siis võib seda lauset tõlkida "Ma elan sees maja" või "Ma elan maja sees". Keelemudeli tabelis on aga märgitud, et "elan maja sees" on tõenäosusega 0.3, ja "elan sees maja" on tõenäosusega näiteks 0.001, seega oskab tõlkemudel teha järelduse, et "Ma elan maja sees" on parem tõlge.

Järgnevalt vaatame erinevaid treenimisalgoritme lähemalt. Täpsemalt, vaatame neid algoritme, mida me oleme kasutanud praktilises töös. Teave algoritmide toimimisest tuleb allikast "Statistical Machine Translation", autoriks Philipp Koehn. Teave selle kohta, mida praktilises töös kasutatakse, on saadud erinevatest veebijuhenditest Mosese ja tema osade kohta.

2.1 Treenimine, sõnamudel

Esimesena tutvume sõnamudeli treenimisega, mis on kõige lihtsam, ning mida tehakse paratamatult isegi enne fraasimudeli tegemist. Tegu on sissejuhatava algoritmiga, mida otseselt praktilises töös ei kasutata.

Sõnamudeli tegemise alguses on meil üks paralleelkorpus - kaks suurt lausete jada, kus ühes on laused V (võõrkeelsed) ja teises tõlkelt vastavuses laused E (eesti keeles). Me teame, et lause V_i vastab lausele E_i, kuid me ei tea midagi selle kohta, mis seos on sõnadel nendes lausetes. Esiteks võivad need kaks lauset olla erineva pikkusega, mis tähendab, et ühes keeles tekib sõnu juurde või kaob ära. Teiseks ei tea me midagi sõnade järjekorra muutumise kohta.

Siinkohal vastavus sõnade v ja e vahel lausetes V ja E tähendab, et sõna v tõlgitakse sõnaks e. Järgnev ülesanne seisnebki selles, et me leiame, mis tõenäosusega sõna v tõlgitakse sõnaks e (edaspidi tähistame $t(e|v)$).

Üks lihtne algoritm selle probleemi lahenduseks on EM (*Expectation Maximization*). Täpsemalt vaatame IBM Model 1 varianti sellest. Alguses me eeldame, et iga sõna v tõlkimine mistahes sõnaks *e*, on võrdse tõenäosusega ehk $\forall i, j, h \; kehtib \; t(e_j|v_i)=t(e_h|v_i)$.

Teiseks rakendame ajutist mudelit ($t(e|v)$ tabelit) korpusele, et hinnata vastavusi. Siinkohal tuleb mõista, et selleks, et kahes paralleelses lauses sõnade "house" ja "maja" esinemine suurendaks tõenäosust, et "house" tõlgitakse sõnaks "maja", peame olema veendunud, et nad on omavahel vastavuses. Reaalselt oleme veendunud *X* %, mispuhul, mida suurem on *X*, seda rohkem me panustame sellele, et "house" tõlgitakse sõnaks "maja". Vastavuse tõenäosusi aga omakorda järeldame sellest, mis on hetkeline mudel. Matemaatiliselt väljendub see nii (Koe10, lk 90):

$$p(a|e,f)=\prod_{j=1}^{l_e}\frac{t(e_j|v_{a(j)})}{\sum_{i=0}^{l_v} t(e_j|v_i)}$$

Järgnevalt loeme paralleelkorpuses üle, kui palju kordi me arvame (ehk võtame arvesse $X\,\%$), et sõna v tõlgitakse sõnaks e ja ka seda, kui palju sõna v üldse esineb. Kõige lõpuks saame välja arvutada uue mudeli ja korrata eelnevat kahte tegevust, kuni enam miski ei muutu, ehk oleme jõudnud lõpliku stabiilse lahenduseni.

Algoritmist 1 on näha, kuidas eelkirjeldatut rakendada.

Igas lausepaaris E, V arvutatakse iga e kohta $\sum_v t(e|v)$. Selle ja $t(e|v)$ põhjal saab arvutada kui suure reaalse tõenäosusega antud lauses on vastavuses sõnad e ja v hetkelise mudeli põhjal. Seda kasutatakse ära järgnevas kahes *for* tsüklis, kus suurendatakse globaalset $count(e|v)$ ja $total(v)$, mille põhjal, kui kõik lausepaarid on läbi käidud, saab arvutada uue mudeli.

$$t_{uus}(e|v) = \frac{count(e|v)}{total(v)} = \frac{\sum_{E,\, kus\, e \in E,\, V,\, kus\, v \in V} \frac{t_{vana}(e|v)}{total_{E,V}(e)}}{\sum_{E,\, kus\, e \in E,\, V,\, kus\, v \in V,\, e_1} \frac{t_{vana}(e_1|v)}{total_{E,V}(e_1)}}.$$

Pärast seda kordame kogu protsessi, kuni uus mudel ei erine enam vanast mudelist. Ehk oleme leidnud antud korpust ja algoritmi kasutades parima võimaliku mudeli.

Algväärtusta $t(e|v)$
while (*ei ole lõpp lahendus*) *do*

$count(e|v) = 0 \;\forall\, e, v$
$total(v) = 0 \;\forall\, v$
for all lausepaar (E, V) *do*
- *for all sõnad* $e \in E$ *do*
  - $total_{E,V}(e) = 0$
  - *for all sõnad* $v \in V$ *do*
    - $total_{E,V} = total_{E,V}(e) + t(e|v)$
- *for all sõnad* $e \in E$ *do*
  - *for all sõnad* $v \in V$ *do*
    - $count(e|v) = count(e|v) + \frac{t(e|v)}{total_{E,V}(e)}$
    - $total(v) = total(v) + \frac{t(e|v)}{total_{E,V}(e)}$

for all võõrkeelsed sõnad v do
- *for all eesti sõnad e do*
  - $t(e|v) = \frac{count(e|v)}{total(v)}$

Algoriim 1: IBM Model 1

2.2 Treenimine, fraasimudel

Tänapäeva parimad statistilised mudelid ei vaatle kõige väiksema osana sõnu, vaid hoopis fraase. Miks see parem on? Esiteks kaovad ära probleemid, kus ühes keeles n sõna vastavad teise keele m sõnale, kus $m \neq n$. Teiseks idioomid - kui tekstis esineb mõni idioom korduvalt, oleme võimeliselt seda õigesti tõlkima, vaadeldes seda fraasi tervikuna. Kui me vaatame aga iga sõna eraldi, siis suure tõenäosusega üritame tõlkida seda otse ja seega valesti.

Käesolevas bakalaureusetöös kasutamegi fraasitabelit kasutavat mudelit. Täpsemalt on meil kahesuunaline fraasimudel koos kahesuunaliste leksikograafiliste kaaludega. Sellest kõigest räägime kohe lähemalt.

Üldiselt tahame, et meil oleks fraasitabel, kus oleksid tähenduslikud fraasid ning nende vahel oleksid vastavuse tõenäosused. Kahesuunalise mudeli korral tahame, et tõenäosused oleksid mõlemat pidi (eesti- ja võõrkeelne fraas võivad ühtpidi saada palju suurema tõenäosuse kui teistpidi). Miks me tahame kahesuunalist tabelit ühte suunda tõlkides, sellest räägime hiljem lähemalt.

Alustame sellest, et kõigepealt on meil vaja kogu korpusest välja tuua kõik mõistlikud fraasipaarid. Mis asi on mõistlik? Kui meil on sõnade vaheline joondus $J(e, v)$ teada, siis kui fraas f_e , mille algus ja lõpp on vastavalt i_e ja j_e , vastab fraasile f_v , mille algus ja lõpp on vastavalt i_v ja j_v , siis kehtivad järgnevad kaks tingimust:

$$\forall (e, v) \in J : (i_e \leq e \leq j_e) \vee (i_v \leq v \leq j_v) \Rightarrow (i_e \leq e \leq j_e) \wedge (i_v \leq v \leq j_v)$$

$$\exists (e, v) \in J, kus (i_e \leq e \leq j_e) \wedge (i_v \leq v \leq j_v)$$

Lühidalt, kui mingi sõna on kas fraasis f_e või f_v , siis tal kas ei ole joondatavat sõna (ing. k. *aligned*) või talle joonduv sõna on vastavas fraasis. Lisaks peab kehtima veel tingimus, et fraas koosneb vähemalt ühest sõnast, mis joondub teise keele fraasi. Neid kahte tingimust kokku nimetatakse fraasi terviklikkuse tingimuseks (Koe10, lk 131). Seda tingimust kasutades kogume kokku kogu korpuse pealt fraasipaarid.

Kui oleme fraasid kätte saanud, siis nende kasutamiseks peame leidma nende vastavuse tõenäosused. Neid leitakse kasutades suurima tõepära printsiipi:

$$\Phi(\bar{e} | \bar{v}) = \frac{count(\bar{e}, \bar{v})}{\sum_{\bar{v}_i} count(\bar{e}, \bar{v}_i)}$$

Kuna kasutame mudelis tõenäosuseid mõlemal suunal, siis arvutatakse ka see teistpidi välja:

$$\Phi(\bar{v}|\bar{e}) = \frac{count(\bar{v}, \bar{e})}{\sum_{\bar{e}_i} count(\bar{v}, \bar{e}_i)}$$

Miks on kahesuunaline mudel parem ükskõik mis pidi ühesuunalisest? Ühesuunalisel, näiteks inglise keelest eesti keelde mudelit ehitades võib tekkida olukord, kus haruldane fraas $\bar{v}$ joondub väga laialt levinud fraasiga $\bar{e}$. Kuna harv fraas võib näiteks esineda ainult üks või kaks korda, siis ülaltoodud valemi põhjal saab ta väga suure tõenäosuse. Teistpidi aga saaks ta ikkagi väikese tõenäosuse, mistõttu mõlemat suunda kasutades on võimalik vältida selliseid eksitusi.

Siinkohal tuleb tähele panna, et antud lahendus ei rahulda meid juhul, kui mõlemas keeles haruldased fraasid tõlgitakse üksteiseks suure tõenäosusega. Meil ei ole küll otsest põhjust arvata, et tegu on vale tõlkega, aga mida vähem meil andmeid mõne teadmise toetamiseks on, seda vähem usaldusväärne on antud teadmine.

Eeltoodud probleemi lahendamiseks tuuakse juurde leksikograafiline kaal. Põhjus peitub selles, et kuigi fraas võib olla haruldane, siis fraasis sisalduvate sõnade kohta on meil suure tõenäosusega palju rohkem informatsiooni. Efekti saavutamiseks piisab juba eelpool väljatoodud IBM mudel 1 kasutamisest. Täpsemalt kasutame fraasipaarile teadaolevat joondust *J(e|v)* ning leksikograafilisi tõenäosuseid *t(e|v)*, et arvutada fraasi leksikograafiline kaal suunale inglise keelest eesti keelde.

$$lex(\bar{e}|\bar{v}, a) = \prod_{i=1}^{pikkus(\bar{e})} \frac{1}{count(j), kus(i, j) \in J} \sum_{\forall (i,j) \in J} t(e_i, v_j)$$

Praktilises töös kasutatud mudelil on leksikograafiline kaal mõlema suuna peal.

2.3 Treenimine, keelemudel

Senimaani oleme vaadelnud sõna- ja fraasimudeleid. Kui me alustaksime tõlkimisega, siis need mudelid ei suuda vastata küsimusele, kas antud lause sõnade järjekord on hea või halb. Nt. võtame lause "I had a good day" ning selle tõlke "Mul hea päev oli". Kui joondada "I"-"Mul", "a good day"-"hea päev" ja "had"-"oli", siis fraasimudel annab omalt poolt hea tõenäosuse, sest iga järjestus samade fraasidega on fraasimudeli jaoks võrdse tõenäosusega. Samas on ilmne, et tegemist ei ole hea eestikeelse lausega. Teine probleem, mis võib esineda, on see, et me võime "I had" tõlkida nii "Mul oli" kui ka "Ma omasin" ning alles konteksti pannes saame aru, et hetkel tahame me kindlasti tõlget "Mul oli".

Nende probleemide lahendamiseks on välja mõeldud keelemudel, mille eesmärk on hinnata, kui suure tõenäosusega antud lause kuulub vaadeldavasse keelde. Kuna antud töös tõlgime inglise keelest eesti keelde, siis meid huvitab just eesti keelemudel.

Üldiselt on keelemudeli ülesanne anda lausele *E* tõenäosus, mis näitaks, kui suure tõenäosusega ta esineb eesti keeles. Seda tehes on eestikeelsetel lausetel väike tõenäosus, kuid eesti keele jaoks ebakorrektsetel lausetel peaaegu olematu tõenäosus (veel väiksem). Praktikas on meile oluline, et korrektsed laused saaksid märgatavalt suurema tõenäosuse kui ebakorrektsed laused. On ilmne, et kõigi võimalike lausete peale tabelit teha on ebapraktiline, sest neid on lihtsalt nii palju, ja isegi, kui meil oleks nii palju mälu, ei saaks me iga lause kohta piisavalt statistilist infot. Seega paratamatult jaotame ka selle ülesande väiksemateks osadeks.

Kõige enam levinud keelemudel on n-gramm keelemudel. See põhineb eeldusel, et keeles saab tihti aimata, mis sõna tuleb järgmisena selle põhjal, mis talle eelnes. Näiteks sõnale "mina" võime leida statistiliselt tõendust erinevate tegusõnade järgnemisele, mis on 1. isiku vormis, kuid arvatavasti ei leia kohta, kus sõnale "mina" järgneks kohe mõni omadussõna. Täpsemalt võtab n-gramm arvesse just n-1 eelnevat sõna. Meie kasutame töös trigrammi (n=3).

Matemaatiliselt kirja panduna näeb n-gramm mudel välja selline:

$$t(E)=t(e_{1,}\dots e_m)=t(e_1)t(e_2|e_1)\dots t(e_n|e_1\dots e_{n-1})\dots t(e_m|e_{m-n+1}\dots e_{m-2},e_{m-1})$$

Trigramm, mida on kasutatud antud bakalaureuse töö lahenduses, on eelneva mudeli erijuhtum:

$$t(E)=t(e_{1,}...e_m)=t(e_1)t(e_2|e_1)t(e_3|e_{1,}e_2)t(e_4|e_{2,}e_3)...t(e_m|e_{m-2},e_{m-1})$$

Nüüd jääb üle ainult küsimus, kuidas arvutada $t(e_x|e_{x-2},e_{x-1})$. Kõige lihtsam viis on taas kasutada suurima tõepära hinnangut.

$$t(e_3|e_{1,}e_2)=\frac{count(e_{1,}e_{2,}e_3)}{\sum_w count(e_{1,}e_{2,}e_w)}$$

See tähendab seda, et arvutame, kui tihti esines protsentuaalselt pärast kahte järjestikust sõna e_1 ja e_2 kolmas sõna e_3 . Reaalselt näitab see ainult esinemise tõenäosust korpuse peal. Kui aga mingi järjestus esineb mitmel korral korpuses, siis ta esineb suure tõenäosusega sama tihti ka edaspidi.

Sellise naiivsel mudelil tekivad teatud probleemid. Esiteks, mis saab neist trigrammidest, mida korpuses ei esine. Kas me anname sellele trigrammile tulevikus tõenäosuse 0? Lisaks sealt edasi, mis siis kui meil juhtub olema trigramm, kus e_1 on e_2 ebatavaline eesliide, ning seetõttu me ei suuda sellele trigrammile anda adekvaatset hinnangut, aga kui vaadelda bigrammi (ainult e_2 ja e_3), siis näeme, et see on väga levinud. Oleks ju mõistlik võtta ka seda arvesse. Käesolevas töös tehtud keelemudelile on peale trigrammide tegemise rakendatud ka Witten-Belli silumist, mis baseerub rekursiivsel interpolatsioonil (www 1).

Interpolatsiooni võte seisneb selles, et trigrammi $(s_{1,}s_{2,}s_3)$ tõenäosuse arvutamiseks kasutame ka teadmist vastavate bigrammide ja unigrammide kohta (vastavalt $n=2$ ja $n=1$) (Koe10).

$$t_I(s_3|s_{1,}s_2)=\lambda_1 t_1(s_3)+\lambda_2 t_2(s_3|s_2)+\lambda_3 t(s_3|s_{1,}s_2)$$

$$\forall \lambda_n : 0 \le \%lamda_n \le 1$$

$$\sum_n \lambda_n = 1$$

See mudel ei ütle midagi spetsiifilist selle kohta, kuidas me saame λ väärtused. Üks võimalus oleks optimiseerida λ -d jättes osa korpusest treenimisest välja ja siis vaadata, milliste λ -de korral saame ülejäänud osas parimaid tulemusi.

Rekursiivne interpolatsioon on lihtsalt eelneva defineerimine üldjuhul. Seda võtet rakendades tahame, et madalama astme n-grammid kompenseeriksid kõrgema astme n-gramme. Lõpuks tahame osata vastata küsimustele: millal me vajame kompenseerimist? Kui usaldusväärne on antud kompenseerimine?

Rekursiivse interpolatsiooni definitsioon (Koe10):

$$t_n^I(s_i|s_{i-n+1},\dots,s_{i-1})=\lambda_{s_{i-n+1},\dots,s_{i-1}}t_n(s_i|s_{i-n+1},\dots,s_{i-1})+(1-\lambda_{s_{i-n+1},\dots,s_{i-1}})\,p_{n-1}^I(s_i|s_{i-n+2},\dots,s_{i-1})$$

Sellist definitsiooni saab rakendada mistahes n-grammi peale. Lisaks jätab see meile vabaduse määrata λ konkreetse juhtumi kohta või leida lihtsalt üldine parim λ .

Witten-Belli silumismeetod võimaldab leida λ mistahes konkreetsele n-grammile. Nimelt panid Witten ja Bell tähele, et osadel sõnadel on nende esinemissagedust arvestades palju vähem erinevaid eelnevusi kui teistel. Nad tõid näiteks ingliskeelsed sõnad *spite* ja *constant*, kus mõlemale oli 993 esinemist korpuses, kuid *spite*'le järgnes 9 erinevat sõna, samas kui *constant*'le järgnevaid erinevaid sõnu oli 415. Selline teave paneb meid mõtlema, et on palju tõenäolisem, et sõnale *constant* järgneb sõna, mis moodustaks varem mitte kohatud bigrammi, kui et see juhtuks sõnaga *spite*. Sellele toetudes defineeriti N_{1+} , mis tähendab erinevate sõnade arvu, mis järgnevad etteantud järjendile korpuses (Koe10):

$$N_{1+}(s_{1,}\dots,s_{n-1},\blacksquare)=|s_n:loend(s_{1,}\dots s_{n-1},s_n)>0|$$

Edasi saab defineerida λ arvutamise valemi:

$$1-\lambda_{s_{1,}\dots s_{n-1}}=\frac{N_{1+}(s_{1,}\dots s_{n-1},\blacksquare)}{N_{1+}(s_{1,}\dots s_{n-1},\blacksquare)+\sum_{s_n} loend(s_{1,}\dots,s_{n-1},s_n)}$$

Antud valemi idee seisneb selles, et mida rohkem erinevaid järgnevusi meil on sama hulga esinemiste korral, seda rohkem kaalu anname järgmisele madalama astmega n-grammile. Samas on oluline märkida, et eeltoodud küsimustest vastab see ainult sellele, millal me tahaksime kompenseerida ning ei vasta sellele, kui usaldusväärne on pärast madalama astme n-gramm.

2.4 Treenimine, ümberpaiknemismudel

Kui me hakkame uut lauset tõlkima, siis me saame välja tuua küll fraasid, millele me fraasitabelist vasted leiame, kuid mis järjekorras nad kokku panna? Eelnevalt tõime välja, et seda probleemi aitab lahendada keelemudel. Siin tuleb arvesse võtta, et meie puhul vaatleb keelemudel korraga ainult 3-sõnalises aknas ning ta ei anna meile otsest informatsiooni juhtude kohta, kus mõni fraas peaks ümber paiknema näiteks 4 või rohkema sõna ulatuses. Selleks on toodud juurde ka ümberpaiknemismudel. Antud bakalaureusetöös oleme kasutanud kahesuunalist fraasiümberpaiknemismudelit (ing. k. *lexicalized reordering model*).

See tähendab, et iga fraasipaari $\bar{e}$ ja $\bar{v}$ kohta on meil teada midagi nende ümberpaiknemise tõenäosuslikkusest. Täpsemalt vaatleme kolme erinevat sündmust.

- Esiteks, kas $\bar{e}$ jääb suhteliselt paigale ehk temale eelneva fraasi vaste on $\bar{v}$ eelnev fraas.
- Teiseks, kas $\bar{e}$ on vahetanud järgneva fraasiga kohad, ehk $\bar{e}$ eelneva fraasi vaste on $\bar{v}$ järgnev fraas.
- Kolmas ja viimane variant on, et $\bar{e}$ on katkendlik ehk ta ei ole ei esimene ega teine variant.

Sündmuse kindlakstegemiseks saab kasutada sõnade joondustabelit, kus fraasid $\bar{e}$ ja $\bar{v}$ moodustavad ristkülikutaolised kujundid ja siis peame vaatlema, kas leidub joonduspunkt ristkülikust vasakul üleval (esimene juht) või paremal üleval (teine juht).

Niimoodi saame erinevad sündmused korpuses kokku lugeda ja iga fraasi vaste kohta saame arvutada sündmuse tõenäosuse:

$$t_s(\text{sündmus}|\bar{v},\bar{e}) = \frac{loend\,(\text{sündmus}|\bar{v},\bar{e})}{\sum_s loend\,(s,\bar{v},\bar{e})}$$

Kuna me kasutasime kahesuunalist mudelit, siis peame seda tegema mõlema suuna peal.

3. Praktiline töö

Kogu praktiline töö on tehtud kasutades statistilise masintõlke süsteemi Moses. Tööd alustades oli lisaks olemas ka inglise-eesti paralleelkorpus ning testkorpused kvaliteedi kontrollimiseks. Lisaks oli olemas ka eesti-inglise statistilise masintõlke mudel, mis oli erinevatest korpustest kaalutult kokku pandud, et saavutada paremat tulemust.

3.1 Mudeli treenimine

Moses kasutab ka muid pakette, mis on väljaspool kirjutatud. Kasutaja saab mõnede osas valida, kas ta soovib neid installeerida või mitte. GIZA++ on aga kohustuslik Mosese töötamiseks. Meie kasutasime töös järgnevaid pakette:

- GIZA++, mis oskab kasutada IBM 1-5 algoritme ja valmis teha joondustabel, mida on vaja, et fraasi-ja ümberpaiknemistabelit valmis teha.
- IRSTLM (IRSTLM), keelemudel, mida kasutasime antud töös. Kasutab vaikimisi Witten-Bell silumist koos rekursiivse interpolatsiooniga. Lisaks on ta võimeline kasutama mitmete gigabaitide suurusi korpuseid olenemata mälust tulenatest piirangutest.

Töö jagunes kaheks osaks. Esiteks pidin Mosest kasutades treenima uue keelemudeli inglise-eesti suunal. Teiseks pidin olemasoleva eesti-inglise tõlkemudeli ümber pöörama inglise-eesti suunale, saades seega kaks erinevat tõlkemudelit. Ümberpööramine oli võimalik, sest fraasitabel oli kahesuunaline leksikograafiliste kaaludega mudel (kirjeldatud ptk. 2.2). Samamoodi oli ümberpaiknemistabel kahesuunaline. Keelemudel oli muidugi inglisekeelne, mistõttu seda eestikeelseks ümber pöörata ei ole võimalik. See tähendas seda, et mõlemad valmivad mudelid kasutasid siiski ühist keelemudelit.

Keelemudeli tegemiseks kasutasin IRSTLM paketti. Kogu mudel tehti käsuga:

```
build-lm.sh -i corpus.est -n 3 -o train.ilm.gz
```

-*i* tähendab siin sisendfaili ehk korpust, mille baasil me keelemudeli teeme.

-*n 3* tähendab, et me kasutame trigramme. Me võime anda *n* väärtuseks ka suurema või väiksema, kuid antud töös kasutasime *n* väärtuseks 3.

Lisaks on siin peidetult palju vaikimisi väärtusi. Näiteks on vaikimisi eeldatud, et silumiseks kasutatakse Witten-Belli meetodit. Kuid lisaks sellele on IRSTLM-s võimalik kasutada ka

ühte teist silumismeetodit - Kneser-Ney. Viimane ei baseeru enam rekursiivsel interpolatsioonil, vaid hoopis *back-off* meetodil, mille ideeks on vaadelda madalama astme n-gramme ainult juhul, kui kõrgema astme n-gramme ei esine üldse.

Järgnevalt, kuna IRSTLM ei väljasta keelemudelit Mosesele sobivas formaadis (.arpa), siis pidin saadud tulemuse ümber konverteerima. Selleks on olemas skript compile-lm. Kasutamiseks pakkisin eelpool saadud faili lahti ja käivitasin: *"compile-lm --text yes train.irstlm eestilm.arpa"*, mis lõi uue keelemudeli faili eestilm.arpa, mis on nüüd Mosesele sobivas formaadis.

Peale seda sai hakata treenima terviklikku mudelit. Seda käsuga:

```
train-model.perl --reordering msd-bidirectional-fe --lm 0:3:./eestilm.arpa
        --root-dir . --f eng --e est --corpus corpus/corpus >&LOG
```

Siin --reordering msd-bidirectional-fe näitab, et tuleb teha kahesuunaline msd (*monotone*, *swap*, *discontinuous*) ümberpaiknemismudel (eelpool kirjeldatud). Ülejäänud parameetritega anname ette keelemudeli asukoha, loodava tõlkemudeli kausta, korpuste laiendite nimed ja korpuste asukohad. Korpuste laienites tähendab *--f* võõrkeele laiendit ja *--e* sihtkeele laiendit (hetkel siis eesti keel). Selle käsu lahenemisel valmib fraasimudel ja ümberpaiknemismudel. Samuti valmib moses.ini fail, kus on kirjeldatud kõigi mudelite asukohad, lisainformatsioon nende kohta ja samuti kaalud kõigile kolmele mudelile.

Enne kui *train-model.perl* valmib, teeb ta päris mitu sammu läbi. Kõigepealt teeb ta sõnatabeli, kus igale sõnale vastab üks number ja selle juures on ka sõna esinemiste arv korpuses. Seejärel viiakse laused numbrilisele kujule. Seda on vaja ainult selleks, et GIZA++ saaks enda IBM Model 4 rakendada korpuse peal (seda ei ole teoorias vaja, kuid GIZA++ lahendus nõuab numbrilist kuju). IBM Model 4 rakendatakse mõlemat pidi ja selle tulemusel valmivad mõlemas suunas joondustabelid. Neid kombineerides saab paralleelkorpusele joondustabeli, mida on vaja, et fraasitabelit treenida. Muidugi jääb IBM Model 4-st üle ka leksikograafiline tõlkemudel, mida saab ära kasutada, et arvutada fraasimudelis mõlemale suunale leksikograafilised kaalud.

Edasi valmivad fraasitabel ja ümberpaiknemistabel teoorias kirjeldatud viisil.

3.2 Mudeli ümberpööramine

Lisaks uuele mudelile valmis ka teine mudel, mis saadi olemasoleva eesti-inglise mudeli ümberpööramisest. Kuna olemasolev mudel oli kasutanud samu algoritme fraasi- ja ümberpaiknemismudeliteks ning nende formaadid olid teada, taandus probleem skripti kirjutamisele, mis pööraks tabeli reahaaval ümber ja viimaks sorteeriks kogu tabeli.

Arvestada tuli failide suurusega. Fraasitabel oli 15 GB ja ümberpaiknemistabel 66 GB. Viimast ei saanud seega tervenisti mälusse lugeda. Esimene lahendus toimis nii, et kasutati C++ *vector*'it, kuhu salvestati kõik read uuel kujul ning kõige lõpuks rakendati *vector*'ile *sort* meetodit. Sellist lahendust kasutades pidi ümberpaiknemistabeli ümber pöörama osade kaupa, võttes aluseks algussümboli vahemikud. Samas on lihtsam lahendus see, kui lasta viimane sorteerimine teha Linuxi enda sisse ehitatud *sort* skriptil ja enda skripti sees saab mälus korraga hoida ainult ühte rida, mille me pärast töötlemist kohe kirjutame väljundfaili.

3.3 Häälestamine

Kõige lõpuks kasutasime Moseses olevat *mert* skripti mõlema mudeli häälestamiseks, mille eesmärk oli leida õiged kaalud mudelitele ja fraasimudeli puhul ka tema komponentidele. Selleks kasutasime käsku:

```
mert-moses-multi.pl /home/smt/corpora/_devtest/devcorpus/clean.eng
/home/smt/corpora/_devtest/devcorpus/clean.est /home/smt/bin/moses
./pööratud/moses.ini --working-dir ./pTuning/mert --threads=6 --decoder-
flags="-threads 6 -use-persistent-cache false -v 0"
```

Idee seisneb selles, et me katsetame erinevate kaalude tulemuslikkust muul korpusel kui see, mille põhjal me mudeli treeninud oleme. Parameetriteks on ette antud inglise testkorpus, eesti testkorpus, Mosese asukoht, häälestatava mudeli moses.ini fail (mis sisaldab kõigi vajalike teiste mudelite asukohad) ning kaust, mida häälestamisprotsess saab kasutada andmete talletamiseks. Viimased kaks parameetrit lasevad meil kasutada rohkem süsteemi ressursse, kuna vaikimisi käivitaks ta ainult ühe lõime, mistõttu jääb mitmetuumalisel protsessoril palju jõudlust kasutamata.

Tulemuseks loodi uus *moses.ini* fail koos uute kaaludega.

4. Analüüs

4.1 Tõlkimine

Kogu töö eesmärk on olnud uurida masintõlkimist inglise keelest eesti keelde. Nüüd, kui mudelid on valmis, saame me seda lõpuks teha. Mosesel on tõlkimiseks kaasas oma töövahendid. Meie kasutasime kahte skripti nimedega: *filter-model-given-input.pl* ja *moses*. Esimene skript teeb uue ajutise mudeli, kus tabelid on kärbitud ainult nendeks kirjeteks, mida antud tõlkimisel vaja läheb. Seda oli vaja kasutada, sest tõlkimisskript *moses* läks algsel juhul mälu kasutuses üle piiride. Skripti kasutamine:

```
filter-model-given-input.pl ./filteredmodel/ ./model/moses.ini ./text.eng

  moses -f ./filteredmodel/moses.ini < ./text.eng  > ./tõlked/tõlge1.est
```

Nagu näha, tuleb mõlemale skriptile sisendiks anda sama ingliskeelne algtekst. Muidu filtreeritakse kokku valed read. *filtered-model-given-input.pl* võtab lisaks sisendiks algse moses.ini faili ja teeb uue mudeli etteantud kausta (*./filteredmodel/)*, kuhu tekib uus moses.ini fail, mida *moses* skript kasutab tõlkimiseks. Viimaseks antakse *moses* skriptile ette tõlke väljastamise asukoht.

4.2 Automaatne hindamine - BLEU

Masintõlkimises, eriti kui tahta katsetada mudeleid erinevate parameetritega või katsetada erinevaid algoritme, on väga oluline saada kiiresti tagasidet mudeli paranemisest või halvenemisest. Selleks kasutatakse automaatseid hindamismeetodeid.

Meie kasutasime mudelite hindamiseks 1000 lausest koosnevat testkorpust ja BLEU-4 (BLEU) automaatse hindamise meetodit. BLEU on laialt levinud ning baseerub ideel, et ta hindab masintõlke ja testkorpuses oleva inimtõlke vahelist sarnasust. Täpsemalt, vaatleb BLEU masintõlkes olevaid n-gramme, unigrammist kuni 4-grammini, ning loendab, mitu neist esineb inimtõlkes iga n-grammi astme kohta.

$$BLEU-4 = lühiduskaristus \times \prod_{i=1}^{4} täpsus_i$$

$$lühiduskaristus = min\left(1, \frac{masintõlke\ pikkus}{päristõlke\ pikkus}\right)$$

$$täpsus_i = \frac{i-gramme\,, mis\ esinevad\ ka\ päris\ tõlkes}{i-grammide\ arv\ masintõlkes}$$

Lühiduskaristus võetakse kasutusele, et mitte anda hea hinnang lausetele, kus on sõnu lihtsalt puudu. Me tõime välja valemi, mida saab rakendada ühele lausele. Lõpptulemus saadakse, rakendades seda valemit kõikidele lausetele ja võttes siis keskmise.

Moseses on BLEU-4 kasutamiseks olemas skript *multi-bleu.pl*. Kasutus on sellel järgmine:

```
multi-bleu.pl inimtõlge1, inimtõlge2... < masintõlge
```

Nagu näha, saab anda ette mitu erinevat inimtõlget. Sellisel juhul vaatab skript i-grammi esinemisi mistahes inimtõlkes, kuid võtab arvesse, et kui mingi i-gramm esineb masintõlkes n korda, siis teda ei loeta rohkem kui kõige rohkem esinemisi ühes kindlas inimtõlkes.

Rakendades BLEU-4 meetodit valminud mudelitele, saime skoorideks 0,16 (treenitud) ja 0,12 (ümberpööratud). Võrdluseks sai mudel, mida me ümber pöörasime, vastupidisel suunal sama korpuse peal tulemuseks 24 %. Lisaks saime väljundiks kattuvuste protsendid n-grammide kaupa unigrammist kuni 4-grammini. Mõlemal juhul algasid need unigrammide juures 45% lähedalt, langedes umbes kaks korda iga järgmise n-grammi juures, mis tähendab, et sõnu tõlgib mudel üsna tihti õigesti, kuid õigeid tervikfraase leiab harvemini. Siinkohal tuleb arvestada, et võrreldi ainult ühe etalontõlkega, mis tähendab, et tulemus võiks olla parem, kui hindamisel kasutada mitmeid erinevaid tõlkeid.

4.3 Analüüs - käsitsi

Järgnevalt vaatame lähemalt saadud tulemust erinevate näidete varal. Lähemalt vaadeldes oli põhilisteks probleemideks konteksti eiramine, käänamine ja lause ülesehitus. Esines ka muid vigu. Järgnevalt toome mõned näited.

Näitelause inglise keeles: *the famous belgian detective hercule poirot operates with captain hastings as his right hand in the underworld of london .*

Selle lause tõlge treenitud mudelil: *kuulus belgia uurija heraklese poirot toimib koos kapten hastings , kui ta parem käsi allilmas londonis .*

Antud lause on päris keeruline ning siin on näha sobimatust konteksti. *Toimib* asemel sobib paremine sõna *tegutseb*. Hastings ei ole käänatud, sest tegu on nimega, ning nimede esinemine käänatud vormis on veel ebatõenäolisem kui sõnadel - mis on kindlasti üks puudujääke statistilisel masintõlkimisel. Siin on näha, kuidas fraas *"as his right hand"* on tõlgitud *"kui ta parem käsi"*, mis oleks muidu väga hea tõlge, aga hetkeolukorras on

ebakorrektne ning stiililiselt halb. Tulles tagasi sõna *toimib* juurde, siis see, mis vihjaks, mis sõna peaks selle asemele minema, on *uurija*, mis on aga eraldatud inimese nimega, mistõttu me ei saa seda järeldada ei fraasimudelist ega keelemudelist ka kõige parema tahtmise juures.

Teine näitelause i. k: *möttöla is the most famous finnish basketball player* .

Tõlge treenitud mudelil: *möttöla on kõige kuulsam soome korvpalli mängija* .

Siin on näha väga head tõlget. Kuna lause sisaldab levinuid fraase, siis nende õige tõlkimine ei ole mingi probleem, sest need fraasid esinevad koos alati käänatuna.

Kaks lauset inglise keeles: *five ducks were swimming on the still smooth water* .

how to change the world into a better place ?

Neile vastavad tõlked: *viis parti ujuvad on endiselt pehme vesi* .

, kuidas muuta maailm paremasse kohta ?

Siin on näide sellest, kuidas on valesti käänatud ning esimese lause puhul ka valesti tõlgitud ning sõnast *still* valesti aru saadud. Kõige üllatavam asjaolu on see, et koma on pandud lause algusesse. On arusaadav, et ", kuidas muuta" võib fraasimudelis olla hea tõenäosusega, kuid selle peaks parandama keelemudel.

Viimane näide ingliskeelsest lausest: *it is necessary to create new laws to restrain the consumption of alcohol* .

Selle lause tõlge: *on vaja luua uusi seadusi piirata alkoholi tarbimist* .

Siin on näide, kus lause ei ole ilusti kokku seotud sõnaga et. Lähemalt uurides: fraasile 'to restrain' ei ole ühtegi vastet, mis kasutaks *et*-i. Miks see nii on? Selleks võib olla mitmeid põhjuseid. Esiteks ei pruugi meil korpuses olla sellist kasutust sõnaga *restrain* ja teiseks on võimalik, et joondustabel on teinud joonduse kohta valed järeldused ning "*, et*" joondanud mõne teise sõna juurde, mistõttu ei ole me vastavat fraasi kätte saanud.

Kokkuvõtvalt, tõlkemudel sai hästi hakkama paljude levinud väljendite ja lihtsamate lausetega. Keerulisemate lausetega tekkisid konteksti-, käänamis- ja sidumisvead ning tihti oli lausetel ebakorrektne ülesehitus. Harvem leidus ka selliseid vigu, kus koma oli näiteks lause alguses, tõlkesse oli jäetud alles inglisekeelne sõna või mõni sõna oli tõlkides üldsegi ära kadunud. Et jõuda korrektse tõlkeni, peavad seda mingil määral toetama kõik mudelid. Kui ükski neist eksib piisavalt suurel määral, mõjutab ta tulemust oluliselt. Kuna tegu on statistiliste meetoditega, mis leiavad seaduspärasusi piiratud arvu heuristiliste meetoditega (3

mudelit), siis paratamatult võib tekkida ettearvamatuid tulemusi. Samas tuleb arvesse võtta, et tegemist on ühe esimese katsega luua statistilist masintõlkemudelit inglise-eesti tõlkesuunal, ning tulemus näitab, et antud tööd on põhjust edasi arendada ning kindlasti on võimalik leida lahendusi selle parandamiseks.

Viimaseks toon välja mõned ideed, mida tulevikus ellu viies võib saada paremaid tulemusi.

Väga oluline eesti keelde masintõlkimise seisukohalt on see, et meil on 14 käänet. See tähendab, et teoorias võib meil sama sõna esineda 14 erineval moel, aga statistiline mudel vaatleb neid kõik kui eraldi sõnu nägemata neis mingit seost. Teisest küljest, kuna me kasutame fraasimudelit, siis fraasi mastaabis ongi igal eraldi käänatud sõnal täiesti oma tähendus ja tegelikult võib see meid aidata hoopis paremini õigeid fraase kokku panna. Samas see ikkagi eeldab, et me saame piisavalt palju andmeid iga sõna käänatud versiooni kohta.

Üks idee on treenida keelemudel ainult konkreetsele valdkonnale, näiteks ilukirjandus, lastekirjandus või juriidiline tekst ning pärast teha ka test ainult antud valdkonnale. Kui selle läbi saaks mudeli, mis tõlgiks antud valdkonda paremini kui üldine mudel, siis sellel oleks juba praktiline kasutus. Võib püstitada hüpoteesi, et selline lähenemine aitaks parandada vigu kontekstist arusaamisel, mida oli hetkel kõige rohkem.

Kokkuvõte

Käesolevas töös on käsitletud statistilist masintõlget nii teoreetiliselt kui ka praktiliselt. Statistiline masintõlge on valdkond, mis üritab panna arvutit tõlkima, ilma et ta teaks midagi keelte ametliku grammatika kohta, vaid saab sisendiks ainult paralleelkorpuse ehk miljoneid lausepaare, kus üks paariline on teise paarilise tõlge.

Praktilises pooles kasutati olemasolevat Mosese statistilise masintõlke raamistikku, et luua uus tõlkemudel inglise-eesti suunal. Lisaks pöörati ümber olemasolev eesti-inglise tõlkemudel, mis oli kaalutult kokku pandud erinevatest korpustest saadud mudelitest. Kogu töö käigus loodi 1 keelemudel, 2 fraasimudelit ja 2 ümberpaiknemismudelit.

Teoreetiline osa oli referatiivne ning käsitles just neid fraasi-, keele- ja ümberpaiknemismudeli algoritme, mida me sisuliselt kasutasime töö praktilises osas. Täpsemalt käsitleti kahesuunalist leksikograafiliste kaaludega fraasimudelit, trigramm keelemudelit, mis kasutas silumiseks rekursiivset interpolatsiooni koos Witten-Belli meetodiga ning kahesuunalist msd (*monotone*, *swap*, *discontinues* ehk jääb paigale, vahetab, katkendlik) ümberpaiknemismudelit.

Töö lõpus tõlgiti rohkem kui tuhandelauseline testkorpus ja hinnati saadud tulemust automaatse hindamismeetodiga BLEU. Lisaks vaadeldi tulemust lähemalt käsitsi. Kuigi paljud kerged laused tõlgiti peaaegu ideaalselt, siis keerulisemate lausetega hakkasid vähemalt osaliselt tekkima raskused. Suurim probleem oli konteksti mittemõistmine, sellele järgnesid käänamine ja lause ülesehitus.

Töö väljundiks on valmiv statistilise masintõlke mudel inglise-eesti suunal ning teadmine, et antud valdkond on perspektiivikas. Töö on lisaks mõeldud inglise-eesti suunal statistilise masintõlke tegemise alustamiseks.

Summary: Inverting an Estonian-english statistical machine translation model

Bachelor Thesis

Indrek Klanberg

The present thesis is about statistical machine translation in both theoretical and practical manner. Statistical machine translation is an area, which aims to make the machine translate without giving it any knowledge about grammar of the languages. It only receives a parallel corpora with millions of sentence pairs, where the only certain knowledge is, that in each pair, the sentences translate to each other.

In the practical part of the work, we used statistical machine translation framework Moses, to create a new language model for English-Estonian direction. In addition an existing opposite direction language model, which was built from different corporas and put together weigthed, was inverted. Throughout the work 1 language-model, 2 phrase-models and 2 reordering-models were created.

Theoretical part of the work involved describing different algorithms that were used inside the framework and its components in the practical part. To be more precise we discussed bidirectional phrase-model with lexical weights, tri-gram language model with recursive interpolation and Witten-Bell smoothing and bidirectional msd(monotone, swap, discontinues) reordering model.

At the end stage of the work a test corpora with more than 1000 sentences was translated using the created models. Result was measured with automatic evaluation method BLEU. In addition, the result was examined close up and even though there were many good to allmost perfect translations for simpler sentences, in more complex sentences, there started to exist errors. Most common was misunderstanding the contex. Others worth mentioning were wrong inflection and bad sentence structure.

As the result of the work a English-Estonian machine translation model was made and we came to the conclusion that it is a promising field for English-Estonian translation. The work at hand is also meant for educational purposes for anyone willing to step into statistical machine translation field for English-Estonian direction.

Viited

- Koe10. "Statistical Machine Translation", Philipp Koehn, School of informatics, University of Edinburgh, 2010
- www1. http://hermes.fbk.eu/people/bertoldi/teaching/lab_2010-2011/img/irstlm-manual.pdf (10.05.2012) - Mosese komponendi IRSTLM juhend.
- www2. http://www.statmt.org/moses/ (10.05.2012) statistilise masintõlke raamistiku Moses koduleheküg
- TMcom. Rico Sennrich; Perplexity Minimization for Translation Model Domain Adaptation in Statistical Machine Translation; Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, Avignon, France, 2012
- Moses. Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, Evan Herbst; Moses: Open Source Toolkit for Statistical Machine Translation, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, demonstration session, Prague, Czech Republic, 2007
- IRSTLM. Marcello Federico, Nicola Bertoldi, Mauro Cettolo; IRSTLM: an Open Source Toolkit for Handling Large Scale Language Models, Proceedings of the 9th Annual Conference of the International Speech Communication Association, Brisbane, Australia, 2008
- BLEU. Kishore Papineni, Salim Roukos, Todd Ward, Wei-Jing Zhu; Bleu: a Method for Automatic Evaluation of Machine Translation, Proceedings of 40th Annual Meeting of the Association for Computational Linguistics