TARTU ÜLIKOOL
Arvutiteaduse instituut
Informaatika õppekava

Kairit Peekman

Automaatse lausestamise ja sõnestamise hindamine uue meedia keele korpusel

Bakalaureusetöö (9 EAP)

Juhendaja: PhD Kairit Sirts

Tartu 2020

Automaatse lausestamise ja sõnestamise hindamine uue meedia keele korpusel

Lühikokkuvõte:

Veebis leidub palju tekste, mis ei ole ortograafiliselt korrektsed (nt foorumite sissekanded, inimestevaheline suhtlus kommentaarides, jututubades jm). See on nn uue meedia keel ehk internetikeel. Bakalaureusetöös vastatakse küsimusele, kui hästi töötavad kolm tekstitöötlusvahendit (EstNLTK, UDPipe ja StanfordNLP) uue meedia keele teksti lausestamisel ja sõnestamisel. EstNTLK sõnestab reeglipõhiselt ja lausestab mudelipõhiselt reeglipõhise järelkontrolliga, UDPipe'il ja StanfordNLP-l on sõnestamiseks ja lausestamiseks eeltreenitud eesti keele mudelid. Kõigil kolmel on uue meedia keele tekstide lausestamisel veel arenguruumi, kuid EstNLTK ja StanfordNLP tulemused olid paremad kui UDPipe'il. Sõnestamise tulemused erinesid vähem ja olid üldiselt head, sest F-skoor oli üle 95%.

Võtmesõnad:

Lausestamine, sõnestamine, tehisnärvivõrk, uue meedia keel

CERCS: P175, Informaatika, süsteemiteooria

Evaluation of Automatic Sentence and Word Tokenization on the Corpus of New Media Language

Abstract:

There are many texts on the web that are not orthographically correct (eg forum posts, people-to-people comments, chat etc.). This is so called new media language or Internet language. The Bachelor thesis answers the question how good are the tokenizers of three language processing tools (EstNLTK, UDPipe, Stanford NLP) for new media language texts. EstNTLK word tokenizer is rule-based and sentence tokenizer is model-based with rule-based follow-up, UDPipe and StanfordNLP have pre-trained Estonian language models. All three still have room for improvement in sentences tokenization of new media language texts, but EstNLTK and StanfordNLP performed better than UDPipe. The results of the words tokenization differed less and were generally high, as the F-score was over 95%.

Keywords:

Sentence Tokenization, word Tokenization, neural network, new media language

CERCS: P175, Informatics, systems theory

Sisukord

Sissejuhatus

Louis Hjelmselv kirjutab oma teoses „Sissejuhatus keeleteooria alustesse“ keelest nii: „Keel on lahutamatult seotud inimesega ja järgneb talle kõigis ta tegemistes. Keel on tööriist, millega inimene vormib mõtet ja tunnet, meeleolu, püüdlust, tahet ning tegutsemist; keel on vahend, millega inimene mõjutab ja millega inimest mõjutatakse; keel on inimühiskonna olemasolu viimane ja kõige sügavam eeldus“ [1, lk 13]. Inimene kasutab seega keelt näiteks rääkides, kirjutades, mõeldes, meenutades, filme vaadates ja lugedes [2]. Kuigi kõne osatähtsus arvuti abil või arvutiga suhtlemises on suurenenud, kasutatakse arvuti abi siiski peamiselt keelt kirjutades ja lugedes.

Kasutajale paremini kohanduvate programmide loomiseks ja inimestele vajaliku info töötlemiseks on arvutil vaja oskust töödelda inimese loomulikku keelt. Samuti on see oskus vajalik selleks, et kasutada arvutiprogramme keele uurimiseks. Inimese loomuliku keele oskus on nii kompleksne nähtus, et selle oskuse andmine arvutile tähendab väga paljude selliste tegevuste automatiseerimist, mis inimese jaoks toimuvad ilma suuremat tähelepanu nõudmata (näiteks oskus eristada tekstis lauseid ja lausetes sõnu).

Keeleteaduse ja arvutiteaduse erialadevahelist valdkonda, mis uurib, kuidas loomulikku keelt arvuti abil kirjeldada ja analüüsida, nimetatakse arvutilingvistikaks. Rakenduslikust küljest tegeleb keeletöötlusvahendite ja keeleressurssidega keeletehnoloogia [3]. Keeletöötlusvahendid on muu hulgas tekstitöötlusvahendid, mis võimaldavad analüüsida teksti morfoloogiliselt ja süntaktiliselt. Sellised tekstitöötlusvahendid töötavad teksti väiksemate üksustega. Morfoloogilise analüüsi jaoks on vaja tekstis tuvastada sõnad, ent süntaktilise analüüsi jaoks ka laused. Tekstitöötlusvahendid pakuvad sellistel juhtudel teksti sõnestamise ja lausestamise funktsiooni. Tekstianalüüsi seisukohalt ei pruugi sõnestamine ega lausestamine olla triviaalne, sest näiteks punkt sõna järel ei tähista alati lauselõppu [4]. Samuti ei pruugi lause kui terviklikku sõnumit väljendava üksuse piir langeda kokku ortograafilise lause piiriga.

Keeletehnoloogilised rakendused kasutatavad sõnestamiseks ja lausestamiseks peamiselt kahte meetodit: statistilist ehk andmejuhitud meetodit ja reeglipõhist meetodit. Statistiline meetod omandab teadmisi korpusi statistiliselt analüüsides. Reeglipõhise meetodi puhul kodeerivad eksperdid kõigepealt grammatilised reeglid ja seejärel koostavad sõnade nimestikud. Reeglite koostamine on aeganõudev ja mahukas töö [5]. Suurem osa statistiliste meetodite mudelitest on omandanud teadmised korpuste põhjal, mis vastavad kirjakeele normile. Ka reeglipõhised meetodid on sageli loodud kirjakeele norme arvestades.

Koos arvutite ja internetiühenduse levikuga on tunduvalt suurenenud inimsuhtlus veebirakenduste vahendusel ehk võrgusuhtlus (nt jututubades, foorumites, kommentaarides ja uudisegruppides). Sageli

ei peeta sellises suhtluses kinni kirjakeele normist, vaid kirjutatakse nii, nagu kasutatakse keelt igapäevases suulises suhtluses. Siinses töös on sellist internetis võrgusuhtluseks kasutatavat keelt nimetatud uue meedia keeleks või ka internetikeeleks [6].

Muischnek jt [6] toovad esile, et uue meedia keel erineb kirjakeele normile vastavast keelest nii sõnade kui ka õigekirja poolest. Nende hinnangul kasutatakse uue meedia keeles rohkem partikleid, emotikone, uudissõnu, lühendeid, toorlaene jm. Väga sageli ei järgita uue meedia keeles õigekirjareegleid just kirjavahemärkide puhul: neid jäetakse ära või kasutatakse rõhutamiseks korduvalt, ning ka suur- ja väiketähti ei kasutata kirjakeele normi alusel. Erinevusi on piisavalt, et tekiks küsimus, kas praegusaja tekstitöötlusvahendid saavad uue meedia keele töötlusega teksti tükeldamise tasandil hakkama.

Bakalaureusetöös otsitaksegi vastust küsimusele, kui hästi töötavad valdavalt kirjakeele normi alusel loodud statistilised ja reeglipõhised meetodid uue meedia keele sõnestamisel ja lausestamisel. Selleks võrreldakse kolme automaatse tekstitöötlusvahendi (EstNLTK [7], UDPipe [8] ja StanfordNLP [9]) lausestamise ja sõnestamise tulemusi manuaalselt annoteeritud korpusega. Valitud kolmest tekstitootlusvahendist on EstNLTK peamiselt reeglipõhine ja ülejäänud kaks põhinevad täielikult statistilisel mudelil.

Varem on põhjalikumalt uuritud uue meedia keele olemust ja eripära [10-14]. Selle morfoloogilise analüüsiga on tegelenud Kadri Muischnek jt [6] ja süntaktilise analüüsiga Dage Särg [15]. Need autorid kasutasid töödeldava ühikuna muu hulgas lausungit, st kasutaja ühe korraga järjest väljendatud teksti, mis võib kirjakeele mõttes sisaldada mitut lauset. Siinse töö autorile teadaolevalt ei ole uue meedia keele lausestamist ja sõnestamist olemasolevate tekstitöötlusvahenditega seni uuritud.

Uurimistöö alguses on täpsemalt selgitatud, mille poolest erineb uue meedia keel õigekirjareeglitele vastavast kirjakeelest ning kuidas mõjutavad need erinevused sõnestamist ja lausestamist. Seejärel tutvustatakse töös kasutatud korpust ja selle manuaalset annoteerimist. Lõpuks võrreldakse erinevate lausestajate ja sõnestajate tulemusi.

Kasutatavad terminid

Annotatsioon – lausepiiri tähistav märgendite kogum.

Annoteerimine – lausepiiride tuvastamine ja märgendamine.

CoNLL-U formaat – tekstifailis teksti esitamise vorm, kus tekst on sõnestatud ja lausestatud ning sõned on lausete kaupa esitatud kümne kindla infovälja kaudu.

Keel – inimese olulisim suhtlusvahend, mis mõtete ja tunnete väljendamiseks kasutab sõnu ja väljendeid ning mida hoiab koos teatav struktuur ehk grammatika [16].

Kirjakeel – ühtne, korrastatud, kirjas ja kõnes kasutatav keelekuju [16].

Kood – tarkvaraprogramm või selle osa (nt lähtekood) [16].

Lause – keelelise suhtluse põhiüksus, mis väljendab terviklikku sõnumit (väidet, küsimust, käsku, soovi vms) [16].

Lausestaja – automaatse tekstitöötlusvahendi lausestamisega tegelev kood.

Lausestamine – teksti tükeldamine lauseteks.

Märgend ehk **märgis** – kokkuleppeline tunnus, mida kasutatakse lause lõpus lause liigi kohta info andmiseks.

Sõna – keele väikseim iseseisev tähenduslik koostisosa [16].

Sõne – tervikuna käsitatav sümbolijada või üksiksümbol [16].

Sõnestaja – automaatse tekstitöötlusvahendi sõnestamisega tegelev kood.

Sõnestamine – teksti tükeldamine sõnadeks.

Süntaktiline lause – üksus, mis väljendab terviklikku sõnumit.

Tarkvarakonveier ehk **toru** (ingl *pipeline*) – mitmest protsessist koosnev tarkvara, kus ühe protsessi väljundvoog muudetakse automaatselt järgmise protsessi sisendvooks (nt UDPipe, StanfordNLP) [17].

Tekst – kirjutatud või trükitud sõnade harilikult mõtestatud järjend [16].

Tekstikorpus, ka **korpus** – kirjalikest või suulistest tekstidest koosnev elektrooniline andmekogu keeleteaduses [16].

Tekstitöötlusvahend – tekstiliste andmete töötluseks ja haldamiseks mõeldud tarkvara: speller, morfoloogiline analüsaator, sõnastike haldamise tarkvara, nimisõnafraaside märgendaja jms [18].

Uue meedia keel ehk **internetikeel** – suhtlusvõrgustikes ja mujal internetis kasutatav, harilikult kõnekeelseid väljendeid sisaldav keel [16].

1. Uue meedia keel ja teksti töötlemine

EKI ühendsõnastiku 2020[1] järgi on tekst kirjutatud või trükitud sõnade mõtestatud järjend. Valdav osa jäädvustatud tekstist (raamatud, artiklid, kirjad jms) vastab õigekirjareeglitele, st on nii-öelda standardne. Internetiühenduse ja arvutite (ka nutitelefonide) levikuga on alates 1990. aastatest suurenenud ebastandardse teksti maht, sest inimesed on hakanud omavahel palju suhtlema uue meedia vahendusel.

Võrgusuhtluses kasutatud keelt hakati nimetama internetikeeleks [19], hiljem lisandus termin „uue meedia keel". Siinses uurimistöös on kasutatud termineid „uue meedia keel" ja „internetikeel" sünonüümidena, kuid eelistatud on terminit „uue meedia keel". Uue meedia keelel on oma erijooned, mida tundes on lihtsam mõista, miks ei pruugi standardse teksti töötlemiseks mõeldud vahendid töötada uue meedia keelt sisaldava tekstiga.

1.1. Uue meedia keele erinevused standardsest keelest

Uue meedia keele erijooni on mõistlik vaadata sõna ja lause tasandil, kuna mõlemad on siinse uurimuse jaoks olulised tekstitöötlusüksused. Sõna tasandil on uue meedia keele omapärasid analüüsinud Muischnek jt [6], kes tõid oma artiklis esile seitse sõnarühma, mida võib pidada uue meedia keele erijoonteks:

1) partiklid, mis on eeskätt suulises tekstis esinevad rõhu- ja abisõnad [16] (nt *irw, ok, ve, we, aa, asoo, jaja, jep, nunuh, auts, icc, krt, oih, wau, tre, tsau, sau, saux, kle, kule, kuule, vata, vaata, ota, oota, ee, hmm, ütleme, ytleme, ommik*);

2) emotikonid, mis on peamiselt kirjamärkidest koostatud piltmärgid emotsioonide väljendamiseks veebisuhtluses [16] (nt *: P , :) , :D*);

3) kirjapildi sihiliku muutustega sõnavormid (nt *raffas, vaffa, koff, näitex, mix, olex, ex, plix, kirurgix, ajatäitex, ärkax, tõestax, kyll, ei viici, täica, ülbicema, veremaice, h2sti, h6be, t88, meez, ommik, uvitav, ullem, hahahahaha, ehhhhh*);

4) väikse tähega või kirjapildi muutustega kirjutatud pärisnimed ehk tegelikud nimed (nt *krizzy, sokrates, abja paluoja, eesti*);

5) võõrkeelsed sõnavormid ja toorlaenud (nt *diskilt bootida, SpyBotil advanced mode- > ignore products-> linnuke ette DSO Exploit ja exit, luk huus tooking, ,, but no !!*);

[1] https://sonaveeb.ee/.

6) normitud kirjakeele seisukohalt valed sõnavormid (kõnekeelsused, murdevormid, slängisõnad, lühenenud sõnad ja sõnakatked, kirjakeele vormimoodustusnõuetele mittevastavad sõnavormid, nt *loogish, logish, friik, tydo, privama ja ruulima, suht, tegelt, norm, aint, kellegil, kellegile, kellegiga, millegist, millegigi, eiole, midaiganes, niiet, eksole, minuteada, midagist, kudagi, mudu, ikkagist, kussa, mingine, lissalt 'lihtsalt', ikki, prääga, õhtast, jummala*);

7) trükivigadega sõnavormid.

Muischnek jt [6] tõdesid, et uue meedia keel erineb standardsest keelest nii sõnavara kui ka õigekirja poolest. Morfoloogilise analüüsi teeb see keeruliseks, kuna analüsaatori sõnastikus ei esine kasutatud sõnu või ei suuda analüsaator kirjapildi erinevuse tõttu seostada sõna sõnastikus oleva sõnaga (nt tähejärjendeid on asendatud või pole jälgitud suur- ja väiketähe õigekirjareegleid) [6].

Teine siinse uurimistöö seisukohalt oluline aspekt on lausete tuvastamine. Uue meedia keeles on lause alguses suurtähe ja lõpus kirjavahemärgi kasutamine juhuslik [15]. Kui peamiselt on uuritud jututubade, uudisegruppide, foorumite ja kommentaaride tekste, siis neist on just kommentaaride keel olnud kirjakeelega kõige sarnasem. Jututubade tekstides on seevastu olnud kõige keerulisem lauseid tuvastada, mistõttu on sageli piirdutud n-ö lausungiga, st jututubade lõikudega (ühe kasutaja postitus). Analüüsimiseks on lausestatud uudisegruppide, foorumite ja kommentaaride tekste automaatselt kirjakeele normi alusel [6], st kõik sellele mittevastavad laused on jäänud tuvastamata.

1.2. Teksti lingvistiline analüüs

Keeletehnoloogilised rakendused põhinevad automaatsel lingvistilisel analüüsil. Selle peamised liigid on [5]:

1) morfoloogiline analüüs,
2) süntaktiline analüüs,
3) semantiline analüüs.

Morfoloogilise analüüsi käigus tuvastatakse sõna algvorm ehk lemma ja grammatiline kategooria [6]. Süntaktilise analüüsi raames uuritakse lauseid, näiteks sõnade funktsioone lauses või lause puukujulist struktuuri [15]. Semantilise analüüsi eesmärk on leida kontekstist lähtuvalt sõnadele sobiv tähendus (ühestamine) või nimisõnadele asesõnad, asendada väljendid ja selgitada välja lause tähendus [5].

Morfoloogilise analüüsi ja ühestamise tulemus on süntaktilise analüüsi sisendiks, mõlema tulemused on omakorda semantilise analüüsi sisendiks [6]. Teksti automaatseks analüüsimiseks on vaja tuvastada selle

sõnad ja laused. Levinuimad tekstitöötlusvahendid on tavaliselt varustatud sõnestaja (sõnestamisega tegelev kood) ja lausestajaga (lausestamisega tegelev kood).

1.2.1. Sõnestamine

Keeleteaduses räägitakse „sõnast“, mis on keele väikseim iseseisev tähenduslik koostisosa; selle kirjas või kõnes esinev vorm, mida kirjapildis eraldavad tühikud ning suulises kõnes rõhud ja pausid [16]. Keeletehnoloogias aga räägitakse „sõnest“, mis on tervikuna käsitatav sümbolijada [16].

Kui sõneks sobib igasugune sümbolijada või üksiksümbol, siis kirjakeele normile vastavaks sõnaks sobivad tähtedest koosnevad sümbolijadad või üksiksümbolid, millel on tähendus. Tavaliselt on üks sõna eristatud teisest vaba täheruumiga.

Üldtunnustatud tekstitöötlusreeglite järgi kirjutatakse paljud kirjavahemärgid vahetult sõna järele ilma vaba täheruumita, teatud märgid ka sõna ette (nt alustavad jutumärgid, alustav kaksisülakoma, alustav sulg). Kirjavahemärk või nende kombinatsioon on keeletehnoloogias „sõne“, kuid need võivad olla ka „sõnad“ keeleteaduses.

Sõnestamisel tuvastatakse sõnad ja kirjavahemärgid, eraldatakse need üksteisest ning tulemusi hoitakse eraldi elementidena üldjuhul mingis andmekogus. Kirjakeele normile vastava teksti puhul on see ülesanne lihtsam kui uue meedia keele teksti puhul, sest uue meedia keeles ei pruugi olla sõna eelmisest tühikuga eraldatud. Keeruliseks teevad sõnestamise ka emotikonid, mis on uue meedia keeles iseseisvat tähendust kandvad sõnad, ja kirjavahemärkide korduvkasutamine (nt rõhutamise eesmärgil) viisil, mis ei vasta kirjakeele normile. Tuleb eristada, kas tegu on kirjavahemärkide korduvkasutamise juhuga või emotikoniga, mis koosneb enamasti kindlast märgikombinatsioonist, mis annab edasi selle tähenduse.

1.2.2. Lausestamine

Lause on keelelise suhtluse põhiüksus, mis väljendab terviklikku sõnumit (väidet, küsimust, käsku, soovi vms) [16]. Ortograafiline lause algab suurtähega ja lõpeb lauselõpumärgiga. Lause kui terviklikku sõnumit väljendava üksuse piir ei pea langema kokku ortograafilise lause piiriga [20]. Selles mõttes võib lause koosneda mitmest ortograafilisest lausest (nt *Kõik nägid seda pealt. Aga ometi ei teinud keegi midagi.*) ja vastupidi, üks ortograafiline lause võib koosneda mitmest lausest (nt *Siin on naine, kellele on äsja teatatud, väga saamatult muide, et ta abikaasa sõidab teisega ära, <u>see</u> naine jumaldab oma meest ja tema ainuke reaktsioon on vastus.*) [20].

Tavaliselt tuvastavad tekstitöötlusvahendite lausestajad ortograafiliste lausete piire. Sellised lausestajad kasutavad peamiselt statistilisi ehk andmejuhitud või reeglipõhiseid meetodeid. Statistilised meetodid omandavad teadmised korpusi statistiliselt analüüsides. Reeglipõhiste meetodite puhul kodeerivad eksperdid kõigepealt grammatilised reeglid (näiteks regulaaravaldiste kaudu) ja koostavad seejärel sõnade nimestikud [5].

Eesti keele automaattöötlusvahendeid koondava teegi EstNLTK dokumentatsioonis on teksti tükeldamise kohta kirjas, et „keele automaatanalüüsi seisukohalt polegi see alati triviaalne ülesanne, näiteks ei saa eeldada, et punkt sõna lõpus tähistab alati lauselõppu (see võib olla ka nt kuupäeva ja aastaarvu lõpus, vanemates tekstides ka lühendite lõpus)“ [4]. Seejuures isegi ei mõelda lausete ja ortograafiliste lausete eristamisele, vaid sellele, et lauselõpumärgid võivad olla ka lause sees kirjavahemärgiks, st ei pruugi alati tähistada lause lõppu.

Kirjeldatud uue meedia keele erijoonte alusel teame, et sageli ei kasutata lause alguses suurtähte või lause lõpus lauselõpumärki või kasutatakse neid korduvalt, samuti pole selge, kas tegu on emotikoni või märgi korduvkasutamise juhtumiga. Seepärast on uue meedia keele tekstides lausete tuvastamine veelgi keerulisem ülesanne: valitsevad samad ohud, mis kirjakeele normile vastava teksti puhul, kuid lisanduvad uue meedia keele eripärast tulenevad raskused.

Uue meedia keele teksti näide[2]

> *Nojah, ... aga see kui ma ühe teise sõbrannaga peole läksikn siis ta ütles: Sa lähed temaga peole? ma ütlesin et jah, siis mingi 4 minta pärast: Ma tulen ka teiega sinna peole !!!! mA ei tahtud teda üldse sinna peole ja üütlesin sellele sõbrantsile kes mind sinna kutsus et ütle talle et sul ei lubata , palun. Sest see teine oleks raudselt hakkanud ainult temaga olema, ma tunnen teda juba . Siis ta ütleski et tal ei lubatud,ja pärast ta küsis emalt ka ja tegelt ta ema ei lubanudki.Aga enne seda kui ta oma emalt päriselt küsis ütles see sõber mulle: Mhh, miks sa pead nii tegema ! ma ei vastanud talle. Siis ta vell ütles et : Vaepeal ma isegi imestan et sa minust vanem oled! Ma olen temast 9 päeva vanem , nt omg, milline ta ise on .Kui ts seda kuulis et ta ei saa ise kaasa tulla siis ütles ta mlle: Ma Nutan ! :'(*

Kuna uue meedia keele teksti laused ei ole sageli ortograafilised laused, siis omandab suurema tähtsuse lause kui terviklikku sõnumit väljendav üksus. Siinses töös on võetud kasutusele termin „süntaktiline lause“, mis tähistabki lauset kui üksust, mis väljendab terviklikku sõnumit. Süntaktiline lause võib kokku langeda ortograafilise lausega, kuid ei pruugi. Süntaktilisest lausest tuleb täpsemalt juttu järgmises peatükis annoteeritud korpuse teema juures.

[2] Näide pärineb siinse uurimistöö aluseks olevast korpusest, millest kirjutatakse pikemalt järgmises peatükis. Tekstid on kättesaadavad näiteks UD Estonian EWT failidest: https://universaldependencies.org/treebanks/et_ewt/index.html.

2. Annoteeritud korpus

Siinse uurimistöö põhieesmärk on võrrelda olemasolevate automaatsete tekstitöötlusvahendite lausestamise tulemusi uue meedia keele korpusel. Selleks on vaja uue meedia keele tekste ehk korpust, mis lausestatakse automaatsete tekstitöötlusvahenditega ja mille tulemust võrreldakse manuaalselt annoteeritud tekstidega (st tekstidega, kus lausepiirid on märgendatud manuaalselt).

2.1. Korpuse taust

Uurimistöö alus on 522 tekstilõigust koosnev korpus, mis pärineb Eesti veebikorpusest 2013[3] ehk etTenTen13 korpusest [21]. Selle koostas 2013. aastal Eesti Keele Instituut koostöös ettevõttega Lexical Computing Ltd [22].

etTenTen[4] sisaldab internetist veebirobotite abil alla laaditud veebilehtede sisu. Korpusse võeti 270 miljonit sõna 686 000 veebilehelt, välja jäeti olemasolevate lehtede koopiad ja lehed, mis on esindatud eesti kirjakeele koondkorpuses[5] [23]. etTenTeni koostamise protsessi on täpsemalt kirjeldanud Jelena Kallas jt [24], andes muu hulgas ülevaate korpuse tekstitüüpidest. Kallase jt arvutuste järgi on sellest korpusest 29% ajakirjandustekstid, 23% foorumite ja blogide tekstid, 9% teabetekstid, 4% usulise ja poliitilise sisuga tekstid ning 35% liigitamata tekstid. Siinse uurimuse aluseks olev korpus sisaldab foorumi-, blogi-, usulise sisuga ja liigitamata tekstitüübi tekste.

etTenTeni korpuse veebilehel on kirjas, et kogutud tekste on automaatselt muudetud. Automaatselt äratuntavad emotikonid ja laused, mis koosnesid ainult kirjavahemärgist, lühendist, kardinaalarvust, järgarvust ja pärisnimest, eemaldati [23]. Uurimuse aluseks oleva korpuse tekste ei ole automaatselt muudetud. Uurimuse jaoks on oluline, et tekstides oleksid säilinud uue meedia keele erijooned.

2.2. Korpuse annotatsioonid

Annoteeritud korpus oli enne siinse uurimistöö alustamist jagatud kaheks osaks: osa A koosnes 261 lõigust ja osa B samuti 261 lõigust. Kumbki osa oli antud annoteerimiseks viiele filoloogiatudengile. Osade annoteerijad ei kattunud. Osa A oli annoteeritud kujul tagasi andnud kolm annoteerijat ja osa B viis annoteerijat.

[3] https://metashare.ut.ee/repository/browse/veebikorpus13-ettenten-toortekst/b564ca760de111e6a6e4005056b4002419cacec839ad4b7a93c3f7c45a97c55f/.
[4] https://www.keeleveeb.ee/dict/corpus/ettenten/about.html.
[5] http://www.cl.ut.ee/korpused/segakorpus/.

Annoteerimise käigus olid annoteerijad lausepiiri tähistamiseks lisanud annotatsiooni, mis sisaldas järgmisi märgendeid:

1) „O“ – ortograafiline e õigekirjareeglitele vastav lause, mis lõpeb lauselõpumärgiga (need võivad olla mitmekordsed, näiteks mitu hüüumärki, küsimärki või punkti) ja/või järgmine lause algab suure algustähega;
2) „S“ – süntaktiline lause e lause, mis ei pea vastama õigekirjareeglitele, kuid moodustab mõttelise terviku;
3) „P“ – lõigu lõpuga tekkiv lausepiir.

Märgendid ei olnud üksteist välistavad, st vastavalt vajadusele oli annoteerija kasutanud ühes kohas ühte või mitut märgendit („S“, „O“, „P“, „SO“, „SP“, „OP“, „SOP“). Igale märgendikombinatsioonile oli algusse ja lõppu lisatud „#“ (nt #SOP#). Märgendikombinatsiooni nimetatakse siinses töös annotatsiooniks. Märgendite järjekord annotatsioonis ei olnud oluline.

Annoteerijal ei olnud lubatud lõigu tervikut muuta, v.a lisada sobiv annotatsioon, mis koosnes märgenditest ja „#“-dest. Annoteerija pidi annotatsiooni lisama sõna või kirjavahemärgi juurde ilma vaba täheruumita.

3. Korpuse eeltöötlemine

Uurimuse jaoks oli vaja koostada korpuse versioonid (e koondversioonid), kus olid ühte teksti koondatud kõikide annoteerijate annoteeringud. Seejärel oli vaja korpus sõnestada, et viia see CoNLL-U formaati. Sõnestamise käigus ilmnesid EstNLTK sõnestaja probleemsed kohad (vt punkt 3.3).

3.1. Annotatsioonide kontroll

Enne korpuse osadest A ja B annoteeritud koondversioonide tegemist kontrolliti manuaalselt ja automaatselt annoteerijate annoteerimisjuhistest kinnipidamist. Selgus, et osa annoteerijaid oli mõnel juhul juhiseid eiranud.

Üks annoteerija oli lõikude vahele läbivalt lisanud kaks tühja rida, need eemaldati automaatselt. Kaks annoteerijat ei olnud lõigu lõppe märgendanud „P-ga“ ja mõnes failis puudus üksikute lõikude lõpus „P“. Et lõigu lõpp on selgelt eristatav tunnus, lisati puuduvad „P-d“ automaatselt. Leidus üksikuid kohti, kus märgend „P“ oli kasutusel lõigu sees. Seal märgend asendati või eemaldati, sõltuvalt olukorrast.

Osas B oli üks annoteerija jätnud lõpuni märgendama (kas täiesti või osaliselt) neli lõiku ja üks annoteerija ühe lõigu. Kuna seda osa annoteeris viis annoteerijat, oli annoteerimata lõikude mõju koondversioonile väga väike. Need lõigud jäeti annoteerimata, kuid lõikude lõppu lisati „P“.

Annoteerijad olid mitmes kohas pannud lausepiiri enne kirjavahemärki, kuigi see tuli ilma tühikuta panna pärast kirjavahemärki. Sellised kohad tuvastati automaatselt ja parandati manuaalselt, kuna automaatne parandamine ei olnud emotikonide jms pärast võimalik. Üks tüüpiline kirjavahemärki puudutav eksimus oli seotud mõttekriipsuga, nimelt oli annotatsioon pandud selle ette. Lausepiirid tõsteti sellistes kohtades ümber ja mõttekriips jäi lõppeva lause viimaseks elemendiks.

Et kõigi annoteeritud tekstide lõigud vaadati üle paralleelselt algtekstiga, siis tulid välja väikseimadki kõrvalekalded. Esines kohti, kus mõni annoteerija oli tühiku kustutanud või lisanud. Ühes kohas oli annoteerija kirjutanud märgendiga üle lauselõpumärgi. Nendes kohtades taastati algne olukord.

Mõnes kohas oli annoteerija kasutanud märgendina väiketähte. Kõikide annoteeritud failide kõik märgendid suurtähestati, et vältida probleemide teket.

Pärast annoteeritud tekstide automaatset ja manuaalset korrigeerimist oli nende alusel võimalik koostada koondversioonid, mida saab kasutada automaatsete lausestajate tulemuste võrdlemiseks erinevatel juhtudel.

Nii osa A kui ka osa B tekstist koostati neli koondversiooni:

1) kõigi süntaktilise ja ortograafilise lause piiri annotatsioonidega korpus, millest tehti omakorda kaks versiooni:
   a) esimeses on kõik annotatsioonid algkujul koos esinemisarvuga selles kohas (nt *Olen ka oma lähedastega sellest rääkinud,#S2# ise oleksin rahul kui tuhastatakse,#S1# ja laul mida võiks sel hetkel lasta vms Eric Clapton : Tears in heaven:)#SP2##SOP1#*) ja
   b) teises on iga märgendi juures selle korduste arv selles asukohas (annoteerijate märgendid samas asukohas on n-ö liidetud, nt *Olen ka oma lähedastega sellest rääkinud,#S2# ise oleksin rahul kui tuhastatakse,#S1# ja laul mida võiks sel hetkel lasta vms Eric Clapton : Tears in heaven:)#S3P3O1#*);
2) enim märgitud süntaktilise ja ortograafilise lause piiri annotatsioonidega korpus, milles on alles jäetud ainult enim „hääli“ saanud märgendid, st osa A puhul oli vähemalt kaks annoteerijat lisanud sellesse kohta sama märgendi ja osa B puhul oli vähemalt kolm annoteerijat lisanud sellesse kohta sama märgendi (nt *Olen ka oma lähedastega sellest rääkinud,#S2# ise oleksin rahul kui tuhastatakse, ja laul mida võiks sel hetkel lasta vms Eric Clapton : Tears in heaven:)#S3P3#*);
3) kõigi ortograafilise lause piiri annotatsioonidega korpus (välja on jäetud süntaktilise lause piiri tähistavad märgendid (nt *Olen ka oma lähedastega sellest rääkinud, ise oleksin rahul kui tuhastatakse, ja laul mida võiks sel hetkel lasta vms Eric Clapton : Tears in heaven:)#P2OP1#*);
4) enim märgitud ortograafilise lause piiri annotatsioonidega korpus (nt *Olen ka oma lähedastega sellest rääkinud, ise oleksin rahul kui tuhastatakse, ja laul mida võiks sel hetkel lasta vms Eric Clapton : Tears in heaven:)#P3#*).

Koondversioonide koostamiseks kasutati programmeerimiskeelt Python Jupyter Notebookis ja koostati vajalikud funktsioonid (vt näide lisas C1).

3.2. Sõnestuse kontroll

Uurimistöö seisukohalt on esmatähtis kõrvutada automaatseid lausetajaid, kuid selle tegevusega kaasneb paratamatult sõnestamisvajadus. See andis võimaluse võrrelda ka automaatsete tekstitöötlusvahendite sõnestajate tulemusi.

Eri automaatsete tekstitöötlusvahendite sõnestajad töötavad erinevalt, sest sõnestamise funktsioon on teostatud erinevalt. Eesti universaalsete sõltuvustega märgendatud puudepanga juures on sõnestamise kohta kirjas, et sõnana käsitatakse tühikute ja/või kirjavahemärkidega eraldatud stringe [25]. Siinses töös kasutati korpuse manuaalse sõnestamise ajakulu vähendamiseks EstNLTK sõnestajat, mis töötab reeglipõhiselt, kasutades teksti osadeks jagamisel regulaaravaldisi ning pannes seejärel vajaduse korral järjestikused osad uuesti kokku [26], luues tähendusega terviklikke sümbolijadasid. Tulemuseks on siiski manuaalselt sõnestatud tekstid, sest automaatse sõnestaja tööd korrigeeriti manuaalselt ühtsete reeglite alusel.

Sõnestamisel ei sekkutud tekstide õigekirja, sh ei parandatud kokku- ja lahkukirjutusvigu. Uue meedia keele tekstides leidub olukordi, kus tühikuid ei kasutata piisavalt, näiteks võib tekst olla täiesti ilma tühikuteta. Eelkõige parandati sümbolite ja kirjavahemärkidega seotud sõnestust, esmajoones siis, kui mingi märgikombinatsioon moodustas tervikliku tähenduse.

EstNLTK sõnestamise tulemuste manuaalsel ülevaatamisel tulid esile probleemsed kohad, mida saaks EstNLTK arendamisel ja reeglite väljatöötamisel arvestada. Järgnevalt on esitatud manuaalselt parandamist vajanud sõnestamisjuhud (vt ka lisa A, kus on rohkem näiteid):

1) loetelu tähistavad loendimärgid on lahku sõnestatud, kuigi on tuvastatav, et tegu on loeteluga (nt EstNLTK: *Minu meelest on õige järjekord : 1) Kise 2) Toy 3) Giku 4) Tzan 5) Kak*; manuaalne: *Minu meelest on õige järjekord : 1) Kise 2) Toy 3) Giku 4) Tzan 5) Kak*);
2) loendimärgina on kasutatud märki „-“ ja see on järgmisest sõnest lahku sõnestamata (nt EstNLTK: *KUI SA ... -leiad , et on ikka veel võimatu kontrollida ärevust ! Ja üleüldse igapäevast stressi ... -tead , et võibolla on sul üks hea nädal , millele aga kohe järgneb uus ja HALB ! -oled hirmul sõites autoga , kartes , et sind tabab järjekordne HOOG !*;
   manuaalne: *KUI SA ... - leiad , et on ikka veel võimatu kontrollida ärevust ! Ja üleüldse igapäevast stressi ... - tead , et võibolla on sul üks hea nädal , millele aga kohe järgneb uus ja HALB ! - oled hirmul sõites autoga , kartes , et sind tabab järjekordne HOOG !*);
3) arvu kirjutades on tuhandike eraldamiseks kasutatud tühikut, kuid seetõttu on arvu teine pool lahku sõnestatud (nt EstNLTK: *mees oma ligi 30 000 palga juures*; manuaalne: *mees oma ligi 30000 palga juures*);
4) protsendimärk ei ole arvust eraldatud (nt EstNLTK: *Kui aga 60% neist*; manuaalne: *Kui aga 60 % neist*);
5) korduvad kirjavahemärgid on sõnestatud eraldi (sellised probleemid olid sageduse poolest esikohal, nt EstNLTK: *rõhutan võibolla . . .*; manuaalne*: rõhutan võibolla ...*);

6) enamiku emotikonidest tundis sõnestaja ära, kuid juhtus ka seda, et emotikoni tervik oli lõhutud (nt EstNLTK: *Äkki puberteet ? : d*; manuaalne: *Äkki puberteet ? :d*);
7) kirjavahemärk ei ole jutumärgist eraldatud (siin ja mõnes teiseski kohas tuleb arvestada, et uue meedia keeles ei kasutata ainult neid märgiversioone, mida standardi järgi kasutatakse eesti keeles märkidena, nt EstNLTK: *« See sõna on ränk , kes suudab seda kuulda ?»*; manuaalne*: « See sõna on ränk , kes suudab seda kuulda ? »*);
8) ülakomade vahel olev tekst ei ole eraldi sõnestatud (nt EstNLTK: *nendest'soovitustest'*; manuaalne: *nendest ' soovitustest '*);
9) veebilehe aadress on eraldi sõnestatud (nt EstNLTK: *tehnokratt . net / hyphenator / mergeAndPack . html*; manuaalne: *tehnokratt.net/hyphenator/mergeAndPack.html*);
10) failinimed on laiendist lahku sõnestatud (nt EstNLTK: *functions . php*; manuaalne: *functions.php*);
11) domeeni riigitähis on punktist lahku sõnestatud (nt EstNLTK: *. ru*; manuaalne: *.ru*);
12) käändelõpp on eraldi sõnestatud (nt EstNLTK: *Quality guidelines ' i*; manuaalne: *Quality guidelines'i*);
13) kellaajas on tunnid ja minutid eraldi sõnestatud (nt EstNLTK*: kell 8 : 25*; manuaalne: *kell 8:25*);
14) suhtarv või seis on lahku sõnestatud (nt EstNLTK: *1 : 1 suuruses*; manuaalne: *1:1 suuruses*);
15) lühendist on punkt lahku sõnestatud (uue meedia keeles ei järgita õigekirjareegleid, kuid seda võib sageli juhtuda ka standardsetes tekstides, nt EstNLTK: *jms .*; manuaalne: *jms.*).

Kuigi mõnel nimetatud juhul (nt 3, 12, 13, 15) oleks EstNLTK dokumentatsiooni järgi pidanud sõnestamine toimuma korrektselt, see siiski nii ei olnud [27]. Esitatud näidete alusel on soovi korral võimalik EstNLTK sõnestaja tulemusi edasi uurida ja sõnestajat täiendada.

3.3. Annoteeritud teksti CoNLL-U formaati viimine

Annoteeritud korpused viidi edasise töö hõlbustamiseks CoNLL-U[6] formaati, mis on loomuliku keele uuringutes laialt kasutatud ja standardne formaat.

[6] Association for Computational Linguisticsil (Arvutilingvistika Ühing) on loomuliku keele töötlemise valdkonnas rühmitus Special Interest Group on Natural Language Learning (SIGNLL) (häälдus [signal]), http://www.signll.org/. Rühmitus korraldab igal aastal aastakonverentsi (ingl Conference on Natural Language Learning, CoNLL). Alates 1999. aastast on aastakonverentsi ühe osana korraldatud jagatud treenimis- ja testandmestikuga ühise ülesande lahendamist [47]. Kümnendal aastakonverentsil (CoNLL-X) kasutati ühise ülesande tulemuste hindamiseks ühtset sõltuvusvormingut [48]. Seda formaati tähistatakse CoNLL-X formaadina. CoNLL-U formaat on CoNLL-X formaadi täiendatud versioon [28].

CoNLL-U formaadis fail on UTF-8 kodeeringus lihtteksti fail, kus ainuke kasutuses olev reavahetuse märk on „\n“ (ingl *LF character*). Selles failis on kolme tüüpi ridu [28]:

1) sõnadega read, kus sõna kohta on kümme erinevat märgendit, mis on eraldatud „\t“-ga;
2) tühjad read, mis tähistavad lausepiire;
3) kommentaaride read, mis algavad „#“-ga.

Iga sõna kohta on real järgmised andmed [28]:

1) ID – sõna indeks, mis algab arvust 1 iga uue lause korral;
2) FORM – sõna või kirjavahemärk;
3) LEMMA – lemma või tüvi;
4) UPOS – universaalne sõnatüüp;
5) XPOS – keelespetsiifiline sõnatüüp (kui puudub, siis alakriips);
6) FEATS – morfoloogiliste tunnuste loetelu (kui puudub, siis alakriips);
7) HEAD – vastava sõna süntaktiline ülem (kas ID väärtus või null (0));
8) DEPREL – süntaktilise sõltuvuse seos ülemaga;
9) DEPS – täiustatud sõltuvusgraaf HEAD-DEPREL-paaride listina;
10) MISC – muud märkused.

Sõna juures olevad infoväljad ei tohi olla tühjad. Ükski väli peale väljade FORM, LEMMA ja MISC ei tohi sisaldada ka tühikut. Alakriipsu („_“) tohib kasutada määratlemata väärtuse tähistamiseks, välja arvatud väljal ID [28].

CoNLL-U formaadist parema ülevaate saamiseks on joonisel 1 toodud kahe ingliskeelse lause alusel kogu andmestikuga CoNLL-U formaadi näide [28].

```
# sent_id = 1
# text = They buy and sell books.
1   They     they    PRON    PRP    Case=Nom|Number=Plur              2   nsubj   2:nsubj|4:nsubj   _
2   buy      buy     VERB    VBP    Number=Plur|Person=3|Tense=Pres   0   root    0:root            _
3   and      and     CONJ    CC     _                                 4   cc      4:cc              _
4   sell     sell    VERB    VBP    Number=Plur|Person=3|Tense=Pres   2   conj    0:root|2:conj     _
5   books    book    NOUN    NNS    Number=Plur                       2   obj     2:obj|4:obj       SpaceAfter=No
6   .        .       PUNCT   .      _                                 2   punct   2:punct           _

# sent_id = 2
# text = I have no clue.
1   I       I       PRON    PRP   Case=Nom|Number=Sing|Person=1     2   nsubj   _   _
2   have    have    VERB    VBP   Number=Sing|Person=1|Tense=Pres   0   root    _   _
3   no      no      DET     DT    PronType=Neg                      4   det     _   _
4   clue    clue    NOUN    NN    Number=Sing                       2   obj     _   SpaceAfter=No
5   .       .       PUNCT   .     _                                 2   punct   _   _
```

Joonis 1. CoNLL-U ingliskeelne näide erinevate andmeväljadega

Automaatsete lausestajate tulemuste hindamiseks ei ole vaja teada sõnade kõiki sõltuvusi. Siinse uurimistöö jaoks piisab vähendatud andmetega CoNLL-U formaadis tekstidest, kuid hindamisskripti kasutamiseks peavad kõik kümme välja säilima. On piisav, kui andmetega on täidetud väljad ID ja FORM, sest teised väljad (v.a HEAD) saab tähistada alakriipsuga. Kuna tulemuste hindamiseks kasutatav skript nõuab väljal HEAD arvulist väärtust, märgiti indeksiga 1 sõna väärtuseks vaikimisi 0 ja lause ülejäänud sõnade puhul 1.

Korpuse CoNLL-U formaati viimiseks kasutati programmeerimiskeeles Python kirjutatud funktsiooni (vt lisa C2), millega viidi CoNLL-U formaati kõik osa A ja osa B koondversioonid, et hiljem oleks võimalik automaatsete lausestajate tulemusi võrrelda korpuse eri versioonidega. Konverteerimise tulemuseks on joonisel 2 esitatud vähendatud andmetega CoNLL-U formaadis tekst.

```
# newpar id = 149414-p1
# newpar_text = Tänapäeva soolise võrdõiguslikkuse taustal on ikkagi õige, et naine maksab oma tarbitu ise. Teeb endale ukse lahti ise ja aitab endale palitu selga ise. Ega naine pole ju mingi alaarenenud nõrguke, naine saab ka ise kõigega hakkama. Soolise võrdõiguslikkuse nime all loodavad palju naised jätta alles varasemad klassikalised hüved (mees maksab) kui ka uuema aja võrdõiguslikkuse hüved (naistele meestega võrdne palk). See muidugi ei ole päris aus suhtumine. Kas kõik vanamoodi või kõik uutmoodi, mõlemast parimaid palasid pole ilus tahta.
# sent_id = 149414-p1s1
# text = Tänapäeva soolise võrdõiguslikkuse taustal on ikkagi õige, et naine maksab oma tarbitu ise.
# label = #SO3#
1     Tänapäeva     _     _     _     _     0     _     _     _
2     soolise       _     _     _     _     1     _     _     _
3     võrdõiguslikkuse    _     _     _     _     1     _     _     _
4     taustal       _     _     _     _     1     _     _     _
5     on      _     _     _     _     1     _     _     _
6     ikkagi  _     _     _     _     1     _     _     _
7     õige    _     _     _     _     1     _     _     _
8     ,       _     _     _     _     1     _     _     _
9     et      _     _     _     _     1     _     _     _
10    naine   _     _     _     _     1     _     _     _
11    maksab  _     _     _     _     1     _     _     _
12    oma     _     _     _     _     1     _     _     _
13    tarbitu       _     _     _     _     1     _     _     _
14    ise     _     _     _     _     1     _     _     _
15    .       _     _     _     _     1     _     _     _

# sent_id = 149414-p1s2
# text =  Teeb endale ukse lahti ise ja aitab endale palitu selga ise.
# label = #SO3#
1     Teeb    _     _     _     _     0     _     _     _
2     endale  _     _     _     _     1     _     _     _
3     ukse    _     _     _     _     1     _     _     _
4     lahti   _     _     _     _     1     _     _     _
5     ise     _     _     _     _     1     _     _     _
6     ja      _     _     _     _     1     _     _     _
7     aitab   _     _     _     _     1     _     _     _
8     endale  _     _     _     _     1     _     _     _
9     palitu  _     _     _     _     1     _     _     _
10    selga   _     _     _     _     1     _     _     _
11    ise     _     _     _     _     1     _     _     _
12    .       _     _     _     _     1     _     _     _
…
```

Joonis 2. Osa annoteeritud tekstist vähendatud andmetega CoNLL-U formaadis

Enne automaatsete tekstitöötlusvahendite testimist ja tulemuste hindamist oli vaja hinnata korpuse annoteerimise kvaliteeti.

4. Annoteerijate ühtsus

Annoteerijatel tuli lausepiire märgendada oma parima äranägemise järgi. Nad võisid tuvastada lausepiire eri kohtades, esmajoones võis see nii olla süntaktilise lause puhul. Kui annoteerijate hinnangud oleksid väga erinevad, ei saaks annoteeritud korpust automaatsete lausestajate hindamiseks kasutada, sest lausepiiride tegelik asukoht ei ole selge. Seepärast on vaja hinnata, mil määral langevad eri annoteerijate annoteeritud laused kokku.

Annoteerijate hinnangute ühtsuse (ingl *inter-annotator* või *inter-rater agreement või reliability*) hindamiseks on mitu meetodit, millest siinses uurimuses on kasutatud kahte mõõdikut: Dice'i koefitsienti ja Fleissi kappat.

4.1. Dice'i koefitsient

Dice'i koefitsient (ingl *Dice coefficient, Sørensen–Dice coefficient*, *Dice similarity index* või ka *Dice similarity coefficient*, DSC) on statistik, millega mõõdetakse kahe hulga X ja Y sarnasust valemiga [29]

$$Dice(X,Y) = \frac{2|X \cap Y|}{|X| + |Y|}.$$

Dice'i koefitsiendi leidmiseks tuleb leida kahes annoteeritud failis annoteeringute ühisosa, st loendada kohad, kus kaks annoteerijat on tähistanud lausepiiri ühes ja samas kohas, ning loendada mõlema faili kõik annoteeringud.

Osa A annoteeris kolm ja osa B annoteeris viis annoteerijat. Seepärast tuli paari kaupa arvutada Dice'i koefitsient osa A puhul kolmele ($C_3^2 = 3$) kombinatsioonile ja osa B puhul kümnele ($C_5^2 = 10$) kombinatsioonile, ning leida kummagi osa kohta keskmine Dice'i koefitsient.

Dice'i koefitsiendi arvutamiseks kasutati programmeerimiskeeles Python koostatud funktsiooni (vt lisa C3), mille abil arvutati Dice'i koefitsient neljal erineval juhul kummagi osa kohta:

1) lausepiiride kattuvus – vaadati ainult seda, kas annoteerijad asetasid lausepiiri samasse kohta, sõltumata sellest, milline oli konkreetne annotatsioon;
2) süntaktiliste lausete kattuvus – vaadati, kas annoteerijad asetasid süntaktilise lause piiri samasse kohta;
3) ortograafiliste lausete kattuvus – vaadati, kas annoteerijad asetasid ortograafilise lause piiri samasse kohta;

4) lauseliikide kattuvus – vaadati, kas sõna järel on täpselt sama annotatsioon („S“, „O“ või „SO“).

Lõigupiiri tähistavat lauselõpumärgendit järgnevates analüüsides ei kasutatud, sest kõik lõigud kõikides annoteeritud tekstides lõppesid sellega ja selle esinemine või mitteesinemine ei sisalda olulist infot.

Tabel 1. Dice'i koefitsiendid neljal eri juhul kahe eri osa kohta

Kattuvuse kirjeldus	Osa A	Osa B
Lausepiiride kattuvus	0,93	0,92
Süntaktiliste lausete kattuvus	0,90	0,90
Ortograafiliste lausete kattuvus	0,95	0,96
Lauseliikide kattuvus	0,83	0,84

Tabelis 1 esitatud Dice'i koefitsientide põhjal saame järeldada, et mõlema osa puhul tuvastasid annoteerijad lausepiire üsna sarnaselt: nad olid paigutanud 92–93% lausepiiridest samadesse kohtadesse. Väiksem on annoteerijate ühtsus süntaktilise lause annoteerimisel, kuid ka see on mõlemal juhul 90% juures. Ortograafilise lause tuvastasid annoteerijad samades kohtades 95–96%-l juhtudest. See on ootuspärane, kuna see lause on väliste tunnuste alusel paremini tuvastatav ja ei eeldanud annoteerijalt lause mõttelise terviku tajumist. Kõige väiksem oli annoteerijate ühtsus lauseliikide kattuvuses (tekstis samas kohas täpselt sama liiki lausepiiri annotatsiooni kasutamises), kuid ka see näitaja on siiski suur.

4.2. Fleissi kappa

Annoteerijate hinnangute ühtsuse määra hindamiseks kasutatakse n-ö kappasid [30]. Üks selline levinud statistik on Coheni kappa. Siinse uurimistöö jaoks see kappa ei sobinud, kuna selle abil saab hinnata kahe annoteerija annoteeringute ühtsust, kuid osa A annoteeris kolm ja osa B annoteeris viis annoteerijat. Seepärast kasutati Coheni kappa täiendatud versiooni Fleissi kappat, mis sobib rohkem kui kahe annoteerija ühtsuse hindamiseks.

Fleissi kappa näitab, kui palju on annoteerijate hinnangute ühtsus parem juhuslike annoteeringute ühtsusest [30]. Fleissi kappa arvutati samuti nelja juhu kohta, mida kirjeldati Dice'i koefitsiendi juures.

Fleissi kappat on keerulisem arvutada kui Dice'i koefitsienti. Erinevalt Dice'i koefitsiendi arvutamisest ei pea sel puhul kõiki annoteeritud tekste omavahel võrdlema, vaid piisab koondversioonist, kust saab teada, mitu annoteerijat millist annotatsiooni millise sõna järel kasutas.

Fleissi kappa üldine valem on järgmine [31]:

$$\kappa = \frac{\bar{P} - \bar{P}_e}{1 - \bar{P}_e}.$$

Tähistame järgmiselt [31]:

N on kõikvõimalike elementide arv (siinses uurimuses saab annotatsioon teoreetiliselt olla iga sõna lõpus, seega käesoleval juhul *N* = sõnade arv),

n on võimalik annotatsioonide arv elemendi juures (käesoleval juhul *n* = annoteerijate arv),

k on kasutatud kategooriate arv (siinses uurimuses on esimesel juhul *k* = 2 (kas sõnale järgneb annotatsioon või mitte), teisel juhul on *k* = 2 (kas sõna järel on süntaktilise lause piir või mitte), kolmandal juhul on *k* = 2 (kas sõna järel on ortograafilise lause piir või mitte) ja neljandal juhul on *k* = 4 (kas sõna järel on sama annotatsioon („S“, „O“ või „SO“) või annotatsiooni pole).

Elementide indeksid on $i = 1 \ldots N$, kategooriate indeksid on $j = 1 \ldots k$. n_{ij} on annoteerijate arv, kes annoteeris *i*-nda elemendi *j*-nda kategooriaga [31].

Kõigepealt tuleb arvutada p_j ehk kõikide kategooriate esinemise sagedus [31]:

$$p_j = \frac{1}{nN} \sum_{i=1}^{N} n_{ij} .$$

Seejärel tuleb arvutada iga elemendi kohta annoteerijate hinnangute kokkulangevus P_i [31]:

$$P_i = \frac{1}{n(n-1)} \sum_{j=1}^{k} n_{ij}(n_{ij} - 1) .$$

Kui iga elemendi kohta on P_i arvutatud, saab arvutada keskmise P_i [31]:

$$\bar{P} = \frac{1}{N} \sum_{i=1}^{N} P_i .$$

Kappa arvutamiseks on vaja arvutada ka $\bar{P}_e$, mis arvutatakse valemiga [31]:

$$\bar{P}_e = \sum_{j=1}^{k} p_j^2 .$$

Näide arvutuskäigu kohta (vt tabel 2), kui $N = 10$, $n = 3$ (annoteerijaid on 3) ja $k = 2$:

Tabel 2. Fleissi kappa andmete näide

Element n_{ij}	Kategooria 1	Kategooria 2	P_i
Sõne 1	3	0	1
Sõne 2	3	0	1
Sõne 3	2	1	0,33
Sõne 4	1	2	0,33
Sõne 5	3	0	1
Sõne 6	3	0	1
Sõne 7	0	3	1
Sõne 8	3	0	1
Sõne 9	3	0	1
Sõne 10	1	2	0,33
Kokku	22	8	7,99
p_j	0,73	0,27	

Tabeli 2 andmete alusel saame Fleissi kappa arvutada järgmiselt:

$\bar{P} = \frac{7,99}{10} = 0,799,$

$\bar{P_e} = 0,73^2 + 0,27^2 = 0,6058,$

$\kappa = \frac{0,799-0,6058}{1-0,6058} = 0,49.$

Kuna Fleissi kappat uuritud teekides implementeeritud ei olnud, siis kirjutati siinse uurimistöö jaoks programmeerimiskeele Python kolm eri funktsiooni Fleissi kappa arvutamiseks. Lisas C4 on toodud Fleissi kappa arvutamise koodi näide.

Tabelis 3 on esitatud Fleissi kappa suurused neljal kirjeldatud juhul kummagi osa kohta.

Tabel 3. Fleissi kappad neljal eri juhul kahe eri osa kohta

Juhtumi kirjeldus	Osa A	Osa B
Lausepiiride kattuvus	0,92	0,91
Süntaktiliste lausete kattuvus	0,90	0,89
Ortograafiliste lausete kattuvus	0,94	0,96
Lauseliikide kattuvus	0,87	0,87

Landis ja Koch on andnud oma uuringus kappa väärtustele järgmise tõlgenduse (vt tabel 4) [32].

Tabel 4. Fleissi kappa tõlgendused [32]

κ	Tõlgendus
< 0	Annoteerijate halb ühtsus
0,01–0,20	Annoteerijate vähene ühtsus
0,21–0,40	Annoteerijate rahuldav ühtsus
0,41–0,60	Annoteerijate mõõdukas ühtsus
0,61–0,80	Annoteerijate oluline ühtsus
0,81–1,00	Annoteerijate peaaegu perfektne ühtsus

Kui vaadata tabelis 3 esitatud kappasid ja arvestada tabelis 4 toodud tõlgendusi, siis võime järeldada, et annoteerijate ühtsus mõlema osa puhul on olnud sõltumata vaadeldud juhtumist perfektne. Peame seejuures arvestama, et eri mõõdikutel on oma iseärasused. Fleissi kappa on seda suurem, mida vähem on kategooriaid. Siinsetes arvutustes oli kategooriaid kolmel juhul kaks ja viimasel juhul neli. Seega saab kappasid omavahel võrrelda, sest kategooriate arvu erinevused ei ole suured.

Nagu Dice'i koefitsient nii näitab ka Fleissi kappa, et süntaktilise lause piiride määramisel olid annoteerijad vähem ühel meelel kui ortograafilise lause piiride puhul. Tähelepanu tuleks pöörata 4. kappale lauseliikide kattuvuse kohta, mis on ka suur, kuigi vaadati konkreetsete annotatsioonide kattuvust. See on küll teistest kehvem, mis on loogiline, sest annoteerijad pidid ühtsuse saavutamiseks kasutama täpselt sama annotatsiooni (kolmest võimalikust), kuid jääb siiski peaaegu perfektse ühtsuse piiresse.

Kokkuvõttes saab sedastada, et korpus oli annoteeritud suure ühtsusega ja sobib tulemuste hindamise aluseks.

5. Automaatsete lausestajate ja sõnestajate võrdlus

Automaatsete lausestajate ja sõnestajate hindamise eesmärk on teada saada, kui hästi need töötavad uue meedia keeles oleva tekstiga. Tulemuste alusel saab teada, kas uue meedia keele jaoks tuleb treenida uued mudelid või saab kasutada olemasolevaid. Samuti saab teavet selle kohta, milliseid tekstitöötlus-vahendeid tasub eelistada.

5.1. Võrreldavad tekstitöötlusvahendid

Automaatsed tekstitöötlusvahendid, mille lausestajate ja sõnestajate tulemusi annoteeritud korpuse alusel siinses töös hinnatakse, on EstNLTK [7], UDPipe [8] ja StanfordNLP [9]. Need kolm valiti uurimusse põhjusel, et neid on suhteliselt lihtne rakendustes kasutada ja seega võib eeldada, et huvi nende kasutamise vastu on suurem.

EstNLTK sõnestaja on reeglipõhine ja lausestaja on mudelipõhine reeglipõhise järelkontrolliga. UDPipe'il ja StanfordNLP-l on sõnestamiseks ja lausestamiseks eeltreenitud mudelid Eesti universaal-sete sõltuvustega märgendatud puudepangal[7] (ingl *Estonian Universal Dependencies' Treebank,* EDT), mis kuulub Universal Dependenciese[8] raamistikku. EDTs on 30 972 lauset ja 437 769 sõnet [33].

5.1.1. EstNLTK

EstNLTK on Pythoni teekide eesti keele töötlemise kogumik [7], mida arendab Tartu Ülikooli arvuti-teaduse instituut. Üks loomuliku keele töötluse põhiteek on EstNLTK teek, milles on ka teksti sõnesta-mise ja lausestamise vahendid. Uurimistöös on põhiliselt kasutatud EstNLTK versiooni 1.6.5beta.

EstNLTK kõige tähtsam klass on *Text* [34], mille kaudu saab kasutada kõiki EstNLTK funktsioone, näiteks teksti tükeldamist. Teksti tükeldamisel on EstNLTK loodav väikseim ühik sõne (ingl *token*), mis on vaba täheruumi (ingl *whitespace*) vahel olev sümbolijada või kirjavahemärk. Väikseim tähendusega ühik on sõna, mis võib olla sama sõnega või koosneda mitmest sõnest. Sõned kombineeritakse sõnadeks normaliseerimise käigus, näiteks liidetakse punktiga eraldatud lühendid või e-posti aadressid. Sõnu tükeldab EstNLTKs TokensTagger, mis implementeerib NLTK[9] regulaaravaldise põhist tükeldajat

[7] https://universaldependencies.org/treebanks/et_edt/index.html.
[8] Universal Dependencies (UD) on raamistik grammatika (kõneosad, morfoloogilised tunnused ja süntaktilised sõltuvused) ühetaoliseks märgendamiseks inimkeeltes. Raamistikus teevad kaastööd 300 inimest rohkem kui 150 puudepanga töötlemisel 90 keeles [33]. Eesti keele kohta on UD-s kaks puudepanka (EDT ja EWT) [51].
[9] Loomuliku keele töötlemise vahendite pakett (https://www.nltk.org/).

WordPunctTokenizer[10] [35]. Seejärel liidab CompoundTokenTagger vajaduse korral eeldefineeritud reeglite järgi sõned, mille TokensTagger oli tükeldanud, uuesti [27].

Suuruselt järgmine ühik on lause, mis on sõnade list. See tähendab, et kõigepealt tükeldatakse tekst sõnadeks ja seejärel tuvastatakse lausepiirid, jälgides, et ükski piir ei satuks sõna keskele. Laused tuvastab EstNLTKs SentenceTokenizer, mis implementeerib NLTK PunktSentenceTokenizeri eesti keele jaoks loodud mudeli alusel [36].

5.1.2. UDPipe

Tšehhi Vabariigi Karli Ülikooli matemaatika-füüsika teaduskonna formaal- ja rakenduslingvistika instituudile kuuluv UDPipe on n-ö toru ehk konveier (ingl *pipeline*), mis hõlmab paljusid keeletöötlusvahendeid [8]. UDPipe'i saab kasutada morfoloogiliseks töötluseks (sealhulgas lemmatiseerimiseks) ja teksti süntaktiliseks analüüsiks. UDPipe'il on 61 keele jaoks eeltreenitud 94 mudelit, kuid mudeleid on võimalik ka ise treenida ja kasutada [37].

UDPipe'i[11] Universal Dependencies 2.5[12] versiooni alusel eesti keele jaoks eeltreenitud mudeli Estonian-EDT[13] sooritus (ingl *performance*) on veebilehe andmetel sõnestamisel 100,0% ja lausestamisel 91,6% [38]. Sooritust all peetakse silmas F-skoori [39].

UDPipe'is töötab teksti tükeldamine täielikult eeltreenitud mudelil [40]. Mudel treenitakse algteksti ja tükeldatud teksti võrdlemise käigus ilma keelespetsiifiliste eelteadmisteta. Mudel õpib, kus tükeldada sõna ja kus lause.

Tekstis tuvastatakse tükeldamise käigus sõne ja lause piirid samal ajal. Tükeldamise eesmärk on, et mudel suudaks jagada sisendtähemärgid sõnadeks ja lauseteks. Iga tähemärk liigitatakse ühte klassi kolmest: järgneb sõnepiir, järgneb lausepiir, piiri pole. Klassifitseerimisega tegelev mudel põhineb ühekihilisel kahesuunalisel väravatega rekurrentsel närvivõrgul, mis ennustab iga tähemärgi kohta tema klassi [39]. UDPipe'i viimases versioonis kasutatakse täiendatud tükeldamise funktsiooni, mis oskab teksti paigutuse järgi tuvastada lausepiiri lõigu lõpus ja teksti lõpus [33].

[10] https://www.nltk.org/api/nltk.tokenize.html.
[11] Uurimistöös kasutati versiooni 1.2.0.3.
[12] http://ufal.mff.cuni.cz/udpipe/models#universal_dependencies_25_models.
[13] https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-3131.

5.1.3. StanfordNLP

Leland Stanford Junior Universityle kuuluv StanfordNLP [41] on Pythoni töövahendite komplekt ehk analüüsimispakett, kus on 53 loomuliku keele jaoks treenitud mudelid 73 puudepanga alusel [9].

StanfordNLP[14] eesti keele mudel et_edt on treenitud Universal Dependenciese versiooni 2 Eesti universaalsete sõltuvustega märgendatud puudepangal (UD Estonian (EDT)). StanfordNLP veebilehe andmetel on et_edt mudeli F-skoor sõnastamisel 99,95% ja lausestamisel 93,82% [42].

Nagu UDPipe nii on ka StanfordNLP täielikult tehisnärvivõrgul põhinev toru (ingl *pipeline*) süsteem, mis võtab sisendina lihtteksti ja töötleb seda mitmel viisil, sh tuvastab laused ja sõned. Sõnede ja lausete piirid määratakse ühe ja sama protsessi raames, kus tekst vaadatakse läbi ühiku haaval. Enamiku keelte jaoks on ühikuks tähemärk. Iga tähemärk klassifitseeritakse kas sõne lõpuks (EOT), lause lõpuks (EOS), püsiühendiks (ingl *multi-word*, MWT), lauset lõpetavaks püsiühendiks (MWS) või klassi „Muu“. Klassifitseerimisega tegeleb mudel, mis põhineb kahesuunalisel väravatega rekurrentsel närvivõrgul (BiLSTM). Iga ühiku kohta ennustab mudel hierarhiliselt järgmist: esmalt seda, kas tähemärk on sõna lõpus, seejärel klassifitseerib sõna lõpu kaheks täpsemaks kategooriaks kahe iseseisva binaarse klassifitseerijaga, millest üks klassifitseerib ühiku lause lõpuks ja teine püsiühendiks.

Kuna lausepiiride ja püsiühendi määramine eeldab konteksti paremat teadmist, antakse sõne tasandi info kahekihilises kahesuunalises BiLSTMis kaasa. Esimene BiLSTMi kiht töötab otse algse tähemärgiga ja teeb kõiki kategooriaid arvestades tähemärgi kohta esialgse ennustuse. Teine kiht töötab sõne tasandil. Kahe kihi tulemuseks olnud skoorid summeeritakse ja tähemärk klassifitseeritakse klasside tõenäosuste alusel [41].

5.2. Eksperimendid

Automaatsete tekstitöötlusvahendite sõnestajate ja lausestajate kasutamiseks töö eesmärgi täitmiseks pandi need tööle Anaconda keskkonnas Jupyter Notebookis programmeerimiskeele Python skriptiga:

1) EstNLTK versiooni 1.6.5beta tulemuste saamiseks kasutati lisas C5 esitatud koodi;
2) UDPipe'i tulemuste saamiseks rakendati ufal UDPipe'i GitHubi lehelt [43] pärinevat koodi[15] lisas C6 esitatud koodiga;
3) StanfordNLP tulemuste saamiseks kasutati lisas C7 esitatud koodi.

[14] Uurimistöös kasutati versiooni 0.2.0.
[15] https://github.com/ufal/udpipe/blob/master/bindings/python/examples/udpipe_model.py.

Kõikide tekstitöötlusvahendite tulemused konventeeriti vähendatud andmetega CoNLL-U formaati.

Kolme automaatse tekstitöötlusvahendi kasutamise empiirilise kokkuvõttena saab öelda, et kõige lihtsam oli sõnestamise ja lausestamise tulemuseni jõuda EstNLTK abil. Kasutusmugavuselt järgmine oli StanfordNLP, kus sai valida protsessori, kuid andmete sobivale kujule konventeerimisega tuli rohkem vaeva näha. Kasutusmugavuselt viimaseks jäi UDPipe, sest seal tuli kõigepealt leida kood, mille abil seda kasutada, ja siis tuli selgeks teha, kuidas saada toru (ingl *pipeline*) mõttes vahetulemus.

5.3. Lausestamise tulemused

Automaatsete tekstitöötlusvahendite lausestajate tulemuste hindamiseks kasutatakse siinses uurimuses Universal Dependenciese raames loodud CoNLL 2018 Shared Taski hindamisskripti [44], mis võrdleb tulemusi sisseloetud CoNLL-U formaadis failide alusel. Sõnestamise ja lausestamise tulemuste salvestamiseks kasutati vähendatud andmetega CoNLL-U formaati. See tähendab, et skripti väljastatud tulemustest saab vaadata ainult sõnestamise ja lausestamise kohta kolme näitajat: täpsus (ingl *precision*), saagis (ingl *recall*) ja F-skoor (ingl F_1-*score*).

Annoteeritud korpus loetakse hindamisskripti sisendiks *gold*-failina ja automaatse lausestaja lausestamise tulemus *system*-failina. Esimesena kontrollib skript, kas CoNLL-U failides on olemas kõik väljad (kõik kümme välja peavad olemas olema). Kui sõnade kohta pole andmeid, siis tuleb kasutada alakriipsu. Lisaks on nõutud, et seitsmes väli HEAD oleks arv.

Hindamisskript võrdleb sisendfailides sõnade ja lausete alguseid ja lõppe. Sõna või lause loetakse korrektseks, kui mõlemas failis langeb sõna või lause algus ja lõpp kokku. Võrdlemise käigus loendatakse kõik korrektsed sõnad ja laused ning kummagi faili sõnad ja laused, seejärel arvutatakse hindamiseks kasutatavad näitajad järgmiste valemitega:

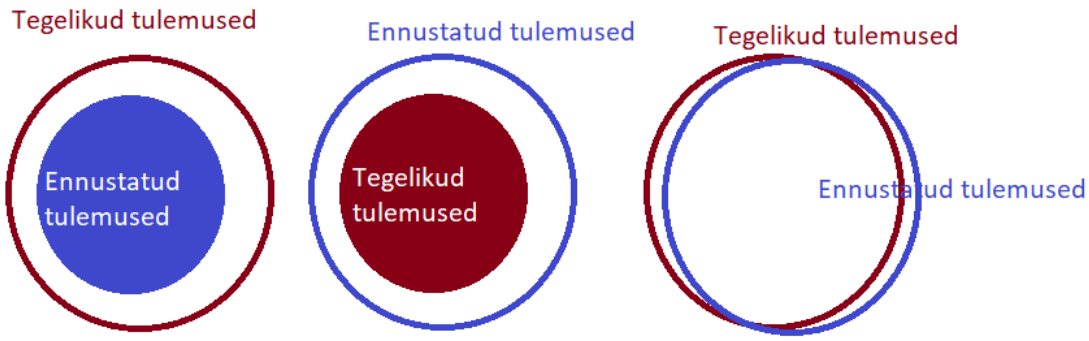
1) $täpsus = \frac{korrektsete\ üksuste\ arv}{üksuste\ arv\ tekstitöötlusvahendi\ tulemustes}$;
2) $saagis = \frac{korrektsete\ üksuste\ arv}{annoteeritud\ korpuse\ üksuste\ arv}$;
3) $F\text{-}skoor = \frac{2 * korrektsete\ üksuste\ arv}{(üksuste\ arv\ tekstitöötlusvahendi\ tulemustes + annoteeritud\ korpuse\ üksuste\ arv)}$.

Täpsus näitab, kui paljud kõikidest lausestaja leitud lausetest olid laused ka annoteeritud korpuses, st korrektsete lausete osakaalu lausestaja leitud kõikide lausete hulgas. Saagis näitab, kui paljud annoteeritud korpuse lausetest tuvastas lausestaja lausena, st korrektsete lausete osakaalu annoteeritud korpuse kõikide lausete seas. Sama kehtib sõnestamise kohta.

F-skoor on täpsuse ja saagise harmooniline keskmine, st sõltub nii täpsusest kui ka saagisest. Kui üks on suur, siis on ka F-skoor suurem, kuid tulemus sõltub veel teisest parameetrist. F-skoor ei saa olla suur, kui täpsus ja saagis mõlemad ei ole suured.

Täpsust ja saagist tuleb tavaliselt vaadata kvaliteedinäitajatena koos, sest eraldivõetuna ei anna nad piisavalt infot. Olukorra kirjeldamiseks tuuakse sageli näide, kuidas täpsus on võimalik suureks saada konservatiivse klassifitseerijaga, st mudeliga, mis ennustab tulemusi arvuliselt vähem, kuid need on korrektsed tulemused. Sellisel juhul on korrektsete tulemuste osakaal suur, kuigi näiteks ennustamata jäi palju juhtumeid. Saagise tulemusi on võimalik mõjutada n-ö liberaalse mudeliga, st mudel ennustab palju tulemusi ja nende hulgas on ka korrektsed tulemused, st korrektsete osakaal tegelike tulemuste hulgas on suur, aga tegelikkuses ennustas mudel veel hulgaliselt tulemusi, mis ei olnud korrektsed.

Joonisel 3 on illustreeritud täpsuse, saagise ja F-skoori erinevaid vahekordi. Esimesel juhul on täpsus maksimaalne, kuid saagis väiksem. Teisel juhul on saagis maksimaalne, kuid täpsus väiksem. Kolmandal juhul on mõlemad suured, st ka F-skoor on suur.



Joonisel 3. Tegelike tulemuste ja ennustatud tulemuste vahekorrad

Siinkohal sooviti võrrelda, kui hästi töötab automaatne lausestaja või sõnestaja võrreldes annoteeritud korpusega. Seega pakub eelkõige huvi see, kui paljud lausestaja märgitud lausepiiridest on ka annoteeritud korpuses lausepiirid. Siinses töös soovitakse leida korrektsete üksuste osakaal annoteeritud üksuste arvus ja peamiselt näitab seda saagis.

Hindamisskript loeb korrektseks üksnes laused, mille algus ja lõpp langevad kokku. Seega ei piisa ainult sellest, kui lause lõpp on märgitud samasse kohta. Kuna tekst moodustab katkematu lõpliku ühekordselt kasutatava tähemärkide jada, milles on algused ja lõpud indeksitega eraldatud, siis ei ole võimalik olukord, kus saagist saab suurendada liberaalse mudeliga. Tegelikkuses on nii, et mida rohkem on tekstis

lauseid (st mida killustatum tekst selle poolest on), seda väiksemaks muutub saagis, sest isegi kui lausete lõpud langevad kokku, siis lausete algused ei lange kokku. Seda kinnitasid katsetuste tulemused.

Hindamisskriptis kasutatava metoodika puhul saab saagis olla 100% ainult siis, kui mõlemas tekstis on kõik lausepiirid tuvastatud ühetaoliselt. Joonisel 3 illustreeritud teine juhtum, kus saagis saab olla 100% ja tuvastati rohkem lauseid, ei ole siinsel juhul võimalik, sest siis peaksid kõik korrektsed laused olema annoteeritud korpuse laused ja lausestaja peaks lisaks tuvastama lauseid, kuid see ei ole võimalik, sest olemasolevad tähemärgid on tuvastatud lausetele n-ö ära kasutatud ja neid ei ole kusagilt juurde võtta.

Olukorra illustreerimiseks on näitena arvutatud täpsus, saagis ja F-skoor kahel juhul. Olgu pidev tähemärkide jada, kus püstkriipsuga on tähistatud lausete algused ja lõpud:

|aaa||bbb||ccc||ddd|,

seega on selles jadas neli lauset.

1) Lausestaja tuvastas laused järgmiselt:

   |aaa||bbbcccddd|,

   järelikult tuvastati korrektseid lauseid üks ja kokku tuvastas lausestaja kaks lauset. Täpsus on 1/2, saagis 1/4 ja F-skoor 1/3, st lausestaja tuvastas ainult ühe neljast annoteeritud korpuse lausest, ent täpsus oli suur.

2) Lausestaja tuvastas laused järgmiselt:

   |aaa||bb||b||cc||c||ddd|,

   järelikult tuvastati korrektseid lauseid kaks ja kokku tuvastas lausestaja kuus lauset. Täpsus on 2/6, saagis 2/4 ja F-skoor 2/5. Kuigi lausestaja tuvastas rohkem lauseid, kui oli annoteeritud lauseid, saame saagisest siiski teada, kui suur osa annoteeritud korpuse lausetest tuvastati.

Selle uurimuse valguses sobib lausestamise ja sõnestamise tulemuste hindamiseks kõige paremini saagis juhul, kui tahetakse hinnata, mil määral lausestas ja sõnestas automaatne tekstitöötlusvahend teksti lauseid sarnaselt annoteeritud korpusega. Tulemuste võrdlemiseks on esitatud kõik kolm näitajat (täpsus, saagis ja F-skoor), kuna igaüks neist sisaldab teavet, nagu näitest oli näha, kuid peamiselt on pööratud tähelepanu saagisele.

Iga lausestaja tulemusi võrreldi kõigi (korpus A) ja enim märgitud annotatsioonidega korpusega (korpus B), kus olid annoteeritud

1) nii süntaktiliste kui ka ortograafiliste lausete piirid (versioon 1);
2) ortograafiliste lausete piirid (versioon 2);
3) ortograafiliste lausete piirid ja lausestaja tuvastatud süntaktiliste lausete piirid (hindamisel ei loetud veaks, kui lausestaja ei tundnud ära süntaktilist lauset, kuid kui lausestaja tundis selle ära, siis see loeti korrektseks, versioon 3).

Edaspidi kasutatakse korpuste kohta nimetusi korpus A1, A2, A3, B1, B2, B3 (A1 on kõigi süntaktilise ja ortograafilise lause piiri annotatsioonidega korpus jne), sest nii on nimetusse koondatud info annotatsioonide ja lauseliikide kohta.

Korpuse A1 ja korpuse B1 erinevusest annab parema ülevaate nende lausete arv: korpuses A1 on 2354 lauset ja korpuses B1 on 1975 lauset. See tähendab, et vähemusse jäänud (n-ö üldisest hinnangust kõrvale kalduvate) lausepiiride eemaldamise järel jäi 379 lauset vähemaks. Alles jäid ainult need laused, mille puhul enamik annoteerijaid arvas, et selles kohas on lause lõpp, seega on selle korpuse versiooni lausepiiride usaldusväärsus suurem. Kui jätta korpusse B ainult ortograafilised laused, jääb alles 1731 lauset, mis on võrreldavatest versioonidest kõige väiksema lausete arvuga korpus.

Tabelis 5 on esitatud automaatsete lausestajate tulemused saagis, täpsus ja F-skoor korpuste A ja B eri versioonidel.

Tabel 5. Lausestajate tulemused (saagis, täpsus, F-skoor)

		EstNLTK			UDPipe			StanfordNLP		
Nr	**Korpus**	**Saagis**	**Täpsus**	**F-skoor**	**Saagis**	**Täpsus**	**F-skoor**	**Saagis**	**Täpsus**	**F-skoor**
1.	Korpus A1	48,68	67,45	56,55	40,36	61,89	48,86	48,94	66,78	56,48
2.	Korpus A2	81,96	86,93	84,38	68,98	80,98	74,50	82,85	86,55	84,66
3.	Korpus A3	78,90	86,93	82,72	65,82	81,30	72,75	80,75	86,84	83,69
4.	Korpus B1	68,91	80,11	74,09	58,13	74,79	65,41	69,82	79,94	74,54
5.	Korpus B2	86,77	88,40	87,58	72,21	81,43	76,55	86,71	87,01	86,86
6.	Korpus B3	85,05	88,40	86,70	70,57	82,15	75,92	85,96	88,06	87,00

Tabelis 5 esitatud tulemustest koos lisas B1 toodud andmetega on näha, et mida rohkem on korpuses lauseid (mida killustunum on tekst), seda kehvemad on tulemused.

Korpuses A1 on lauseid kõige rohkem ja seega võiks eeldada, et lausestajal oli suurem võimalus lausepiiri tabada, tegelikkuses see aga nii ei olnud. Korpuse A1 saagis on vahemikus 40–49%. Korpuses B1 oli 379 lauset vähem ja saagis jäi vahemikku 58–70%. Põhjus seisneb selles, nagu juba mainitud, et hindamismeetodi järgi loetakse samaks lause, mis algab ja lõpeb samas kohas. Kuna kõigi annotatsioonidega korpuses on laused rohkem killustatud, siis isegi juhul, kui nende lõpud kattuvad, ei kattu algused.

Et korpuses B1 on laused vähem killustatud, on ka suurem võimalus, et kattuvad nii lause algus kui ka lõpp ja lause loetakse korrektselt tuvastatuks.

Korpuses A on n-ö müra rohkem, sest seal on kõikide annoteerijate lausepiirid annoteeritud, kuigi ainult üks kolmest või viiest võis anda hinnangu, et lausepiir on just selles kohas. Seda arvestades tuleks pöörata lausestajate võrdlemisel peatähelepanu korpuse B versioonidega seotud tulemustele, sest sellest korpusest on eemaldatud annoteerijate teistest erinevad (n-ö kõrvalekalduvad) annotatsioonid.

Teine korpustega seotud näitaja on lauseliigid. Versioonis 1 on annoteeritud nii süntaktilised kui ka ortograafilised laused, versioonis 2 ainult ortograafilised laused ja versioonis 3 ortograafilised laused koos tuvastatud süntaktiliste lausetega. Tabelist 5 on näha, et tulemused sõltuvad paljuski sellest, kas tegu on korpuse versiooniga, kus on süntaktilise lause piirid annoteeritud. Näiteks süntaktilisi lauseid sisaldavate korpuste A1 ja B1 saagised on 40–70%, samas kui ainult ortograafilisi lauseid siseldavate korpuste A2 ja B2 saagised on vahemikus 68–87%. Seega on olnud lausestajatel lihtsam tuvastada ainult ortograafilisi lauseid. Seda võis ka eeldada, sest nende mudeleid on treenitud või reegleid koostatud just selliste lausete tuvastamiseks.

Kuna aga uue meedia keele tekstid on siinse töö alguses kirjeldatud põhjustel võrreldes standardse tekstiga mürarikkad, ei saavutanud lausestajad ka ortograafiliste lausetega korpuse lausestamisel nende kohta veebilehtedel avaldatud F-skoori. StanfordNLP ja UDPipe'i veebilehel on väidetud, et StanfordNLP lausestaja F-skoor on 93,82% ja UDPipe'i Estonian-EDT lausestaja puhul 91,6%. Kui võrrelda korpusel B2 saadud F-skoori tulemusi eelnimetatud F-skooridega, siis need on siiski väiksemad (StanfordNLP: 86,86%, UDPipe: 76,55%).

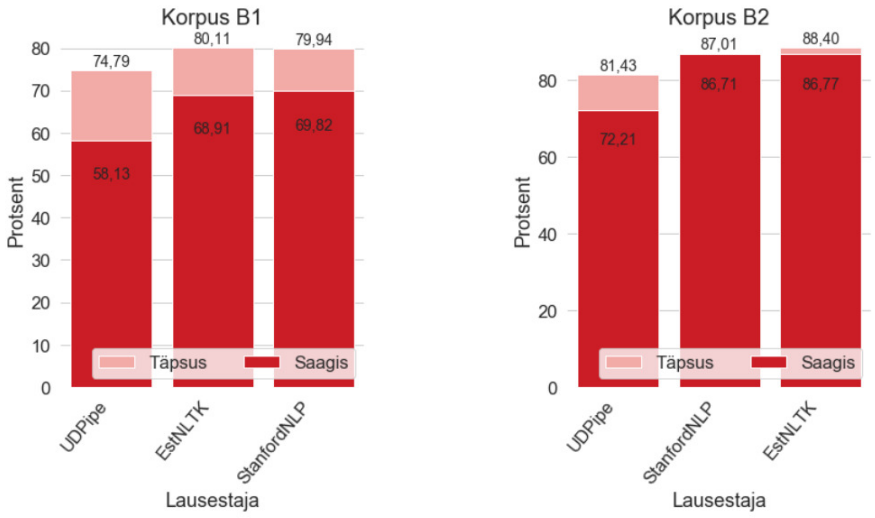
Spetsiaalselt uue meedia keele jaoks treenitud lausestajaga oleks tõenäoliselt võimalik tulemusi parandada, eriti süntaktilise lause puhul. Uue meedia keele tekstis on sellise lause osakaal suurem kui standardses tekstis, sest kirjavahemärke ja suurtähti kasutatakse lause alguses juhuslikult. Lausestajate tulemused süntaktiliste lausetega korpusel on siiski liiga väiksed, arvestades, et lausestajate saagis on korpuse B1 puhul vahemikus 58–70%.

Kui hinnata, kas korrektselt tuvastatud süntaktilised laused ortograafiliste lausete kõrval parandavad tulemusi, siis tegelikkuses see nii ei ole. Korpuse versioonis 3 olid märgistatud ortograafilised laused ja lisaks need süntaktilised laused, mille lausestaja tuvastas. Sellegipoolest on tulemused läinud kehvemaks: korpuste A3 ja B3 puhul on saagis halvem kui korpuse A2 ja B2 puhul. Korpuste A3 ja B3 lausete arv on suurem kui korpustes A2 ja B2 (vt lisa B1). See omakorda mõjutas saagist, kuna tegu on suhtarvuga. Kuigi korrektselt tuvastatud lausete arv oli arvuna korpuse B3 puhul suurim, oli ka lausete

arv nii palju suurem, et suhtarv ei muutunud versiooni 3 puhul paremaks. Et osaline süntaktilise lausega arvestamine tulemusi oluliselt ei mõjutanud, tuleks edaspidi pöörata tähelepanu versioonidele, mis sisaldasid ortograafilisi ja süntaktilisi lauseid või ainult ortograafilisi lauseid.

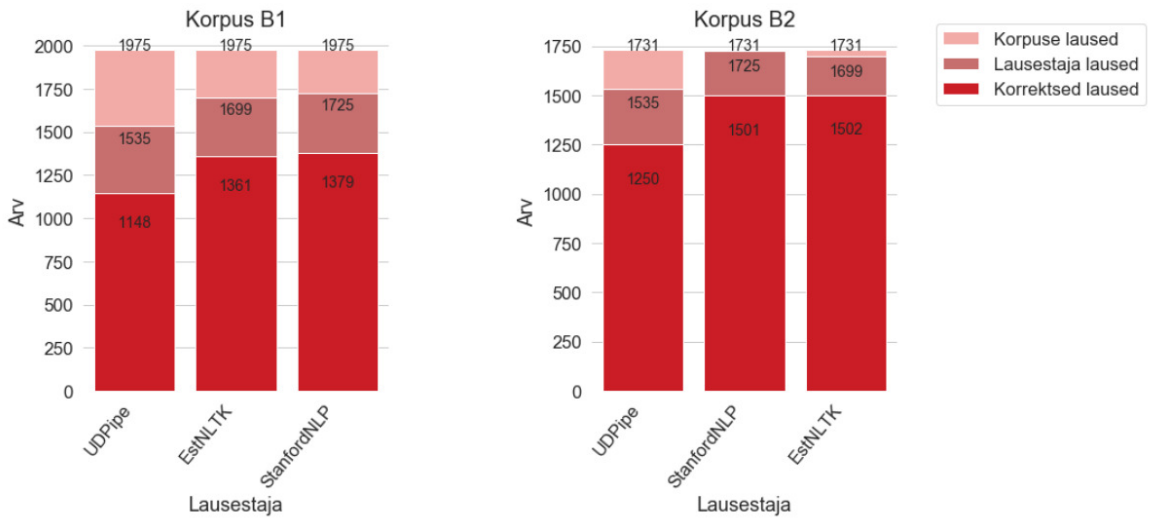
Vahekokkuvõttena võib järeldada, et lausestajate võrdlemiseks on piisav, kui võrrelda saagist korpuse versioonidel B1 ja B2.

Ehkki saagis näitab, kui palju annoteeritud korpuse lausetest tuvastati, on siiski oluline ka täpsus, sest täpsus näitab teistpidi ka seda, kui suur oli nende lausete osakaal, mida korpuses ei olnud. Kui lausestaja killustas teksti väga paljudeks lauseteks, mida korpuses ei olnud, väheneb täpsus. Üldiselt on kõikide lausestajate täpsus korpuse eri versioonidel olnud suurem kui saagis. Lisast B1 on näha, et lausestajad on üldjuhul tuvastanud arvuliselt vähem lauseid, kui neid on korpuse versioonides. Kuna täpsus on suhtarv, mis sõltub lausestaja tuvastatud lausete arvust, siis ongi võimalik, et täpsus on valdavalt suurem kui saagis. Täpsuse ja saagise vahe väheneb korpuste neis versioonides, kus süntaktilise lause piire ei arvestata (vt joonis 4). Sellisel juhul kahaneb ka korpuses olevate lausete arv (vt joonis 5) ja see avaldab saagisele mõju.



Joonis 4. Lausestajate täpsus ja saagis korpustel B1 ja B2 (tulemused järjestatud saagise alusel)

Nagu jooniselt 4 on näha, on StanfordNLP ja EstNLTK tulemused üsna sarnased, UDPipe'i tulemused on tunduvalt kehvemad. Saagise alusel järjestades on StanfordNLP tulemus parem korpusel B1 ja EstNLTK tulemus korpusel B2. See tähendab, et süntaktiliste ja ortograafiliste lausetega korpuse korral on StanfordNLP tuvastanud lauseid paremini kui EstNLTK, ent puhtalt ortograafiliste lausete tuvastamises on EstNLTK parem, kuigi vahe on väike.



Joonis 5. Lausete arvud korpuste B1 ja B2 lausestamisel

Kui vaadata joonisel 5 näha olevaid korrektselt tuvastatud lausete arvu, siis korpusega B1 võrreldes tuvastas EstNLTK korrektselt 1361 lauset ja StanfordNLP 1379 lauset, võrreldes korpusega B2 vastavalt 1502 ja 1501. Kuna ortograafilisi lauseid on mõlemad tuvastanud korrektselt suhteliselt sarnaselt, siis võib oletada, et StanfordNLP tuvastab süntaktilisi lauseid paremini. Tuvastatud lausete koguarv on StanfordNLP puhul suurem kui EstNLTK puhul ning see näitab samuti, et StanfordNLP on teksti killustanud rohkemateks lauseteks ja teinud seega lühemaid lauseid (st lauseid tükeldades ei ole ta jõudnud tingimata ortograafilise lause piirini, vaid on tükeldanud ka väljaspool seda).

Täpsuse järgi on EstNLTK tulemus mõlema korpuse puhul StanfordNLP tulemusest parem. EstNLTK on tuvastanud vähem lauseid, seega on korrektsete lausete osakaal tuvastatud lausete seas suurem kui StanfordNLP puhul, kuigi korrektselt tuvastatud lausete arv on suhteliselt sarnane.

Kokkuvõttes saab järeldada, et uue meedia teksti lausestamisel andsid EstNLTK ja StanfordNLP ligilähedase tulemuse, kuid mõlemal on uue meedia keele eripära arvestamisel arenguruumi, eelkõige võrreldes tulemusega, mille nad saavutavad süntaktilisi lauseid sisaldava korpusega.

Uurimistöö alguses testiti ka EstNLTK versiooni 1.4.1, mille tulemused on esitatud lisas B1. Tulemustest on näha, et selle versiooni saagised olid korpuste B1 ja B2 puhul UDPipe'i saagistest paremad, kuid kehvemad kui StanfordNLP-l ja EstNLTK versioonil 1.6.5beta.

Uurimistöö tegemise ajal toimus testitavas tarkvaras teinegi muudatus: StanfordNLPd arendati edasi ja see nimetati ümber Stanzaks [45]. Kuna annoteeritud korpus oli olemas ja StanfordNLP rakendamise skripti oli vaja muuta ainult väikses osas, siis testiti töös ka Stanzat (tulemused on esitatud lisas B1).

Stanza[16] eesti keele mudel et_edt on treenitud Eesti universaalsete sõltuvustega märgendatud puudepanga (UD Estonian (EDT)) versiooni 2.5 alusel. Stanza veebilehel on et_edt mudeli F-skooriks märgitud sõnestamisel 99,96% ja lausestamisel 93,32% [46]. Mudeli sooritus on võrreldes StanfordNLPga sõnestamisel 0,01% kasvanud ja lausestamisel 0,5% vähenenud. Testimisel saadud tulemused kinnitasid, et StanfordNLP uuema versiooni (Stanza) tulemused lausestamisel on kehvemad kui StanfordNLP tulemused. Korpuse versioonide B1 ja B2 puhul jäi Stanza saagis alla nii StanfordNLP kui ka EstNLKT versiooni 1.6.5beta saagisele, kuid oli parem kui EstNLTK versiooni 1.4.1 saagis.

5.4. Sõnestamise tulemused

Uurimistöös oli peale lausestamise tulemuste võimalik võrrelda ka sõnestamise tulemusi, sest lausestamine eeldab sõnestamist. Töö raskuskese on siiski lausestajate tulemuste võrdlemisel. Sõnestamise hindamiseks kasutati sama CoNLL 2018 Shared Taski hindamisskripti, mida kasutati lausestamise puhul.

Tabelis 6 on toodud automaatsete tekstitöötlusvahendite sõnestamise tulemused.

Tabel 6. Automaatsete tekstitöötlusvahendite sõnestamise tulemuste võrdlus annoteeritud korpusega

Nr	Sõnestaja	Täpsus	Saagis	F-skoor
1.	EstNLTK	98,13	98,93	98,53
2.	UDPipe	97,06	96,16	96,61
3.	StanfordNLP	96,54	96,46	96,50

Selgus, et EstNLTK sõnestaja on tulemuste poolest parim ja StanfordNLP sõnestaja tulemused on kõige kehvemad, v.a saagise järgi on halvim tulemus UDPipe'i sõnestajal. UDPipe'i F-skoor on tänu suuremale täpsusele parem kui StanfordNLP oma. Siinjuures tuleks aga pöörata tähelepanu asjaolule, et korrektseid sõnu on StanfordNLP tuvastanud arvuliselt rohkem (vt lisa B2).

Kui lausestamise tulemuste poolest oli StanfordNLP uuema versiooni Stanza tulemused kehvemad, siis sõnestamisel töötab Stanza tulemuslikumalt, kuigi erinevused on väikesed (vt lisa B2). EstNLTK versiooni 1.4.1 sõnestamise tulemused järgnesid EstNLTK versioonile 1.6.5beta ja olid teistest siin nimetatutest paremad (vt lisa B2).

Standardse teksti kohta oli veebilehtedel F-skooridena toodud UDPipe Estonian-EDT mudeli puhul 100%, StanfordNLP puhul 99,95% ja Stanza et_edt puhul 99,96%. Tulemused erinevad võrreldes

[16] Uurimistöös kasutati versiooni 1.0.1.

standardse teksti F-skooridega, kuid vahed ei ole väga suured (UDPipe: 96,16%, StanfordNLP: 96,5%, Stanza: 96,78%).

Kokkuvõttes selgus, et kõik sõnestajad töötavad hästi ja pole olulisi erinevusi, mille pärast tuleks eelistada ühte automaatset tekstitöötlusvahendit teisele.

Kokkuvõte

Uue meedia keel on võrgusuhtluskeel, mille osakaal järjest suureneb. Töös näidati, et sellise keele tekstid erinevad kirjakeele normile vastavatest tekstidest nii suurel määral, et on põhjust seada kahtluse alla olemasolevate automaatsete tekstitöötlusvahendite tulemuslikkus nende lausestamisel.

Töös võrreldi kolme automaatse tekstitöötlusvahendi eri versioonide lausestamise tulemusi annoteeritud korpusega. Kuna uue meedia keele tekstides on lauselõpumärkide kasutus juhuslik, siis võrreldi lausestajate tulemusi annoteeritud korpusega, kus olid tähistatud ka nn süntaktilise lause (üksus, mis väljendab terviklikku sõnumit) piirid.

Parim automaatne tekstitöötlusvahend suutis tuvastada 69,82% (saagis) annoteeritud korpuse lausetest juhul, kui peale ortograafiliste lausete olid annoteeritud ka süntaktilised laused. Samas oli ainult ortograafiliste lausete tuvastamise korral parim tulemus 86,77% (saagis). F-skooride võrdlus näitas, et ortograafiliste lausete puhul oli lausestajate tulemus siiski kehvem, kui kasutatud tekstitöötlusvahendite veebilehel on näitajana nende kohta avaldatud. Tõenäoliselt erineb uue meedia keele tekst oma mittestandardsusega kirjakeele normile vastava keele tekstist nii palju, et lausestajad ei saavuta standardsete tekstidega saadud tulemusi isegi siis, kui tuvastatakse ainult ortograafilisi lauseid.

Kui mõelda ka semantilise analüüsi peale, ei ole süntaktiline lause oluline mitte ainult uue meedia keele teksti puhul, vaid sisaldab terviklikku sõnumit ka kirjakeele normile vastavas tekstis. Selle tuvastamise suutlikkus võib olla laiemalt vajalik selleks, et teha loomulik keel arvutile arusaadavaks.

Uue meedia keele tekstide osakaalu suurenemisega seoses tuleb rohkem tähelepanu pöörata ka olemasolevate automaatsete tekstitöötlusvahendite täiendamisele. Kindlasti on neil arenguruumi. Ka teised autorid on asunud oma uurimustes seisukohale, et uue meedia keelele sobivaid tööriistu ja meetodeid on väga vaja [21].

Kokkuvõttes olid eestikeelse uue meedia teksti puhul sõnestamise ja lausestamisega seotud näitajad kõige paremad EstNLKT versioonil 1.6.5beta. Kui anda lausestamise tulemustele suurem kaal kui sõnestamise tulemustele, järgneb sellele näitajate poolest StanfordNLP. Kui arvestada aga, et StanfordNLPs töötavad nii sõnestaja kui ka lausestaja täielikult statistiliste meetodite alusel (st lausestamine ja sõnestamine on selgeks õpitud korpuste pealt, ilma eelteadmisteta), on järelikult võimalik saavutada täielikult statistilise mudeliga samaväärseid tulemusi kui reeglipõhise mudeliga.

Lõpetuseks tänan oma juhendajat Kairit Sirtsi võimaluse eest aidata tal uurida tema jaoks olulist teemat. Sageli juhtub ülikooli lõputöödega nii, et need kirjutatakse valmis, aga päriselus pole nende tulemusi

kellelegi vaja, sest need teenivad üksnes laiemat eesmärki õpetada üliõpilast akadeemiliselt mõtlema. Tänu oma juhendajale sain kasutada oma aega mõttestatult, tehes abistavaid tegevusi tema uurimuse jaoks. Olen talle tänulik huvitava ja õpetliku rännaku eest keeletehnoloogia valdkonda ning ajale, mille ta pühendas mu juhendamisele.

Viidatud kirjanduse loetelu

[1] Hjelmslev L. Sissejuhatus keeleteooria alustesse. Tallinn: Eesti Keele Sihtasutus. 2012.

[2] Karlsson F. Üldkeeleteadus. Tallinn: Eesti Keele Sihtasutus. 2002.

[3] Ariva L., Eskor L. Mis on arvutilingvistika? *Oma Keel*, 2004, 1, lk 34–44.

[4] EstNLTK. Teksti tükeldamine lõikudeks. https://estnltk.github.io/estnltk/1.1/tutorials/tokenization_est.html (15.04.2020)

[5] Liin, K., Muischnek, K., Müürisep, K.,Vider, K. Eesti keel digiajastul = The Estonian language in the digital age. META-NET Valge raamatu sari. Heidelberg: Springer. 2012.

[6] Muischnek K., Kaalep H.-J., Sirel R. Korpuslingvistiline lähenemine Eesti internetikeele automaatsele morfoloogilisele analüüsile. Eesti Rakenduslingvistika Ühingu aastaraamat, 2011, 7, 111–127.

[7] EstNLTK. https://estnltk.github.io/ (27.04.2020)

[8] UDPipe. Charles University, Czech Republic. http://ufal.mff.cuni.cz/udpipe (27.04.2020)

[9] StanfordNLP. https://stanfordnlp.github.io/stanfordnlp/index.html (27.04.2020)

[10] Oja A. Eesti keel internetis. Keel ja arvuti. Tartu Ülikooli üldkeeleteaduse õppetooli toimetised, 2006, 6, 259–267.

[11] Oja A. Sissevaateid internetisuhtlusse. *Oma Keel*, 2010, 1, lk 11–18.

[12] Salla S. Jututuba kui võrgusuhtlusvorm. Tekstid ja taustad. Artikleid tekstianalüüsist. Tartu Ülikooli eesti keele õppetooli toimetised, 2002, 23, 128–156.

[13] Kerge K. Veebikommentaariumi mitmetahuline maailm. Tekstid ja taustad III. Lingvistiline tekstianalüüs. Tartu Ülikooli eesti keele õppetooli toimetised, 2004, 28, 51–73.

[14] Soodla K. Morfoloogilisi, morfosüntaktilisi ja sõnamoodustuslikke erijooni eesti internetikeeles. Teadusmagistritöö. Tartu: Tartu Ülikool, filosoofiateaduskond, eesti keele osakond. 2010.

[15] Särg D. Internetikeele automaatne süntaktiline analüüs kitsenduste grammatikaga. Eesti Rakenduslingvistika Ühingu aastaraamat, 2015, 12, 253–267.

[16] Sõnaveeb. sonaveeb.ee (21.04.2020)

[17] e-Teatmik: IT ja sidetehnika seletav sõnaraamat. http://www.vallaste.ee (28.04.2020)

[18] Keeleressursid. https://keeleressursid.ee/et/keeleressursid (28.04.2020)

[19] Crystal D. Language and the Internet. Cambridge: Cambridge University Press. 2001.

[20] Erelt M., Erelt T., Ross K. Eesti keele käsiraamat. Kolmas, täiendatud trükk. Tallinn: Eesti Keele Sihtasutus. 2007.

[21] Särg D., Muischnek, K., Müürisep, K. Annotated Clause Boundaries' Influence on Parsing Results. 21st International Conference, TSD 2018, Brno, Czech Republic, September 11–14, 2018. *Proceedings: 21st International Conference on Text, Speech and Dialogue, Brno, September 11–14, 2018*, 2018, pp. 171–179.

[22] Eesti keele koondkorpus. https://www.cl.ut.ee/korpused/segakorpus/ (04.04.2020)

[23] etTenTen. https://www.keeleveeb.ee/dict/corpus/ettenten/about.html (24.04.2020)

[24] Kallas J., Koppel, K., Tuulik, M. Korpusleksikograafia uued võimalused eesti keele kollokatsioonisõnastiku näitel. Eesti Rakenduslingvistika Ühingu aastaraamat, 2015, 11, 75–94.

[25] EstSyntax / EstUD. https://github.com/EstSyntax/EstUD (25.04.2020)

[26] Raudvere, U. estnltk, 23. märts 2016. https://github.com/estnltk/estnltk/issues/56 (25.04.2020)

[27] Text segmentation: Compound tokens. https://github.com/estnltk/estnltk/blob/version_1.6/tutorials/nlp_pipeline/B_02_segmentation_compound_tokens.ipynb (30.04.2020)

[28] Universal Dependencies. CoNLL-U Format. https://universaldependencies.org/format.html (25.04.2020)

[29] Dice, L. R. Measures of the Amount of Ecologic Association Between Species. *Ecology*, 1945, 26, pp. 297–302.

[30] Inter-rater agreement Kappas. https://towardsdatascience.com/inter-rater-agreement-kappas-69cd8b91ff75 (26.04.2020)

[31] Fleiss, J. L. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 1971, vol. 76, nr. 5, pp. 378–382.

[32] Landis J. R., Koch G. G. The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 1977, vol. 33, nr. 1, pp. 159–174.

[33] Universal Dependencies. https://universaldependencies.org (27.04.2020)

[34] Basic NLP toolchain. https://github.com/estnltk/estnltk/blob/version_1.6/tutorials/nlp_pipeline/A_01_short_introduction_and_tutorial_for_linguists.ipynb (30.04.2020)

[35] Text segmentation: Tokens. https://github.com/estnltk/estnltk/blob/version_1.6/tutorials/nlp_pipeline/B_01_segmentation_tokens.ipynb (30.04.2020)

[36] Text segmentation: Sentences. https://github.com/estnltk/estnltk/blob/version_1.6/tutorials/nlp_pipeline/B_04_segmentation_sentences.ipynb (30.04.2020)

[37] Tag and parse text with UDPipe. https://corpy.readthedocs.io/en/stable/guides/udpipe.html#overview (27.04.2020)

[38] UDPipe Models. http://ufal.mff.cuni.cz/udpipe/models (27.042020)

[39] Straka, M., Strakova, J. Tokenizing, POS Tagging, Lemmatizing and Parsing UD 2.0 with UDPipe. *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, august 2017*. http://ufal.mff.cuni.cz/~straka/papers/2017-conll_udpipe.pdf (27.04.2020)

[40] Straka, M., Hajič, J., Straková, J. UDPipe: Trainable Pipeline for Processing CoNLL-U Files Performing Tokenization, Morphological Analysis, POS Tagging and Parsing. *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, 2016. http://ufal.mff.cuni.cz/~straka/papers/2016-lrec_udpipe.pdf (27.04.2020)

[41] Qi, P., Dozat, T., Zhang, Y., Manning, C. D. Universal Dependency Parsing from Scratch. *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. https://nlp.stanford.edu/pubs/qi2018universal.pdf (27.04.2020)

[42] System Performance. https://stanfordnlp.github.io/stanfordnlp/performance.html (27.04.2020)

[43] ufal/udpipe. https://github.com/ufal/udpipe/blob/master/bindings/python/examples/udpipe_model.py (27.04.2020)

[44] Universal Dependencies. CoNLL 2018 Shared Task. Evaluation. http://universaldependencies.org/conll18/evaluation.html (25.04.2020)

[45] Stanza. https://stanfordnlp.github.io/stanza/index.html (30.04.2020)

[46] System Performance. https://stanfordnlp.github.io/stanza/performance.html (30.04.2020)

[47] Association for Computational Linguistics (ACL). SIGNLL. http://www.signll.org/ (25.04.2020)

[48] Buchholz, S., Marsi, E. CoNLL-X shared task on Multilingual Dependency Parsing. *Proceedings of the 10th Conference on Computational Natural Language Learning* (CoNLL-X), 2006, pp. 149-164. https://www.aclweb.org/anthology/W06-2920.pdf (25.04.2020)

[49] Qi, P., Zhang, Y., Zhang, Y., Bolton, J., Manning, C. D. Stanza : A Python Natural Language Processing Toolkit for Many Human Languages. Association for Computational Linguistics (ACL) System Demonstrations, 2020. https://arxiv.org/pdf/2003.07082.pdf (01.05.2020)

[50] estnltk 1.4.1. https://estnltk.github.io/estnltk/1.4.1/index.html 30.04.2020

[51] UD_Estonian-EDT, UD_Estonian-EWT. https://universaldependencies.org/treebanks/et-comparison.html (02.05.2020)

Lisad

Lisa A. Näiteid käsitsi parandatud EstNLTK sõnestamisjuhtude kohta

Näidetes on esimesena esitatud EstNLTK sõnestaja tulemus ja teisena manuaalselt parandatud tulemus.

1) Loetelu tähistavad loendimärgid on lahku sõnestatud, kuigi on tuvastatav, et tegu on loeteluga.

ja mitte sellepärast , et a) tal on võib-olla rohkem raha , b) tema on väljakutsuja , c) ta on " tugevam " sugupool , vaid austusest naisterahva ees .
ja mitte sellepärast , et a) tal on võib-olla rohkem raha , b) tema on väljakutsuja , c) ta on " tugevam " sugupool , vaid austusest naisterahva ees .

Minu meelest on õige järjekord : 1) Kise 2) Toy 3) Giku 4) Tzan 5) Kak
Minu meelest on õige järjekord : 1) Kise 2) Toy 3) Giku 4) Tzan 5) Kak

kui (a) huvi kaob (b) minust saab profikasutaja
kui (a) huvi kaob (b) minust saab profikasutaja

2) Loendimärgina on kasutatud „-“ ja see on järgmisest sõnest lahku sõnestamata.

KUI SA ... -leiad , et on ikka veel võimatu kontrollida ärevust ! Ja üleüldse igapäevast stressi ... -tead , et võibolla on sul üks hea nädal , millele aga kohe järgneb uus ja HALB ! -oled hirmul sõites autoga , kartes , et sind tabab järjekordne HOOG !
KUI SA ... - leiad , et on ikka veel võimatu kontrollida ärevust ! Ja üleüldse igapäevast stressi ... - tead , et võibolla on sul üks hea nädal , millele aga kohe järgneb uus ja HALB ! - oled hirmul sõites autoga , kartes , et sind tabab järjekordne HOOG !

3) Arvu kirjutamisel on tuhandike eraldamiseks kasutatud tühikut, kuid seetõttu on arvu teine pool lahku sõnestatud.

mees oma ligi 30 000 palga juures
mees oma ligi 30000 palga juures

64 000 euro
64000 euro

10 000 dollari
10000 dollari

600 000 planeeritavast miljonist ruutmeetrist
600000 planeeritavast miljonist ruutmeetrist

200 000 ruutmeetrit
200000 ruutmeetrit

4) Protsendi märk ei ole arvust eraldatud.

Kui aga 60% neist
Kui aga 60 % neist

lasi umbes 97% kõigist kuludest mul kanda , 98% kõigist töödest mul teha
lasi umbes 97 % kõigist kuludest mul kanda , 98 % kõigist töödest mul teha

sest 99% ajast

sest 99 %

5) Korduvad kirjavahemärgid on sõnestatud eraldi. Sellised probleemid olid sageduse poolest esikohal.

Mida siin keerulist on ? ? ? Esmakohtingul Väljakutsuja maksab . . . Õigemini peab olema valmis maksma . . . aga viisakas oleks enne Daamilt küsida , kas ta tohib maksta ! ! !
Mida siin keerulist on ??? Esmakohtingul Väljakutsuja maksab ... Õigemini peab olema valmis maksma ... aga viisakas oleks enne Daamilt küsida , kas ta tohib maksta !!!

rõhutan võibolla . . .
rõhutan võibolla ...

ma tahan teda tundma õppida . .
ma tahan teda tundma õppida ..

see miskit maksaks . . . . .
see miskit maksaks .....

On see siis rõõm ! ?
On see siis rõõm !?

Rääkides veel võrdõiguslikkusest . . .
Rääkides veel võrdõiguslikkusest ...

Kust see raha tuli ? ?
Kust see raha tuli ??

Kalevipoega pole tulnud ! !
Kalevipoega pole tulnud !!

mis mull selle loo lõpus nyyd siis oli ? ?
mis mull selle loo lõpus nyyd siis oli ??

“ Ahaa , VAHELE JÄID ! Nüüd me alles KEERAME sulle !!!”
“ Ahaa , VAHELE JÄID ! Nüüd me alles KEERAME sulle !!! ”

sest tuli on valus ju ! ! !
sest tuli on valus ju !!!

ühest äärmusest teise ? !
ühest äärmusest teise ?!

kst sa ei saa ära minna ! !
kst sa ei saa ära minna !!

seal sama koha peal prügimägi ! !
sama koha peal prügimägi !!

seal olnud juba ! !
seal olnud juba !!

Jälle Edgar ! ! Küll tema ka jõuab ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! !
Jälle Edgar !! Küll tema ka jõuab !!!!!!!!!!!!!!!

kas talle ei meeldi kajakad vää ? ?
kas talle ei meeldi kajakad vää ??

Kas mul on õigus ? ? ?
Kas mul on õigus ???

6) Emotikoni tervik on lõhutud (enamiku tundis ära).

Äkki puberteet ? : d
Äkki puberteet ? :d

7) Kirjavahemärk ei ole jutumärgist eraldatud (siin ja mõnes teiseski kohas tuleb arvestada, et uue meedia keeles ei kasutata ainult neid märkide versioone, mida standardi järgi kasutatakse eesti keele märkidena).

Nüüd me alles KEERAME sulle !!!”
Nüüd me alles KEERAME sulle !!! ”

« See sõna on ränk , kes suudab seda kuulda ?»
« See sõna on ränk , kes suudab seda kuulda ? »

« Seepärast ma olengi teile öelnud , et keegi ei saa tulla minu juurde , kui talle ei ole seda andnud Isa .»
« Seepärast ma olengi teile öelnud , et keegi ei saa tulla minu juurde , kui talle ei ole seda andnud Isa . »

“ registreerimisel ”,
“ registreerimisel ” ,

8) Ülakomade vahel olev tekst ei ole eraldi sõnestatud.

nendest'soovitustest'
nendest ' soovitustest '

9) Veebilehe aadress on eraldi sõnestatud.

tehnokratt . net / hyphenator / mergeAndPack . html
tehnokratt.net/hyphenator/mergeAndPack.html

http :// vimeo.com / 23869111
http://vimeo.com/23869111

10) Failinimi on laiendist lahku sõnestatud.

hyphenator . js jaoks et . js
hyphenator.js jaoks et.js

functions . php
functions.php

index . php
index.php

tehnokratt . net’i
tehnokratt.net’i

11) Domeeni riigitähis on punktist lahku sõnestatud.

. ru
.ru

mitte-.ru
mitte - .ru

12) Käändelõpp on eraldi sõnestatud.

Quality guidelines ’ i
Quality guidelines’i

WPtouch ‘ ile
WPtouch‘ile

13) Kellaaeg on eraldi sõnestatud.

kell 8 : 25
kell 8:25

1 : 20
1:20

14) Suhtarv või seis on lahku sõnestatud.

kõige 50 / 50 poolitamist
kõige 50/50 poolitamist

1 : 1 suuruses
1:1 suuruses

6 : 7 , 2 : 6
6:7 , 2:6

7 : 2
7:2

1 : 6 , 6 : 7
1:6 , 6:7

6 : 7 , 6 : 3 , 6 : 2
6:7 , 6:3 , 6:2

1 : 6 , 2 : 6
1:6 , 2:6

6 : 4 , 4 : 6 , 6 : 2
6:4 , 4:6 , 6:2

15) Punkt on lühendist lahku sõnestatud (ka selles ei järgita uue meedia keeles õigekirjareegleid, kuid seda võib sageli juhtuda ka standardsetes tekstides).

ps .
ps.

jms .
jms.

PS .
PS.

jne .
jne.

e .
e.

Lisa B. Automaatsete tekstitöötlusvahendite tulemused

B1. Lausestamise tulemused

Tabel 1: Testitud lausestajate tulemused

Nr	Lausestaja	Võrreldav korpuse versioon	Korrektsete lausete arv	Korpuse lausete arv	Lausestaja tuvastatud lausete arv	Täpsus (%)	Saagis (%)	F-skoor (%)
1	EstNLTK 1.4.1	Korpus A1	1065	2354	1646	64.70	45.24	53.25
2	EstNLTK 1.4.1	Korpus A2	1392	1802	1646	84.57	77.25	80.74
3	EstNLTK 1.4.1	Korpus A3	1392	1847	1646	84.57	75.37	79.70
4	EstNLTK 1.4.1	Korpus B1	1277	1975	1646	77.58	64.66	70.53
5	EstNLTK 1.4.1	Korpus B2	1412	1731	1646	85.78	81.57	83.62
6	EstNLTK 1.4.1	Korpus B3	1414	1752	1646	85.91	80.71	83.23
7	EstNLTK 1.6.5beta	Korpus A1	1146	2354	1699	67.45	48.68	56.55
8	EstNLTK 1.6.5beta	Korpus A2	1477	1802	1699	86.93	81.96	84.38
9	EstNLTK 1.6.5beta	Korpus A3	1477	1872	1699	86.93	78.90	82.72
10	EstNLTK 1.6.5beta	Korpus B1	1361	1975	1699	80.11	68.91	74.09
11	EstNLTK 1.6.5beta	Korpus B2	1502	1731	1699	88.40	86.77	87.58
12	EstNLTK 1.6.5beta	Korpus B3	1502	1766	1699	88.40	85.05	86.70
13	UDPipe 1.2.0.3	Korpus A1	950	2354	1535	61.89	40.36	48.86
14	UDPipe 1.2.0.3	Korpus A2	1243	1802	1535	80.98	68.98	74.50
15	UDPipe 1.2.0.3	Korpus A3	1248	1896	1535	81.30	65.82	72.75
16	UDPipe 1.2.0.3	Korpus B1	1148	1975	1535	74.79	58.13	65.41
17	UDPipe 1.2.0.3	Korpus B2	1250	1731	1535	81.43	72.21	76.55
18	UDPipe 1.2.0.3	Korpus B3	1261	1787	1535	82.15	70.57	75.92
19	StanfordNLP 0.2.0	Korpus A1	1152	2354	1725	66.78	48.94	56.48
20	StanfordNLP 0.2.0	Korpus A2	1493	1802	1725	86.55	82.85	84.66
21	StanfordNLP 0.2.0	Korpus A3	1498	1855	1725	86.84	80.75	83.69
22	StanfordNLP 0.2.0	Korpus B1	1379	1975	1725	79.94	69.82	74.54
23	StanfordNLP 0.2.0	Korpus B2	1501	1731	1725	87.01	86.71	86.86
24	StanfordNLP 0.2.0	Korpus B3	1519	1767	1725	88.06	85.96	87.00
25	Stanza 1.0.1	Korpus A1	1131	2354	1748	64.70	48.05	55.14
26	Stanza 1.0.1	Korpus A2	1463	1802	1748	83.70	81.19	82.42
27	Stanza 1.0.1	Korpus A3	1472	1862	1748	84.21	79.05	81.55
28	Stanza 1.0.1	Korpus B1	1348	1975	1748	77.12	68.25	72.41
29	Stanza 1.0.1	Korpus B2	1462	1731	1748	83.64	84.46	84.05
30	Stanza 1.0.1	Korpus B3	1481	1778	1748	84.73	83.30	84.00

Korpuse versioonide selgitus:

Korpus A1 - kõigi süntaktilise ja ortograafilise lause piiri annotatsioonidega korpus

Korpus A2 - kõigi ortograafilise lause piiri annotatsioonidega korpus

Korpus A3 - kõigi ortograafilise ja tuvastatud süntaktilise lause piiri annotatsioonidega korpus

Korpus B1 - enim märgitud süntaktilise ja ortograafilise lause piiri annotatsioonidega korpus

Korpus B2 - enim märgitud ortograafilise lause piiri annotatsioonidega korpus

Korpus B3 - enim märgitud ortograafilise ja tuvastatud süntaktilise lause piiri annotatsioonidega korpus

B2. Sõnestamise tulemused

Tabeli 1. Testitud sõnestajate tulemused

Nr	Sõnestaja	Korpus	Korrektsete sõnade arv	Korpuse sõnade arv	Sõnestaja tuvastatud sõnade arv	Täpsus	Saagis	F-skoor
1.	EstNLTK versioon 1.4.1	Korpus A ja B	26567	27159	27262	97,45	97,82	97,64
2.	EstNLTK versioon 1.6.5beta	Korpus A ja B	26869	27159	27381	98,13	98,93	98,53
3.	UDPipe 1.2.0.3	Korpus A ja B	26116	27159	26908	97,06	96,16	96,61
4.	StanfordNLP 0.2.0	Korpus A ja B	26197	27159	27136	96,54	96,46	96,50
5.	Stanza 1.0.1	Korpus A ja B	26313	27159	27217	96,68	96,89	96,78

Lisa C. Töös kasutatud kood

C1. Funktsioon „finalall1“

Lõigu haaval vaadati paralleelselt läbi algtekst ja annoteeritud versioonid ning annoteeritud versioonidest võeti välja kõik annotatsioonid ja paigutati need kokku algversiooni teksti. Funktsiooniga koostati osast A kolm koondversiooni. Sarnast koodi vajalike muudatustega kasutati ka osa B koondversioonide tegemiseks.

```
def finalall1(kirjutatav):
    # Sõnastikud märgendite statistika jaoks
    margendid = {}
    margendid1by1 = {}
    # Kõikide failide sisud loetakse listi
    sisu1 = loeridarealt('korpus/1_I.txt')
    sisu3 = loeridarealt('korpus/3_I.txt')
    sisu4 = loeridarealt('korpus/4_I.txt')
    algtxt = loeridarealt('algne/laused1.txt')
    # Kolm koondfaili kirjutatatakse paralleelselt lõik lõigu haaval
    # kirjutatav = laused1finalall.txt
    with open(kirjutatav, 'w', encoding='utf-8') as f:
        for i in range(len(sisu1)):
            # Lõigud, mis ei sisalda teksti kirjutatakse failidesse
            if sisu1[i].startswith("# newpar id") or sisu1[i] == "\n":
                # Kontrollin igaks juhuks üle, et ikka kõikides on samad read sama
                 sisuga
                if sisu1[i] == sisu3[i] == sisu4[i]:
                    f.write(sisu1[i])
                    finalall1by1('laused1finalall1by1.txt', sisu1[i])
                    finalall1by1most('laused1finalall1by1most.txt', sisu1[i])
                else:
                    f.write("#######ERROR - standardtekst ei kattunud kolmes
                    failis#######")
                            finalall1by1most('laused1finalall1by1most.txt',
                    "#######ERROR - standardtekst ei kattunud kolmes
                    failis#######")
                    finalall1by1('laused1finalall1by1.txt', "#######ERROR -
                    standardtekst ei kattunud kolmes failis#######")
                continue
            # Edasi töödeldakse ainult sisuga lõike
            # Vastav lõik tehakse sõnede listiks, mägrend jääb mõne sõne külge,
            sest tühikut ei võinud olla
            sisu1lst = sisu1[i].split()
            sisu3lst = sisu3[i].split()
            sisu4lst = sisu4[i].split()
            algtxtlst = algtxt[i].split()
            # Sõnastikku pannakse kõikidest failidest vaatlusall oleva lõigu
            sõned, milles on märgend.
            # Võtmeks on ennik (sõne, sõne indeks lõigus/listis, # esimese
            esinemise indeks) ja väärtuseks on selle esinemiskordade arv
            # Kui annoteerijad on need annoteerinud samamoodi, siis ta loendab
            need kokku.
```

```
# Kuna märgend oli ilma tühikuta alati mõne sõne küljes või sees, mis
oli olemas ka algfailis,
# siis oli nii kõige lihtsam märgendeid kindla asukohaga (sõnega
algtekstis) siduda.
# Märkide ja indeksite põhine lähenemine poleks sobinud, sest erinevad
annoteerijad lisasid erinevaid märgendeid
# (erineva pikkusega märgendeid) ja lõigud olid erinevates tekstides
erineva pikkusega (erinev arv märke).
# Lõigu sõned on listis alati sama indeksiga kõikides annoteeritud ja
annoteerimata tekstides.
margendiga = {}
# Tsükkel läbi lõigu sõnede listi. Kuna sõnede arv on kõigis sama,
siis võis valida suvalise neist.
for li in range(len(algtxtlst)):
    # Iga teksti puhul tuvastatakse sõned, mis sisaldavad märgendit
    if sisu1lst[li].count('#') >= 2:
        # Märgendiga sõne koos tema asukoha indeksiga lisatakse
        sõnastikku
        if (sisu1lst[li], li, sisu1lst[li].index('#')) not in
        margendiga:
            margendiga[(sisu1lst[li], li, sisu1lst[li].index('#'))] =
            1
        else:
            margendiga[(sisu1lst[li], li, sisu1lst[li].index('#'))] +=
            1
    if sisu3lst[li].count('#') >= 2:
        if (sisu3lst[li], li, sisu3lst[li].index('#')) not in
        margendiga:
            margendiga[(sisu3lst[li], li, sisu3lst[li].index('#'))] =
            1
        else:
            margendiga[(sisu3lst[li], li, sisu3lst[li].index('#'))] +=
            1
    if sisu4lst[li].count('#') >= 2:
        if (sisu4lst[li], li, sisu4lst[li].index('#')) not in
        margendiga:
            margendiga[(sisu4lst[li], li, sisu4lst[li].index('#'))] =
            1
        else:
            margendiga[(sisu4lst[li], li, sisu4lst[li].index('#'))] +=
            1
# Kui kolme lõigu kõik märgendiga sõned on sõnastikku lisatud, siis
saab nendest kokku panna
# esinemiste arvu sisaldava uue märgendi, mis lisatakse märgendamata
algteksti ja kirjutatakse faili laused1finalall.txt.
# Muutuja margendid sisaldab kõikide märgendite statistika üle 3
teksti.
lst = sorted(margendiga.items(), key = lambda x:x[0][1])
lst = sorted(margendiga.items(), key = lambda x:x[0][2], reverse=True)
for k,v in lst:
    #k[0] on sõne
    #k[1] on tema indeks listis
    #k[2] on # esinemise esimene indeks
    #v on mitu märgendajat on sellist märgendust kasutanud
    osad = k[0].split('#')
    algused = []
    summa = 0
    # Märgendite algusindeksid sõnes jäetakse meelde, tekstis oli
kohti, kus annoteeritud lause oli lause sees.
```

```
    for c in range(len(osad)):
        algused += [summa]
        summa += len(osad[c])
        if c%2 != 0:
            summa += 3
    for c in range(1,len(osad),2):
        m = '#' + osad[c] + str(v) + '#'
        algtxtlst[k[1]] = algtxtlst[k[1]][:algused[c]] + m +
        algtxtlst[k[1]][algused[c]:]
    for n in range(1,len(osad),2):
        if (osad[n], v) not in margendid:
            margendid[(osad[n], v)] = 1
        else:
            margendid[(osad[n], v)] += 1
uus = " ".join(algtxtlst)
f.write(uus+"\n")
#Teeme kohe ära ka selle variandi, kus on kõik märgendid ühe kaupa
loendatud
osad = uus.split("#")
#Korrutan kõiki märgendeid nende esinemise arvuga, st tekib kordustega
string
for c in range(1,len(osad),2):
    osad[c]=osad[c][:-1]*int(osad[c][-1])
osad = "#".join(osad)
#Kui kaks märgendit on kõrvuti, tekib olukord, kus ## on kõrvuti,
jätan need ära, et märgend saaks üheks
osad= osad.replace("##","")
#Jagan uuesti osadeks ja loendan iga märgendi kokku ning lisan uue
märgendina
osad = osad.split("#")
# Teen koopia finalall1by1most jaoks, sel juhul saab need kaks
versiooni paralleelselt ära teha
osadmost = copy.deepcopy(osad)
tyhjad = []
for c in range(1,len(osad),2):
    abi = ""
    counter = 0
    for el in osad[c]:
        if el in abi:
            continue
        mitu = osad[c].count(el)
        counter += osad[c].count(el)
        abi += el + str(mitu)
        if (el, mitu) not in margendid1by1:
            margendid1by1[(el, mitu)] = 1
        else:
            margendid1by1[(el, mitu)] += 1
        if counter == len(osad[c]):
            break
    osad[c] = abi
    if len(abi) > 2:
        abi=abi.replace("O1","")
        abi=abi.replace("S1","")
        abi=abi.replace("P1","")
    osadmost[c] = abi
osad = "#".join(osad)
osadmost = "#".join(osadmost)
osadmost = osadmost.replace("#O1#","")
osadmost = osadmost.replace("#S1#","")
```

```
            osadmost = osadmost.replace("#P1#","")
            finalall1by1('laused1finalall1by1.txt', osad+'\n')
            finalall1by1most('laused1finalall1by1most.txt', osadmost+'\n')
```

C2. Funktsioon „conllu“

```
def conllu (alg, margendet, sonestet):
    ## Algne - laused1/2.txt
    algne = loeridarealt(alg)
    ## Märgendatud tekst - laused1/2finalall.txt
    margendatud = loeridarealt(margendet)
    ## Märgendatud ja sõnestatud tekst - sonestatud1/2allmanual.txt
    sonestatud = loeridarealt(sonestet)
    CoNLLUfin = []
    par = ""
        ## Loendab lauseid
        counter = 1
    for i in range(len(algne)):
        if sonestatud[i] == "\n":
            continue
        if sonestatud[i].startswith("# newpar id"):
            CoNLLUfin += [sonestatud[i]]
            par = sonestatud[i]
            CoNLLUfin += ["# newpar_text = " + algne[i+1]]
            continue

        laused = sonestatud[i].split("#")
        lausedalgne = margendatud[i].split("#")
        for l in range(0,len(laused)-1,2):
            #tuleb lisada lause
            CoNLLUfin += ["# sent_id = " + par[14:].strip('\n') + 's' +
            str(counter) + '\n']
            CoNLLUfin += ["# text = " + lausedalgne[l] + '\n']
            CoNLLUfin += ["# label = " + "#" + lausedalgne[l+1] + "#" + '\n']
            soned = laused[l].strip().split(' ')
            for s in range(len(soned)):
                if s == 0:
                    CoNLLUfin += [str(s+1) + '\t' + soned[s].strip() +
                    '\t_\t_\t_\t_\t0\t_\t_\t_' + '\n']
                else:
                    CoNLLUfin += [str(s+1) + '\t' + soned[s].strip() +
                    '\t_\t_\t_\t_\t1\t_\t_\t_' + '\n']
            counter += 1
            CoNLLUfin += ['\n']
    CoNLLU = "".join(CoNLLUfin)
    return CoNLLU
```

C3. Funktsioon „Dice“

```
def Dice(sisu1, sisu2, margend):
    yhisosa = 0
    esimene = 0
    teine = 0
    for i in range(len(sisu1)):
```

```
        if sisu1[i].startswith("# newpar id") or sisu1[i] == "\n":
            continue
        sisu1lst = sisu1[i].split()
        sisu2lst = sisu2[i].split()
        for li in range(len(sisu1lst)):
            if sisu1lst[li].count('#') == 2 and sisu2lst[li].count('#') == 2 and
            margend in sisu1lst[li].split("#")[1] and margend in
            sisu2lst[li].split("#")[1]:
                yhisosa += 1
            if sisu1lst[li].count('#') == 2 and margend in
            sisu1lst[li].split("#")[1]:
                esimene += 1
            if sisu2lst[li].count('#') == 2 and margend in
            sisu2lst[li].split("#")[1]:
                teine += 1
    return 2 * yhisosa / (esimene + teine)
```

C4. Funktsioon „Fleiss“

Funktsiooniga saab arvutada Fleissi kappa juhtudel, kus kontrollitakse, kas sõne järel on süntaktilise või ortograafilise lause piir.

```
def Fleiss(conllu, annotarv, tahis):
    margend = ''
    sonesidkokku = 0
    Pikokku = 0
    kat0kokku = 0
    kat1kokku = 0
    n = annotarv
    for i in range(len(conllu)):
        if conllu1[i].startswith("# label"):
            a = 0
            m = ''
            margend = conllu[i].replace("# label = #","").replace("#","").strip()
            if not "§§" in conllu[i]:
                m = margend[:-1]
                a = m.count(tahis)*int(margend[-1])
            else:
                osad = margend.split("§§")
                for el in osad:
                    m += el[:-1]
                    a += el[:-1].count(tahis)*int(el[-1])
        if conllu[i][0].isnumeric() and conllu[i+1] != "\n":
            sonesidkokku += 1
            Pi = 1
            Pikokku += Pi
            kat0kokku += n
            soneveerg = conllu[i].strip().replace("\t"," ")
        elif conllu[i][0].isnumeric() and conllu[i+1] == "\n":
            sonesidkokku += 1
            if a == n:
                Pi = 1
                Pikokku += Pi
                kat1kokku += n
                soneveerg = conllu[i].strip().replace("\t"," ")
            elif a == 0:
```

```
                Pi = 1
                Pikokku += Pi
                kat0kokku += n
                soneveerg = conllu[i].strip().replace("\t"," ")
            else:
                kat1 = a
                kat1kokku += kat1
                kat0 = n - a
                kat0kokku += kat0
                Pi = (1/(n*(n-1)))*(kat0**2 + kat1**2 - n)
                Pikokku += Pi
                soneveerg = conllu[i].strip().replace("\t"," ")

    N = sonesidkokku
    Pikeskmine = Pikokku/N
    maxAnnotatstioonideArv = n*N
    p0 = kat0kokku/maxAnnotatstioonideArv
    p1 = kat1kokku/maxAnnotatstioonideArv
    Pe = p0**2 + p1**2
    k = (Pikeskmine - Pe)/(1 - Pe)
    return k
```

C5. Kood EstNLTK versiooni 1.6.5beta tulemuste saamiseks

Koodi kasutati Anaconda keskkonnas, kuhu oli installitud EstNLTK 1.6.5beta ja Python 3.6.

```
import estnltk
from estnltk.text import Text

tekst = loeridarealt("../algne/laused1.txt")
tekst += loeridarealt("../algne/laused2.txt")

conllu = ''
count = 0
for i in range(len(tekst)):
    if tekst[i] == "\n":
        continue
    if tekst[i].startswith("# newpar id"):
        conllu += tekst[i]
        continue
    conllu += "# newpar_text = " + tekst[i]
    txt = Text(tekst[i]).analyse('segmentation')
    laused = [s.enclosing_text for s in txt.sentences]
    for l in laused:
        count += 1
        conllu += "# sent_id = " + str(count) + '\n'
        conllu += '# text = ' + l + '\n'
        txt = Text(l).analyse('segmentation')
        soned = txt.words.text
        for i in range(len(soned)):
            if i == 0:
                conllu += str(i+1) + '\t' + soned[i].strip() +
                '\t_\t_\t_\t_\t0\t_\t_\t_' + '\n'
            else:
                conllu += str(i+1) + '\t' + soned[i].strip() +
                '\t_\t_\t_\t_\t1\t_\t_\t_' + '\n'
```

```
        conllu += '\n'
kirjutafaili('estnltk.txt', conllu)
```

C6. Kood UDPipe'i tulemuste saamiseks

Ufal UDPipe'i GitHubi lehelt [43] pärinev kood UDPipe kasutamiseks:

```
class Model:
    def __init__(self, path):
        """Load given model."""
        self.model = ufal.udpipe.Model.load(path)
        if not self.model:
            raise Exception("Cannot load UDPipe model from file '%s'" % path)

    def tokenize(self, text):
        """Tokenize the text and return list of ufal.udpipe.Sentence-s."""
        tokenizer = self.model.newTokenizer(self.model.DEFAULT)
        if not tokenizer:
            raise Exception("The model does not have a tokenizer")
        return self._read(text, tokenizer)

    def read(self, text, in_format):
        """Load text in the given format (conllu|horizontal|vertical) and return
        list of ufal.udpipe.Sentence-s."""
        input_format = ufal.udpipe.InputFormat.newInputFormat(in_format)
        if not input_format:
            raise Exception("Cannot create input format '%s'" % in_format)
        return self._read(text, input_format)

    def _read(self, text, input_format):
        input_format.setText(text)
        error = ufal.udpipe.ProcessingError()
        sentences = []

        sentence = ufal.udpipe.Sentence()
        while input_format.nextSentence(sentence, error):
            sentences.append(sentence)
            sentence = ufal.udpipe.Sentence()
        if error.occurred():
            raise Exception(error.message)

        return sentences

    def tag(self, sentence):
        """Tag the given ufal.udpipe.Sentence (inplace)."""
        self.model.tag(sentence, self.model.DEFAULT)

    def parse(self, sentence):
        """Parse the given ufal.udpipe.Sentence (inplace)."""
        self.model.parse(sentence, self.model.DEFAULT)

    def write(self, sentences, out_format):
        """Write given ufal.udpipe.Sentence-s in the required format
        (conllu|horizontal|vertical)."""

        output_format = ufal.udpipe.OutputFormat.newOutputFormat(out_format)
        output = ''
```

```
        for sentence in sentences:
            output += output_format.writeSentence(sentence)
        output += output_format.finishDocument()
        return output
```

Eelneva koodi rakendamiseks loodud kood:

```
def UDPipe(mudelinimi, tekstkoos, failinimi):
    model = Model(mudelinimi)
    conllu = model.write(model.tokenize(tekstkoos), "conllu")
    return conllu

conllu = UDPipe('estonian-edt-ud-2.5-191206.udpipe',tekstkoos,"udpipe_et_edt.txt")
osad = conllu.split('\n')
for i in range(len(osad)):
    if len(osad[i]) > 0 and osad[i][0].isnumeric():
        rida = osad[i].split('\t')
        if osad[i].startswith('1\t'):
            osad[i] = rida[0] + '\t' + rida[1] + '\t_\t_\t_\t_\t0\t_\t_\t_'
        else:
            osad[i] = rida[0] + '\t' + rida[1] + '\t_\t_\t_\t_\t1\t_\t_\t_'
kirjutafaili("udpipe_et_edt.txt", "\n".join(osad))
```

C7. Kood StanfordNLP tulemuste saamiseks

```
import stanfordnlp as stanford

tekst = loeridarealt("../algne/laused1.txt")
tekst += loeridarealt("../algne/laused2.txt")
#seda oli vaja üks kord käivitada mudeli allalaadimiseks
stanford.download('et')
nlp = stanford.Pipeline(processors='tokenize', lang="et", treebank="et_edt")

conllu = ''
count = 0
for i in range(len(tekst)):
    if tekst[i] == "\n":
        continue
    if tekst[i].startswith("# newpar id"):
        conllu += tekst[i]
        continue
    conllu += "# newpar_text = " + tekst[i]
    doc = nlp(tekst[i])
    for j,l in enumerate(doc.sentences):
        count += 1
        conllu += "# sent_id = " + str(count) + '\n'
        conllu += "# text = " + " ". join(["{}".format(word.text) for word in
        l.words]) + '\n'
        for token in l.tokens:
            if token.index == '1':
                conllu += token.index +'\t' + token.text +
                '\t_\t_\t_\t_\t0\t_\t_\t_' + '\n'
            else:
                conllu += token.index +'\t' + token.text +
                 '\t_\t_\t_\t_\t1\t_\t_\t_' + '\n'
        conllu += '\n'
kirjutafaili("stanfordnlp.txt", conllu)
```

Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Kairit Peekman,

1. Annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose „Automaatse lausestamise ja sõnestamise hindamine uue meedia keele korpusel“, mille juhendaja on PhD Kairit Sirts, reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni.

2. Annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 3.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni.

3. Olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile.

4. Kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Kairit Peekman

30.04.2020