TARTU ÜLIKOOL

MATEMAATIKA-INFORMAATIKATEADUSKOND

Arvutiteaduse instituut

Informaatika

Jane Jürgenson

Meetodite võrdlus automaatses õigekirja-vigade paranduses

Bakalaureusetöö (9 EAP)

Juhendaja(d): Margus Treumuth

Tartu 2015

Meetodite võrdlus automaatses õigekirjavigade paranduses

Lühikokkuvõte:

Selle töö käigus võrreldakse kolme erinevat õigekirjavigade paranduse meetodeid. Kirjeldatakse nende algoritme ja arendamist. Võrreldakse lisaks automaatset õigekirjaparandust ja kasutaja poolt valitavaid parandust üldiselt.

Võtmesõnad:

Õigekirjakontroll

Comparison of automatic spell correction methods

Abstract:

This paper compares three different automatic spell corrector methods. It also describes spell correctors algorithms and development of those algorithms. In the paper there is also a comparison of automatic spell correction and spell checker in general.

Keywords:

Spell checker, spell correction

Sisukord

1. Sissejuhatus 4
2. Õigekirjavea automaatne parandamine ja tuvastamine 7
2.1 Automaatparanduse vajadus 7
2.2 Automaatne versus ise valimine 8
3. Meetodid 11
3.1 OÜ Filosofti õigekirjaparandaja 11
3.1.1 Korpus 11
3.1.2 Eesti keele sõnade moodustamine 12
3.1.3 Õigekirjaparanduse arendamine ja tööpõhimõte 12
3.2 Asünkroonses dialoogsüsteemi raamistikus kasutatav õigekirjakontroll 15
3.3 Masinõppel loodud mittekeelepõhine õigekirjaparandaja 16
3.3.1 Lähenemisviis 17
3.4 Võrdlus 21
4. Kokkuvõte 25
5. Kasutatud kirjandus 26
Lisad 29
I. Litsents 29

1. Sissejuhatus

Tekstitöötluses on laialdaselt kasutusel keeletehnoloogiline tarkvara õigekirja kontrollimiseks ning vigaste sõnade korral parandusvariantide pakkumiseks kasutajale. Populaarseimad õigekirja kontrollijad baseeruvad sõnastikel, mis sisaldavad korrektselt kirjutatud sõnu, mida võrreldakse kasutaja poolt kirjutatud sõnadega. On ka süsteeme, mis tuginevad dokumentide kogumikule, kus on toodud sõna õiged vormid ja nende sagedused. Teiste keeleliste allikate juurde kuuluvad veel erinevad loendid, mis koosnevad sageli valesti kirjutatud sõnadest, otsingu päringu logidest või valdkonnale omastest andmeallikatest. Õigekirjavead üldjoontes liigitatakse kahte gruppi, milleks on *tundmatu sõna viga* ja *teatud sõna viga*. Tundmatu sõna viga tähendab seda, et sellist sõna ei ole sõnastikus. Teatud sõna viga tähendab, et sõna on sõnastikus olemas, kuid mõeldi mingit muud sõna selle asemel. Näiteks kasutaja oli kirjutanud teksti kogemata sõna *selg*, kuigi oli mõelnud sõna *velg*. Selliseid vigu, kus sõna on kirjutatud õigesti, on automaatse õigekirjakontrolliga raske parandada.

Automaatses õigekirja paranduses on kasutatud erinevaid meetodeid, kus kasutatakse sõnastikke, reegleid, masinõpet, nende kombinatsiooni või mõnda muud meetodit.

Õigekirjakontrolli ja automaatset õigekirja parandajat kasutatakse peale tekstitöötluse veel optilises märgituvastuses, masintõlkes, dialoogsüsteemis [1] ja keeleõppes. Viimasel ajal on erinevad rakendused, nagu näiteks veebibrauserid ja e-posti kliendid lisanud juurde õigekirjakontrolli.

Esimesed õigekirjakorrektorid võrdlesid sisestatud sõnu leksikonis olevate korrektsete sõnadega. Tänapäevasemad õigekirjaparandajad on tunduvalt keerulisemad.

Teadustöö õigekirjakorrektori kohta ulatub tagasi aastasse 1957. Algne põhjus õigekirjakorrektori jaoks oli parandada andmete sisestamisel tehtud vigu. Esialgsem alus oli aga otsida ja parandada vigu, mis tulenesid konkreetse konteksti sisendseadmest. Leon Davidson [2] oli huvitatud lennureisijate nimede leidmisest, mille mõlemad nimed võisid olla valesti kirjutatud, kindla lennureisi jaoks. Genealoogilistes andmebaasides olevate nimede ja kohtade õigsusest tundis huvi Gary Carlson [3]. McElwain ja Evans [4] olid huvitatud süsteemi väljundi parandamisest, et see tunneks ära morse koodi.

Esimene õigekirjakontroll, mis loodi kui rakendus ja mitte kui uurimustöö, on ilmselt 1971 aastal Stanfordis Ralph Gorini poolt kirjutatud SPELL DEC PDP-10 jaoks [5]. See oli esi-

mene arvutiprogramm, mis oli spetsiaalselt kirjutatud tekstidokumentide õigekirja kontrolliks. See luges sisse faili ja tagastas väljundfaili, mille sisuks oli võimalikud valesti kirjutatud sõnad [6]. Iga sõna seejuures asus eraldi real. Erinevate täienduste lisamisel arenes välja rahvusvaheline *Ispell*[1], mis soovitas asendust valele sõnale ja toetas paljusid Euroopa keeli. Asenduse leidmiseks kasutab *Ispell* Levenshteini teisendamiskaugust [7], mille suuruseks on 1.

GNU Aspell'i[2] või lihtsalt *Aspell,* mis on peamiselt arendatud Kevin Atkinson-i poolt, eesmärgiks on asendada *Ispell.* Seda saab kasutada iseseisva programmina või teegina. *Aspell* võimaldab lihtsalt kontrollida ka dokumente, ilma spetsiaalset leksikoni kasutamata, mis on *UTF-8* kodeeringus. Samuti proovib see anda paremaid soovitusi valesti kirjutatud sõna jaoks.

Morfoloogiliselt rikka keele ja keeruliste liitsõnade või märgikodeeringu jaoks on olemas õigekirjakontroll ja morfoloogiline analüsaator *Hunspell*[3], mis algselt oli mõeldud ungari keele jaoks. Õigekirjakontrollimiseks kasutavad seda programmi *OpenOffice.org, Mozilla Firefox 3, Thunderbird, Chrome* ja paljud teised programmid.

1980-ndatel muutusid õigekirjakontrollijad kättesaadavaks personaalarvutitele ja neid integreeriti populaarsetele tekstitöötlusprogrammidele nagu WordStar ja WordPerfect. Õigekirjakontrolli Microsoft Wordi jaoks tutvustati esmakordselt Word 95-s.

Eestis algas eesti keele spelleri loomine aastal 1991 ning see on olnud tihedalt seotud eesti keele morfoloogiaanalüsaatori ESTMORF[4] arenguga. Mõlema aluseks on 36000-sõnaline leksikon ja kõikide sõnavormide moodustamiseks vajalikud reeglid. Enne seda oli lihtne masinloetav sõnastik, mis sobis käändsõnade jaoks. Vähe oli aga teada eesti keele kasutamisest reaalsetes tekstides. Samuti ei oldud viidud läbi katseid käsitlemaks eesti keele tuletusi ja koostamist arvutuslikult, kuigi need tuletus- ja liitsõnad moodustavad kuni 20% eestikeelsetest sõna märkidest [8]. Esimene versioon sellest õigekirjakontrollist avaldati aastal 1994.

[1] http://www.lasr.cs.ucla.edu/geoff/ispell.html

[2] http://aspell.net/

[3] http://hunspell.sourceforge.net/

[4] http://www.eki.ee/keeletehnoloogia/projektid/estmorf/estmorf.html

Eestis on ka teised inimesed tegelenud eesti keele õigekirjakontrollide loomisega ja vajalike sõnastike moodustamisega. Nendest tuntuim peale OÜ Filosofti õigekirjakontrolli on *Ispell*-i ja selle analoogide jaoks mõeldud leksikonid ja reeglid. Uuem sõnastik selle programmi jaoks valmis Jaak Pruulmanni poolt. Pruulmann-Vengerfelti [9] arvates on selle reeglistiku puuduseks see, et seal puudub liitsõnamoodustuse kirjeldus.

Töö eesmärk on anda ülevaade erinevatest automaatsetest õigekirjakontrollidest, tuues välja nende erinevused. Töö käigus võrreldakse samuti automaatset õigekirja parandust ja õigekirjakontrolli ning tuuakse välja nende eelised ja puudused.

Esimeses peatükis võrreldakse automaatset parandamist ja kontrollimist. Teine peatükk kirjeldab kolme erinevat meetodit, mida kasutatakse õigekirja kontrollimiseks mingis kindlas süsteemis või laialdaselt ning tuuakse välja nende meetodite erinevused.

2. Õigekirjavea automaatne parandamine ja tuvastamine

Selles peatükis võrreldakse õigekirjavea automaatset parandamist ja tuvastamist. Välja tuuakse automaatparanduse vajadused ning milliseid probleeme võib automaatne parandamine endas kujutada.

2.1 Automaatparanduse vajadus

Süsteemides, kuhu tuleb andmeid sisestada, on tihti oluline ka nende andmete korrektsus. Ebatäpsete andmete tõttu võib tekkida palju arusaamatusi, millel võivad olla mõjuvad tagajärjed. Mõnedel juhtudel on oluline ka see, et süsteem mõistaks sisendteksti, sest peab tegema järeldusi sellest.

Meditsiinis on väga oluline, et andmed oleksid korrektsed nii patsiendi ohutu ravi huvides, kui efektiivseks infoedastamiseks meditsiinitöötajate vahel ja uurimustööde [10] jaoks. Õigekirjavead ja vale informatsioon võib kaasa tuua valede ravimeetodeite rakendamist patsiendi ravis, mis võib omakorda panna ravitava inimese suurde ohtu. Segadust võivad eelkõige tekitada raviminimed, mis on sarnased nii kirjaviisi kui ka hääldus poolest [11]. Seega on oluline, et meditsiinitöötajate poolt sisestatud andmed oleksid korrektsed. Üha enam luuakse spetsiaalselt meditsiinitekstide jaoks automaatseid õigekirjavigade parandajaid [12] [13] ja kõne transkripteerimist [14], et muuta märkmete sisestamist kiiremaks.

Automaatset parandust rakendatakse samuti otsingumootorite päringutes. Tänapäeval üha rohkem kasutatakse interneti otsingumootoreid info leidmiseks, kuid päringu kirjutamisel satuvad sisse ka kirjavead. Laias laastus umbes 10-15% päringutest sisaldavad vigu [15]. Päringud on aga väljakutsed tavalistele õigekirjaparandajatele. Enamus päringutest koosnevad kas ühest mõistest või nende loendist ning sisaldavad õigustatud sõnu, mida tavalistest leksikonidest ei leia. Samuti on raske ära määrata, mis on kehtestatav päring, sest veebilehed sisaldavad ise ka õigekirjavigu. Kasutada ei saa ka lihtsat staatilist leksikoni, sest paljud uued sõnad, nimed ja kontseptsioonid muutuvad populaarsemaks iga päev (*Twitter, haveli, infomania, App*). Sellepärast oleks väga raske, kui mitte võimatu säilitada kõrge-katvusega sõnastikke. Viimasel ajal ongi seepärast uuringud keskendunud veebikorpuse ja päringulogide kasutamisele, et järeldada sealt teadmisi õigekirjavigade ja sõnakasutuse kohta, mida päringute tegemisel kasutatakse [16] [15] [17].

Google oli esimene suur otsingumootor, kes informeeris kasutajaid võimalikest õigekirjavigadest päringus ning pakkus parandust nendele [18].

Dialoogsüsteemid, mille käigus inimene küsib küsimusi süsteemi käest või vastab süsteemi küsimustele, kasutatakse mõnedel juhtudel automaatset õigekirjakorrektorit, et süsteem saaks aru, mida kasutaja tema käest küsib. See aitab omakorda süsteemil pakkuda paremat vastust kasutajale, kui sisestatud tekst on ilma kirjavigadeta. Intelligentses õpetamissüsteemis CIRCSIM-Tutor [19], mis on mõeldud meditsiinitudengite jaoks ja kus teostatakse loomuliku keele dialoogi kasutajaga, leiti arendamise käigus, et õpilased teevad infot sisestamisel palju vigu ja seega oli vaja juurde lisada automaatne õigekirjavigade parandaja, et süsteem saaks aru sisendtekstist. Üheks probleemiks, mida pidi selle süsteemi puhul veel lahendama, oli see, et tudengid ei kirjutanud tihti pikemaid sõnu lõpuni, vaid kasutasid lühendeid. Näiteks *specification* asemel kirjutati lihtsalt *spec.* Asünkroonse dialoogsüsteemi (ADS) raamistiku [1] arendamise käigus leiti, et enamus kasutajatest teevad sisestamisel kirjavigu võtmesõnade puhul ja seega rakendati seal õigekirjakorrektorit kasutajasisendi jaoks.

2.2 Automaatne versus ise valimine

Nii automaatsel õigekirjaparandajal kui ka ise valimisel on eeliseid ja miinuseid üksteise ees. Erinevates tekstitöötlusprogrammides saab kasutaja ise reguleerida, kas ta soovib, et tema vead automaatselt parandatakse või valib ta ise paranduse oma veale. OÜ Filosofti õigekirjakontroll [8] ja *Microsoft Word*[5] pakuvad võimalust ka kasutajal endal lisada sõnu leksikoni, et speller seda pärast enam veaks ei loeks. Peamised sellised sõnad on kohanimed ja inimeste nimed. *Microsoft Word* ja *Google Documents*[6] pakuvad lisaks veel võimalust kasutajal ise lisada sõnu, mida automaatselt parandada. See on hea võimalus kasutajale lisada sinna selliseid sõnu, milles ta kirjutamisel tihti eksib.

[5] https://products.office.com/en-us/word

[6] https://www.google.com/docs/about/

Mõnede programmide puhul on võimalik kasutada korraga nii automaatset õigekirja parandust kui õigekirja vea tuvastamist. Sellistel puhkudel parandab tekstitöötluse programm üldiselt ära kergemad õigekirjavead automaatselt ja raskema vea puhul annab kasutajale sellest märku ja pakub sobivaid variante selle korrigeerimiseks.

Kõikide õigekirjavigade leidmiseks ja nende parandamiseks ei saa ainult lootma jääda õigekirjavea kontrollijale, sest see ei pruugi tuvastada kõiki vigu, mida teksti kirjutamisel tehakse. Näiteks sõnad, mis on korrektselt kirjutatud, kuid kasutaja mõtles nende sõnade asemel teisi sõnu või sõnad, mis on valesti kirjutatud ja speller ei tuvasta neid ära. Speller võib märkida valeks ka sõnad, mis tegelikult on korrektselt kirjutatud ja need siis automaatselt valedeks sõnadeks ära parandada, sest need sõnad ei asunud tekstitöötlusprogrammi leksikonis. Sellist sündmust nimetatakse "Cupertino efektiks" [20].

Vanemates õigekirjakontrollide leksikonides oli olemas sõna *co-operation* (koostöö), kuid puudus sõna *cooperation*, ning see märgiti spelleri poolt veaks. Esimene pakkumine sellele veale ei olnud *co-operation* vaid Cupertino, mis on linn põhja Californias. Selle näite vigu võib leida rahvusvaheliste organisatsioonide veebilehekülgedest domeeniga ".int", nagu Põhja-Atlandi Lepingu Organisatsioon (NATO)[7], Euroopa Liit (EU) ja Ühinenud Rahvaste Organisatsioon (ÜRO). Õigekirjavigade parandajad ei asenda enam regulaarset sõna *cooperation* sõna Cupertino vastu, kuid see efekt on pidev teiste sõnade puhul. See nähtus mõjutab eriti võõrkeelseid sõnu, väljendeid, ebatavalisi pärisnimesid, kuid ka tavapäraseid sõnu, mis on valesti kirjutatud.

Tekstitöötluses kasutatavad õigekirjakontrollijad ei pruugi pakkuda ühtegi parandust raskesti valesti kirjutatud sõnale, järelikult ei saa ainult toetuda sellele, et speller oskab igakord anda paranduse varianti või variante. Samuti pakutavad variandid ei tarvitse olla sobivad parandused sellele valesti kirjutatud sõnale.

Eelis ise korrektse sõna valimisel automaatse parandaja ees on see, et kuna sõna võib ühe tähe muutmisel, kustutamisel, lisamisel või vahetamisel kõrval tähega tähenduse poolest täielikult muutuda, siis kasutaja saab ise valida pakutavate variantide seast selle, mida tema mõtles. Näiteks, kui kasutaja kirjutab *Microsoft Word* sõna *vis*, siis selle pakututavad asendused on *viss, viis, visa, vise* ja *vist*. OÜ Filosofti spellerit kasutades, on selle võimalikeks asendusteks *vs, bis, veis, viis, väis, võis, vöis, viss* jne. Nende näidete põhjal on kohe näha,

[7] http://www.nato.int/docu/speech/2001/s010319a.htm

kui palju võib sõna tähendus muutuda, ja kui kasutada automaatset korrektorit, siis nende paranduseks oleks kas *viss* või *vs*, kuigi näiteks kasutaja mõtles tegelikult sõna *viis*.

Mõlema viisi plussiks võib pidada, et see aitab kiiresti leida üles lihtsaid kirjavigu, mis vahepeal võivad kirjutamisel kahe silma vahele jääda, mis vähendab ajakulu tekstide kontrollimisel ja töötlemisel ning võib ära hoida nii mõnegi rumala vea.

Miinuseks ja probleemiks enamustel õigekirjakorrektoritel ja –kontrollidel on kindlasti see, et need ei arvesta ümbritsevat konteksti ja seega võivad pakkuda parandusteks valesid sõnu või vääralt sõna ära parandada. Samuti märgivad nad valeks sõnad, mis on teises keeles kirjutatud. Näiteks kirjutad inglise keeles teksti ja seal sees on mõned ladinakeelsed sõnad.

Nendel meetoditel on nii positiivseid kui negatiivsed külgi, kuid ei tohiks ainult sellele panustada, et need meetodid leiavad kõik kasutaja poolt tehtud vead üles. Seega, ei saa tekstitöötluses panustada ainult õigekirjakontrollidele ja –parandajatele kõikide väärate sõnade ülesleidmiseks, sest kõiki vigu ei leia kirjutaja vahel ka ise üles.

3. Meetodid

Selles peatükis antakse ülevaade kolmest erinevast õigekirjavigade parandajate meetoditest, milleks on OÜ Filosofti poolt arendatud õigekirjakontroll, asünkroonses dialoogsüsteemi raamistikus kasutatav õigekirjakontroll ja masinõppe abil loodud õigekirjakontroll.

3.1 OÜ Filosofti õigekirjaparandaja

Erafirma OÜ Filosofti poolt arendatud õigekirjakontroll kontrollib reeglite abil, kas sõna on õigesti kirjutatud ja lisaks kasutab leksikoni, et kontrollida keerulisemate sõnade korrektsust. Speller arendati paralleelselt morfoloogilise analüsaatoriga ESTMORF.

3.1.1 Korpus

Õigekirjakontrolli arendamise käigus kasutatud korpus koosnes teistest erinevatest korpustest, milleks on Eesti Kirjakeele korpus, Balti Võrgu-uudiste korpus ja Eesti Ajalehed korpus. [8]

Eesti Kirjakeele korpust hakati looma 1991 aastal Tartu Ülikoolis eesti keele laboratooriumis. See korpus disainiti järgides teiste korpuste printsiipe nagu Browni, Lancaster-Oslo/Bergeni ja London-Lundi. Tekstid, mida kasutati korpuse jaoks pärinesid aastatest 1983-1987, et saada kokku miljon sõna. Selle loomiseks kasutati umbes 500 teksti, mis koosnesid igaüks 2000 sõnast. Korpuse tekstid on jaotatud erinevatesse tekstiklassidesse (ilukirjandus, uudised jne) sarnaselt Lancaster-Oslo/Bergeni'le. Arendamise alguses otsustati, et Eesti Kirjakeele korpus tuleb märgistada (*tag*), et sellest oleks rohkem kasu keeleteadlastele. Märgistamsel kasutati TEI (*Text Encoding Inittative*) [21] suuniseid. Hiljem leiti, et märgistamine ei andnud mingeid eeliseid õigekirjakontrolli arendamisel. Arendamise algfaasis olid ilukirjanduslikud tekstid ja uudised peamised tekstiklassid Eesti Kirjakeele korpuse loomisel, koosnedes umbes 300000 sõnast. Sellel korpusel oli suurim mõju leksikoni sisule ja algoritmi disainile. [8]

Balti uudiste korpus, mis käivitati aastal 1994, koosneb uudistest, mida uudisteagentuur Balti Võrgu-uudised toodab. Tekstid, mida saadakse arhiveeritakse automaatselt ja iga aasta kasvab see korpus 3-4 miljoni sõna võrra. Iga uudise puhul märgendatakse ainult selle algus ja lõpp, mitte kogu tekst. See korpus koosneb paljudest õigekirjavigadest ja seda kasutati

spelleri kontrollimiseks toimetamata tekstide peal. Tüüpiliste vigade kindlaksmääramiseks, mida eesti emakeelerääkijad teevad, kasutati selle korpuse ühe kuu uudiseid, kust osaliselt käsitsi võeti dokumentidest välja 1000 valesti kirjutatud sõna spelleri arendamiseks. See aitas spelleril soovitada õigeid sõnavorme valesti kirjutatud sõnade asemele. Samuti see korpus oli hea ressurss pärisnimede, lühendite ja akronüümide jaoks. [8]

Eesti Ajakirjade korpus asutati 1993 aastal. Selle eesmärgiks on järgida keele muutust aja jooksul. Korpus koosneb erinevatest ajakirjade tekstidest, mis pärinevad perestroika ja iseseisvuse ajast. Aastast 1996 kasvab see korpus 4 miljoni sõna võrra aastas. [8]

3.1.2 Eesti keele sõnade moodustamine

Eesti keeles moodustatakse uusi sõnu käänamise, tuletamise ja liitmise käigus. Need moodustavad suure osa eesti tekstide sõnavormidest.

Eesti keele käänamine hõlmab sõnatüvedele liite lisamist ja vaheldumist sõnatüves endas. Sõna tuletamise käigus liidetakse sõna lõppu järelliide. Ühele sõnale saab järjest lisada mitu sufiksit. Tuletamise käigus kasutatakse ka eesliiteid, kuid tavaliselt neid ei kasutata järjestikuliselt. Sõnu moodustatakse veel liitmise abil, mille käigus sõnad liidetakse oma-vahel kokku. [8]

Sõna tüvesid, käändsõnu, modifitseerituid sõna tüvesid või tuletatuid sõnu, mis kuuluvad mingisse sõnaklassi, välja arvatud sidesõnad ja akronüümid, võib kokku liita, et moodustada uusi liitsõnu. Kõik kombinatsioonid ei ole aga lubatud. Reeglina ei ole liitsõnad verbide lõppvormid. Sellele reeglile esineb ka erandeid näiteks "abielluma". Ühtlasi on ka piirang tüvi sõnade arvule ning liiga pikkade ja kohmakate liitsõnade puhul kirjutatakse need sidekriipsuga.

3.1.3 Õigekirjaparanduse arendamine ja tööpõhimõte

OÜ Filosofti spelleri käigus võrreldakse sõna vorme lekseemikombinatsioonidega leksikonist. Võrdlus sisaldab ainult grammatilist sõnade võrdlust.

Peamised spelleri omadused on [8]:

1. Arvestab kirjakeelega, mitte räägitava keelega.

2. Leksikon sisaldab sõnade lihttüvesid, mis kuuluvad eesti keele põhisõnavarasse, kõige sagedamini kasutatavaid pärisnimesid, lühendeid ning akronüüme. Leksikon ei sisalda liitsõnu ja tuletisi, mis on moodustatud.
3. Tuletus- ja liitsõnu analüüsitakse algoritmilaselt, mis eemaldab vajaduse lisada enamik eesti keele tuletus- ja liitsõnad leksikoni ning see omakorda võimaldab sobivat analüüsida uutele tuletus- ja liitsõnadele.
4. Algoritm tuletiste ja liitsõnade analüüsimiseks on kavandatud selliselt, et see leiab kõige tõenäolisema kombinatsiooni antud sõna osadest.
5. Analüüs baseerub sõnastiku vaatamisest ja ei sisalda heuristikat. Kui sõna ei saa analüüsida determinismlikult, siis ei tehta ka teadmistele tuginevat hinnangut.

Õigekirjaparandus analüüsib sõna paremalt vasakule, sooritades liidete eemaldamist. See kasutab sellist analüüsi seetõttu, et tulla välja niinimetatud „musta kasti“ tööriistaga spelleri jaoks. Leksikoni põhjaks kasutati Ülle Viksi „Väike vormisõnastikku“. Arendamise käigus võttis palju aega vastamine küsimustele, milleks olid:

1. Kui piisav on leksikon käsitlemaks eesti keele sõnavara päris tekstides?
2. Kuidas peaks käsitlema tuletusi ja liitsõnu?
3. Millised märgid eksisteerivad eestikeelsetes tekstides peale tavasõnade ja kuidas peaks neid käsitlema? [8]

Nende küsimuste vastuste abil selgitati välja spelleri kasulikkus.

Arendamise algusest peale oli oluline samuti programmi töökiirus, sest programm loodi kaubanduslikul eesmärgil. Efektiivsema algoritmi kujundamisel oli abi korpustest. Arvesse võeti terviktekstides kindla struktuuriga olevate sõnade statistilisi omadusi. Nende omaduste analüüsimine aitas kaasa parema algoritmi loomisele. [8]

Spelleri jaoks on komplekt liste, millest suurim on tüvisõnade leksikon [8].

Õigekirjakontrolli arendamine algas aastal 1991, kui saadi Ülle Viksi „Väike vormisõnastik“ masinloetav versioon. „Väike vormisõnastik“ koosneb ligikaudu 36000 eestikeelsest lihtsõnast koos täiskirjeldusega sõnavormi paradigma loomiseks. [8]

Töö käigus uuriti igat struktuurimustrit eraldi ja alguses oldi piirangutega range, kuid testimise käigus selgus, et piiranguid tuleks lõdvendada. Iga kord, kui muudeti analüüsitavaid struktuurimustreid ja piiranguid, testiti algoritmi sama teksti peal, mis eelmisel korralgi. Kui

tundmatute sõnade arv oli piisavalt väike, korrati seda uute tekstide peal. Algoritmi parema töökiiruse saavutamiseks otsustati kirjutada programm selliselt, et see kõigepealt proovib kõige tõenäolisemaid sõna struktuurimustreid. [8]

Jaanuaris 1992 algas liit- ja tuletussõnade jaoks algoritmi arendamine ja see kestis kuni 1994 aastani, mille tulemusena sai valmis õigekirjakontroll [8].

Algselt eeldati, et lubatud on ainult kaheliikmelised liitsõnad. Keerulisemate liitsõnade analüüsimiseks lisati juurde kaks listi, kus tüvesid saab lisada sõna lõppu ja algusesse. Samuti lisati sõnastikku tuhandeid ebaregulaarseid liitsõnu. [8]

1994 aastal lisati leksikoni juurde nimesid, numbreid ja teisi mittesõnu, sest morfoanalüsaatorit ESTMORFi kasutati keelelistes uuringutes ja see ei tundnud paljusid sõnu ära.

Aja jooksul tehti palju teste ja leksikoni lisati uusi sõnu, millest paljud olid nimed. Samuti kustutati ka mõned sõnad lihtsõnade leksikonist, sest need olid vananenud sõnad või murdesõnad. [8]

Speller töötab liht- ja liitsõna jaoks erinevalt. Lihtsõna, mis võib olla tuletamata sõna või tuletis, puhul eemaldatakse kõigepealt sõna lõpust liide. Seejärel vaadatakse, kas sõna esimene pool leidub tüvisõnade leksikonis. Misjärel kontrollitakse, kas liide ja tüvisõna sobivad kokku. Näiteks sõna *rääkida* tüvi on *rääki* ja liide *da*. Sobivuse kontrollimine on oluline, et filtreerida välja sõnad nagu *rääkita*, mis koosneb normaalsest tüvest ja liitest, kuid on koos sobimatud. [8]

Liite demonteerimine algab kõige väiksematest võimalikest variantidest ja mida pikemad on liited, seda vähetõenäolisem on, et see kuulub mingi sõna juurde [8].

Tuletatud sõna õigekirja kontrollimine toimub lihtsõna jaoks sarnaselt, kuid seal on erinevad ees- ja järelliited. Kuna järelliiteid võib mõnel sõnal olla korraga mitu, siis kasutatakse seal listi, mis koosneb üle 100 erinevast järelliite paarist. On olemas ka sõnatuletusi, mis on lisatud leksikoni, et vältida kahtlaseid reegleid. [8]

Osad, mis vastavad järgnevatele reeglitele moodustavad eesliite listi [8]:

1. Komponent ei ole iseseisvalt sõna või omab selgelt teist tähendust ühendis.
2. Komponendi moodustamine mingist sõnast ei ole triviaalne.
3. Komponenti saab kasutada vabalt uute sõnade moodustamiseks.
4. See esineb küllalt sagedaselt.

Liitsõnade kontrollimiseks olevad reeglid ja piirangud on seotud kahe peamise omadusega [8]:

1. Komponentide arv sõnas
2. Komponentide enda omadused: on see tüvi või järelliide, millisesse sõnaklassi tüvi kuulub, mis on tüve viimased tähed jne.

Speller on kirjutatud C programmeerimiskeeles DOS ja UNIX süsteemis. Kompileerides seda programmi Microsoft Word 95-ga leiti, et see töötab sama kiiresti kui ingliskeele jaoks olev õigekirjakontroll. [8]

Võimalik on alla laadida OÜ Filosofti spellerit *LibreOffice.org*[8] ja *OpenOffice.org*[9] tekstitöötlusprogrammi versioonidele 3.0.1 ning uuematele ja *IBM Lotus Notes*[10] versioonile 8.5. Samuti on võimalik kasutada seda õigekirjakontrolli veebikeskkonnas.

3.2 Asünkroonses dialoogsüsteemi raamistikus kasutatav õigekirjakontroll

Asünkroonse dialoogsüsteemi (ADS) raamistiku [1] põhjal tehtud dialoogsüsteemi logisid uurides, leidis Treumuth, et 80% kasutajatest teevad põhilistes võtmesõnades kirjavigu. Kuna võtmesõnad on olulised sisestatud info mõistmiseks, siis ADS raamistiku üheks töövahendiks on ka lihtne õigekirjakontroll.

Selles raamistikus kasutatav õigekirjakontroll on loomulikust keelest sõltumatu selles mõttes, et sama algoritmi saaks edukalt kasutada nii eesti- kui inglisekeelse dialoogi korral. Lähenemisviis, mis seal on kasutusel, on sõnevõrdlus, mille käigus võrreldakse sõnu leksikonis olevate sõnadega ja arvutatakse välja nende sarnasuse tõenäosus. Õigekirjakontrolli käigus on rakendatud Jaro-Winkleri sarnasusalgoritmi [22] , et kontrollida kasutaja sisendit süsteemi sõnastikuga. Õigekirjakontrolli arengu käigus prooviti ka Levenshteini teisenduskaugust [23], kuid leiti, et see parandas mõned kasutaja poolt sisestatud õiged sõnad valeks, sest süsteem leidis oma sõnastikust piisavalt sarnase sõna. Probleem oli seoses esitähtedega. Näiteks, kui kasutaja sisestab sõna *häda* ja sõnastikus leidub sõna *mäda,* siis Levenshteini

[8] https://www.libreoffice.org/

[9] https://www.openoffice.org/

[10] http://www.ibm.com/ee/et/

algoritm tegi vahetuse. Arendamise ja testimise käigus seetõttu otsustati kasutada Jaro-Winkleri [22] meetodit, kus sõna esimene täht omab nii suurt kaalu, et eksimust esimeses tähes peetakse liiga suureks erinevuseks ning sarnasusskoor tuleb selle tõttu väga väike. [1]

Jaro-Winkleri [22] võrdleb kahe sõne sarnasust ja mida sarnasemad kaks sõne on, seda lähedasemad need on. Dialoogisüsteemis [1] on see implementeeritud funktsioonina, mis tagastab sarnasuse, mis kuulub vahemikku [0..1]. Sarnasuse skoor 0 tähendab, et ei ole mingit sarnasust ja 1 tähendab täpset vastet. Kasutaja poolt sisestatud sõnu võrreldakse süsteemis oleva sõnastikuga, ja kui sõnad on piisavalt sarnased, siis vahetatakse vigane sõna korrektse sõna vastu sõnastikust. Kui sarnasus on madal, siis automaatset parandust ei rakendata.

Sarnasuse lävendiks seati katsetamise põhjal süsteemis 0.912 [1]. See tähendab, et kui kasutaja poolt sisestatud sõna ja sõnastikus oleva sõna sarnasus on suurem või võrdne 0.912 ja sisestatud sõna ei võrdu leksikonis oleva sõnaga, siis vahetatakse need sõnad ära.

Süsteemi konteksti põhist sõnastikku värskendatakse iga päev kell 6.00. ADS raamistik automaatselt genereerib domeenisõnastiku baseerudes sõnadele mis vastavad regulaaravaldistele. Sõnad, mis on lühemad kui 6 tähemärki ei kontrollita, sest nende sõnade mitmetähenduslikkus on liiga suur ja seega ei lisata neid sõnastikku. Leksikon koosneb sõnadest, mis esinevad reeglites ja ei kontrollita sõnu, millest süsteem “ei saa aru“. Reeglid sisaldavad ainult neid sõnu, mida on vaja kinni püüda ja millest aru saada. [1]

3.3 Masinõppel loodud mittekeelepõhine õigekirjaparandaja

Masinõppel loodud mittekeelepõhine õigekirjaparandaja kasutab veebi (*World Wide Web*) "puhastamata korpust" (*noisy corpus*), kus järeldatakse teadmisi õigekirjavigade ja keelekasutuse kohta, et moodustada sõna n-grammi keelemudel, veamudel ja potentsiaalsete paranduste "puhastamata list" (*noisy list*), mis koosneb sagedasematest täheldatud oskussõnadest [16].

Tõenäosuse ja arvutilingvistika valdkonnas on n-gramm pidev järjestus n osade jaoks antud tekstis või sõnas. Osad võivad olla kas häälikud, silbid, tähed, sõnad või aluspaar vastavalt rakendusele. Näiteks sõna *maja* tähtede 2-gramm (bigramm) on *ma, aj, ja*. "Puhastamata

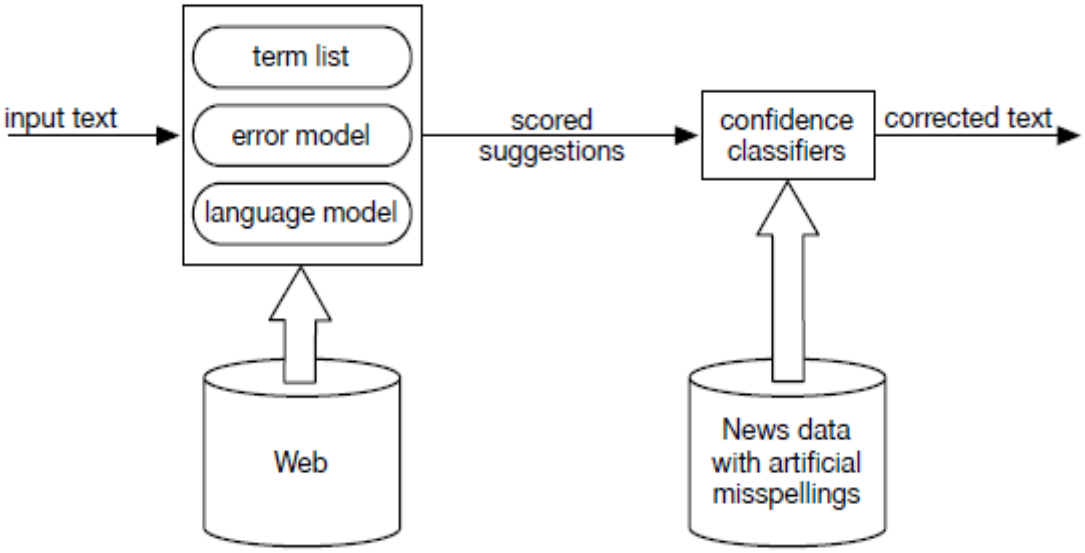
korpus" ja "puhastamata list" on automaatselt kogutud ja ei ole kuidagi filtreeritud ega käsitsi muudetud.

Veamudel põhineb alamstringide hindamisel. Seal rakendatakse veel teisejärgulist uudiste komplekti tehislikult sisestatud õigekirjavigadega, et reguleerida täpsuse klassifitseerijaid. Süsteemis ei kasutata käsitsi sisestatud annotatsioone või detailselt koostatud sõnastikku õigesti kirjutatud sõnadest. See mitmetasemeline süsteem integreerib oma teadmised statistilistest veamudelitest ja keele mudelitest koos statistilise masinõppe klassifikaatoriga. Igas etapis kasutati treenimiseks andmeid, et mehaaniliselt harjutada treeningmudeleid ja määrata klassifikaatorite kaalud. Mudeleid ja klassifikaatoreid treeniti automaatselt sagedusarvudega, mida saadi veebi ja uudiste andmetest. [16]

Täpsuse klassifikaatorid määravad õigekirjavea ja õigekirjakorrigeerimise lävendi vastavalt vea ja õppemudeli tulemustest. Neid klassifikaatoreid treeniti tekstidega, kus olid mõned valesti kirjutatud sõnad ja neile vastavad õigesti kirjutatud sõnad. Tekstideks kasutati väikest osa veebist saadud uudiste lehekülgede tekste. [16]

3.3.1 Lähenemisviis

Paranduste kanditaatideks koostati termide list, mis koosneb kümnest miljonist kõige sagedamatest veebis olevatest märkidest koos lihtsate filtritega mittesõnade (*non-words*) jaoks. See list peaks sisaldama kõige sagedasemaid õieti kirjutatuid sõnasid, suures hulgas mittesõnu (*non-word*) või õigekirjavigu.



Joonis 1 Õigekirjaparanduse protsess ja teadmiste allikad, mida kasutatakse [16]

Üleval oleval joonisel (Joonis 1) on üldjoontes kirjeldatud teksti töötlemise protsessi ja süsteemi ehitamiseks kasutatavaid andmeid. Iga sisestatud märgi jaoks võetakse oskussõna listist pakutavad soovitused ja hinnatakse neid veamudeli alusel. Pakutavaid soovitusi hinnatakse kontekstis, kasutades masinõpet ja järjestatakse uuesti. Iga märgi jaoks kasutatakse klassifitseerijaid, et kindlaks teha, kas sõna on valesti kirjutatud ja kui on, siis kas seda peaks parandama parima hinnangu saanud soovituse vastu.

Õigekirjasüsteem järgib õigekirjavigade segatud sidemudelit (*noisy channel model)* [24]. Vaadatava sõna jaoks *w* ja paranduse kanditaat *s* jaoks arvutatakse tõenäosus järgmiselt $P(s \mid w) = P(w \mid s) \times P(s)$. Kasutatakse alamsõne veamudelit, et hinnata $P(w \mid s)$-d. Veamudeli tuletamiseks võetakse kasutusele jaotus *R* , mis on *s* naaberalamsõne ja jaotus *T*, mis on *w* naaberalamsõne nii, et $|T| = |R|$. Jaotused on seega üks-ühele vastavuses. Alamsõnede pikkus on maksimaalselt 2. Hindamiseks rakendatakse Brill ja Moore [25] hinnangut $P(w \mid s)$ jaoks, mis on järgmine:

$$P(w \mid s) \approx \max_{R,Ts.t|T|=|R|} \prod_{i=1}^{|R|} P(T_i \mid R_i)$$

ja maksimaalset tõenäosuse hindamist kasutatakse $P(T_i \mid R_i)$ hindamiskeks. [16]

Veamudeli treenimiseks kasutati kolmikut, kuhu kuulus mõeldud sõna, vaadeldud sõna ja kokku loetud arv, mis on koostatud veebist saadud andmetega. Need kolmikud on saadud

termide listist, kus protsess automaatselt tuvastab sellest listist oletatavad paarid õigesti kirjutatud ja valesti kirjutatud sõnade kohta koos nende esinemisarvuga. Neid kolmikuid ei kasutatud otseselt paranduste pakkumiseks ja kuna rakendati veel alamsõnede mudelit, siis ei pidanud kolmikud olema põhjalik list valesti kirjutatud õigekirjavigadest. Protseduur nende kolmikute leidmiseks ja uuendamiseks arvestab järgmisi punkte:

1. Õigekirjavead on tavaliselt kirjaviisilt sarnased kavatsetud sõnale.
2. Sõnad kirjutatakse tavaliselt nii nagu need on kavatsetud. [16]

Selle mudeli jaoks kasutati suurt korpust, mille maht ulatub kuni 3.7×10^8 leheküljeni, mis kõik on saadud avalikest veebilehtedest [16].

Sarnaste sõnade otsimiseks terminite listist kasutatakse Damerau-Levenshtein kaugusvahemikku [26] koos konservatiivse ülempiiriga, mis suureneb koos sõna pikkusega [16]. Kui sõna on kuni neli tähte pikk, siis selle teisendamiskaugus on 1. Sõnad, mille tähtede arv jääb vahemikku, siis selle maksimaalne teisendamiskauguste arv võib olla 2 ning pikemate sõnade puhul on selleks arvuks 3. Näiteks sõna *mägi* ja *nägi* teisendamiskaugus on 1, sest vaja on ühte tähevahetust, et saada sõnast *mägi* sõna *nägi.* Damerau-Levenshteini algoritmis [26] on lubatud tähe kustutamine, lisamine, vahetamine ja kõrvuti olevate tähtede ümberreastamine. Efektiivseks otsinguks kõikide termide jaoks maksimaalse teisendamiskaugusega koostati termide loendist prefikspuupõhine andmete struktuur [16].

Filtreeriti ka sõnavormide listi nii, et esimese sõna sagedus on vähemalt 10 korda suurem kui järgmise sõna sagedus listis [16].

Viimasena kasutati konteksti, kus term esines, et koguda õigekirjavigade suunakaale. Selle käigus koguti korpusest konteksti hulkasid, kus term esines. Kontekstiks oli term koos ees ja taga asuva sõnaga. Need kontekstid, mida oli vähem kui 10 korda, jäeti kõrvale. Valiti ainult term ja temaga sarnased termid, mis esinesid vähemalt 10 korda sagedamini kui esialgne term valitavate termide hulgast. Valitavate termide hulgast valiti antud kontekstist kõige sagedamalt esinenud ehk kavatsetud term. Tulemuseks saadud andmete kogum koosnes kolmikutest – originaalne term, kavatsetud term ja kavatsetud termi esinemiste arv. Kokku on selles andmete kogumis umbes 100 miljonit sellist kolmikut. [16]

Selle treenimisprotsessi käigus leiti, et see andmete hulk sisaldab endiselt tähtsusetut müraga infot, sest ilmselgesti valesti kirjutatud sõnale ei pakutud asendust või pakuti mõnele

üksikule juhtumile. Vead, mida leiti andmete kogumist, sisaldasid kõige tavalisemaid õigekirjavigu. Andmete kogumis on ka näiteid õigesti kirjutatud sõnavigade kohta. [16]

$P(s)$ hindamiseks kasutati n-grammi keelemudeleid, mida treeniti andmetega veebist, kasutades rumaltaganemist (*Stupid Backoff)*. Veamudeli ühendamisel keelemudeli skooridega kasutati tõenäosust valemiga $P(w \mid s) * P(s)^{\lambda}$, kus keelemudelit kaaluti võttes sellest λ aste. Parameeter λ väljendab ka suhtelisi määrasid kuni milleni tuleks keelemudelit ja veamudelit usaldada. Veel mängib see parameeter täiendavat rolli parandades veamudeli valearvestuse määra, millal inimesed teevad vigu. Parameetrit λ treeniti keskmist pöördejärku optimeerides õigesti kirjutatud sõna järguga treeningkorpusest, kus järku arvutati üle kõigi soovituste, mis olid iga märgi jaoks. [16]

Esialgsel eksperimenteerimisel leiti, et süsteem ennustas liiga palju väärasid automaatparandusi, eriti lausete alguses ja lõpus. Selle käigus muudeti λ tingimusi võttes arvesse konteksti kättesaadavust hankides $\lambda_{i,j}$ väärtuse, kus *i* ja *j* väljendasid sõnast vasakul ja paremal kättesaadavat konteksti suurust keelemudeli jaoks. $\lambda_{i,j}$ treenimiseks jaotati andmed konteineritesse, mis vastasid paaridele *i* ja *j* ja igat $\lambda_{i,j}$ optimeeriti iseseisvalt. Treenimise käigus saadi λ väärtuseks 5.77. [16]

Õigekirjakontrolli ja automaatset parandajat implementeeriti kolmes jaos. Esimesena järjestati kõik soovitused *s* sõna *w* jaoks nende $P(s|w)$ tulemuse järgi. Teisena kasutati õigekirjakontrolli klassifitseerijat, et kontrollida, kas sõna *w* on õigesti kirjutatud. Kolmandana, kui *s* on mittetühi ja *w* on valesti kirjutatud, siis kasutatakse automaatse parandaja klassifikaatorit, et ennustada, kas parim soovitus valesti kirjutatud sõna jaoks on õige. [16]

Õigekirjakontrolli klassifikaatorit implementeeriti kasutades kahte seesolevat klassifitseerijat, millest ühte kasutati siis, kui *s* oli tühi ja teist siis, kui *s* oli mittetühi. Lihtsam klassifitseerija võrdleb $log(P(s|w)) - log(P(w|w))$ väärtust originaalse sõna *w* jaoks ja parima pakutud *s* jaoks koos lävendi väärtusega. Kui ei ole teisi pakkumisi peale *w*, siis ignoreeritakse $log(P(s \mid w))$ termi. [16]

Logistiline regressiooni klassifikaator kasutab viite tunnuse komplekti. Esimene komplekt on hindetunnus, mis ühendab järgneva punktisüsteemiinfo $log(P(s|w)) - log(P(w|w))$ edukaimaks soovituseks *s*, keelemudeli tulemuserinevused originaalse sõna *w* ja parima soovituse *s* vahel, $log(P(s|w|)) - log(P(w|w))$ paremuselt teine edukaim soovitus *s*, keelemudeli tulemuserinevused originaalse sõna *w* ja paremuselt teise edukaima soovituse *s*

vahel. Ülejäänud neli erinevat tunnuse komplekti kodeerivad informatsiooni juhunimemärkidest, saadaval olevate paranduste arv, sümboli pikkus ning vasaku ja parema konteksti suurusest. [16]

Koefitsiendi klassifikaatorite treenimiseks ja häälestamiseks kasutati kontrollitud andmeid (*supervised data*), mis olid paaride kujul, kus esimeseks liikmeks olid dokumendid, mis sisaldasid õigekirjavigu ja teiseks liikmeks dokumendid ilma õigekirjavigadeta. Selleks kasutati puhast korpust, sest eeldati, et robustsete klassifikaatorite treenimiseks on vaja suhteliselt "puhtaid" (*noiseless*) andmeid. Ilma õigekirjavigadeta dokumendid valiti veebist olevatest uudistest ja vigadega dokumendid koostati nende põhjal, sisestades kunstlikult nendesse õigekirjavigu keskmiselt 2 viga saja tähe kohta. Vead sisestati võrdse tõenäosusega ja nendeks olid tähe kustutamine, transponeerimine või juhuslikult valitud tähemärgi lisamisel samas dokumendis. Andmed jaotati veel omakorda kolmeks mittekattuvaks andmete kogumikuks: treenimise ja arendamise andmekogud, mida kasutati mehaanilisel treenimisel ja koefitsiendi klassifikaatorite häälestamisel ning testandmetekoguks, mida kasutati tulemuste hindamiseks. [16]

Meetodi lõpptulemuseks võib olla mitu varianti, milleks on :

- Valesti kirjutatud sõna parandati automaatselt sobiva sõna vastu.
- Valesti kirjutatud sõna parandati valesti ära.
- Valesti kirjutatud sõna ei parandatud ega märgitud ära.
- Valesti kirjutatud sõna märgiti ära, kuid seda ei parandatud automaatselt.
- Õigesti kirjutatud sõna on parandatud valeks.
- Õigesti kirjutatud sõna on ära märgitud kui valesti kirjutatud sõna.

Eksperimentide käigus leiti, et loodud õigekirjakorrektor on sama hea või mõnel juhul isegi tunduvamalt parem kui *GNU Aspell*. Meetodit testiti saksa, inglise, araabia ja vene keelega. [16]

3.4 Võrdlus

Need kolm meetodit valiti võrdlemiseks selle poolest, et nad kõik on mingi osa suhtes sarnased oma algoritmis, kuid nendes meetodites on ka suuri erinevusi. Samuti erinevad nende otstarbed mingil määral üksteisest.

OÜ Filosofti [8] meetod koosneb peamiselt leksikonist ja reeglitest, millega otsustatakse sõna õigsus. ADS raamistiku õigekirjakorrektoris [1] kasutatakse samuti leksikoni ja lisaks Jaro-Winkleri algoritmi [22], et hinnata kui sarnased kaks sõna omavahel on ja siis vastavalt tõenäosusele otsustada, kas teha parandus või mitte. Mittekeelepõhises õigekirjakontrollis [16] kasutatakse samuti leksikoni sõna õigsuse kontrollimiseks, kuid valesti kirjutatud sõna puhul rakendatakse seal Damerau-Levenshteini teisenduskaugust [26], et selle abil leida valesti kirjutatud sõnale sobiv parandusvariant. Üheks suureks osaks on seal ka veamudel, kus on ära toodud sõnad koos nende variatsioonidega, mis sisaldavad kirjavigu. Oluliseks etapiks on ka klassifitseerijad, mis hindavad sõna õigsust. See meetod arvestab veel konteksti, et tuvastada, kas sõna sobib teksti.

Võrreldes OÜ Filosofti [8] ja masinõppel loodud mittekeelepõhist meetodit [16], siis sarnasus nende vahel peale leksikoni on see, et mõlemad kasutasid arendamise käigus korpust, et moodustada vajalik leksikon ja OÜ Filosofti spelleri jaoks veel reeglid. Korpuste suurus, mida aga kasutati nende meetodite arendamisel erinesid mitmeid kordi, sest mittekeelepõhisel meetodil oli võimalik kasutada internetist saadud tekste suuremas koguses. Erinevuseks nende kahe meetodi puhul on lisaks see, et OÜ Filosofti õigekirjakontrolli saab rakendada ainult eesti keelele, kuid mittekeelepõhist meetodit saab kasutada kõigi keelte peal. Üheks väikeseks sarnasuseks nende meetodite juures mingil määral on, et tuvastatakse ka uusi sõnu, mis aja jooksul tekivad. OÜ Filosofti spelleri puhul tuleb need sõnad lisada lihtsalt kasutajasõnastikku, et neid hiljem ära ei märgita kui vale sõna, kuid neid sõnu parandada ei osata, sest ei ole vastavaid reegleid selle jaoks ja seega ei tunta uute sõnade käändeid.

ADS raamistiku [1] ja mittekeelepõhine [16] õigekirjakorrektorid kasutavad õige sõna töötlemisel sõnavõrdluse meetodeid, kus esimese puhul leitakse kahe sõna sarnasus ja teise puhul teisendamiskauguste arv, mida läheb vaja, et ühest sõnast saaks teine sõna. Teise meetodi puhul arvestatakse lisaks veel sõna pikkust, sest sellest oleneb, mitu teisendamiskaugust on lubatud sõna töötlemisel. Sarnasus nende kahe vahel on lisaks see, et neid saab rakendada edukalt ka teiste keelte peal, kuid ADS puhul on nendeks keelteks peamiselt eesti- ja inglise keel. Samuti on mõlemad loodud eesmärgil, et süsteem saaks paremini aru, mida kasutaja mõtles, et pakkuda neile sobivaimat vastust või vastuseid, arvestades sellega, et kasutaja võis teha tekstis kirjavigu. ADS spelleri eelis mittekeelepõhise meetodi ees on see, et sõna töötlemisel ei arvestata seal sõna esitähte, sest tavaliselt kasutaja teab, mis tähega sõna al-

gab, kuid mittekeelepõhisel meetodil on lubatud esitähe muutmine. Mittekeelepõhise eeliseks ADS õigekirjakontrollija ees aga on, et see arvestab sõna pikkust, kui rakendatakse valesti kirjutatud sõnale teisendamiskaugust, sest pikemate sõnade puhul võib tulla sisse rohkem kirjavigu ja siis ei tarvitse olla piisav ainult ühe tähe lisamisest, vahetamisest, kustutamisest või transponeerimisest.

Õigekirjakontrollijate ADS [1] ja OÜ Filosofti [8] sarnasuseks on, et need mõlemad kasutavad reegleid, kuid ADS puhul kasutatakse neid selleks, et anda kasutajale sobiv tagasiside, mitte analüüsida, kas sõna tüvi sobib mingi teise tüve või lõpuga. Samuti on nende sarnasuseks veel leksikoni tarvitamine ja see, et mõlemad on loodud arvestades eeskätt eesti keelega. ADS õigekirjakontrolli käigus ei näe kasutaja, kas ta on sõna kirjutamisel teinud vea. See protsess toimub ainult süsteemi jaoks, et see saaks otsustada, kuidas kasutajale vastata, leides reeglite seast sobiva vastuse. OÜ Filosofti õigekirjakontrollija juures on kasutajal aga võimalik näha vigast sõna ja seejärel ise valida sobivat parandust.

Mittekeelepõhise meetodi [16] eelis teiste ees on, et see võtab arvesse lisaks kirjakeelele ka kõnelevat keelt, sest uusi sõnu tuleb juurde igapäevaselt, mida kasutatakse otsingu tegemistel ja seega on oluline, et ta oskaks neid ära tunda ja vajaduse korral parandada. Üheks plussiks on samuti korpuse suurus, mille pealt on mudeleid treenitud ning testimised samuti näitasid, et suurema korpusega on veamäär väiksem. Selle suurimaks eeliseks teiste ees on, et see ei nõua käsitsi annoteeritud ressursse ja oma keeleteadmised omandab see täielikult veebist.

OÜ Filosofti spelleri [8] pluss võrreldes teistega on, et seal on keskendutud ainult eesti keele õigekirjavigade parandamisele. Samuti õigekirjaparandaja reegleid ja leksikoni saab ühendada populaarsemate tekstitöötlusprogrammidega nagu *LibreOffice.org, OpenOffice.org, IBM Lotus Notes* ja *Microsoft Word* OÜ Filosofti lehelt[11].

ADS raamistiku õigekirjakorrektori [1] eeliseks teiste ees on see, et sõnade võrdlusel ei arveteta kasutaja poolt kirjutatud sõna esitähte, sest raamistiku arendamise käigus leiti, et vigu ei tehta sõna esimeses tähes ja seega ei arvestada seda sarnasusalgoritmis.

Nende kolme meetodi ühine eesmärk on eelkõige leida üles kasutaja poolt valesti kirjutatud sõna ja see võimalikult kiiresti ära parandada, et süsteem saaks aru millega tegemist on ja

[11] http://www.filosoft.ee/products/

vähendada kasutaja poolt tehtud kirjavigu tekstis. Lähenemisviis nende meetodite väljaarendamisel on olnud aga erinev, sest on keskendutud kindlale asjale, mida soovitakse teha. Samuti on lähenemisviise mõjutanud ressursid, mis arendamise käigus on olnud erinevad.

4. Kokkuvõte

Antud bakalaureusetöö eesmärgiks oli anda ülevaade erinevatest kaasaegsematest õigekirjakorrektoritest võrreldes nende algoritmide tööpõhimõtteid. Töö raames kirjeldati kolme erinevat meetodit ning toodi välja nende arendamisel esinenud probleeme.

Tulemusena on toodud välja nende kolme meetodi erinevused ja sarnasused algoritmides üksteise suhtes. Kirjeldatud on ka üldiseid negatiivseid ja positiivseid külgi, mis võivad esineda automaatset õigekirjakorrektorit kasutades. Välja on toodud ka eelised, mis automaatsel õigekirjaparandajatel ja õigekirjakontrollijatel on üksteise suhtes.

5. Kasutatud kirjandus

[1] M. Treumuth, A Framework for Asynchronous Dialogue Systems: Concepts, Issues and Design Asspects, Tartu: Tartu Ülikooli Kirjastus, 2011.

[2] L. Davidson, „Retrieval of Misspelled Names in an Airlines Passenger Record System," *Communications Of The ACM,* kd. 5, nr 3, pp. 169-171, 1 March 1962.

[3] G. Carlson, „Techniques for Replacing Characters that are Garbled on Input," %1 *Proceedings of the 1966 Spring Joint Computer Conference,* 1966.

[4] C. K. McElwain ja E. M. B., „The degarbler—A program for correcting machine-read morse code," *Information and Control,* kd. 5, nr 4, p. 368–384, 1962.

[5] J. L. Peterson, "Computer Programs for Detecting and Correcting Spelling Errors," *Communications of the ACM,* vol. 23, no. 12, pp. 676-687, 1980.

[6] K. Homik ja D. Murdoch, „Warch Your Spelling!," *The R Journal,* kd. 3, nr 2, pp. 22-28, 2010.

[7] V. I. Levenshtein, „Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics Doklady,* kd. 10, nr 8, pp. 707-710, 1966.

[8] H.-J. Kaalep, "An Estonian Morphological Analyser and the Impact," *Computers and the Humanities 31,* p. 115–133, 1997.

[9] J. Pruulmann-Vengerfeldt, Praktiline lõplikel automaatidel põhinev eesti keele morfoloogiakirjeldus, Tartu, 2010.

[10] Ö. Uzuner, I. Solti ja E. Cadag, „Extracting medication information from clinical text," *Journal of The American Medical Informatics Association,* kd. 17, nr 5, pp. 514-518, 2010.

[11] B. L. Lambert, „Predicting look-alike and sound-alike medication errors," *American Journal of Health-System Pharmacy,* nr 54, pp. 1161-1171, 15 Mai 1997.

[12] K. H. Lai, M. Topaz, F. R. Goss ja Z. Li, „Automated misspelling detection and correction in clinical free-text records," *Journal of Biomedical Informatics,* 2015.

[13] H. D. Tolentino, M. D. Matters, W. Walop, B. Law, W. Tong, F. Liu, P. Fontelo, K. Kohl and D. C. Payne, "A UMLS-based spell checker for natural language processing in vaccine safety," vol. 7, no. 1, 2007.

[14] A. Paats, T. Alumäe, E. Meister ja F. Ivo, „Evaluation of Automatic Speech Recognition Prototype for Estonian Language in Radiology Domain: A Pilot Study,“*16th Nordic-Baltic Conference on Biomedical Engineering*, Gothenburg, 2014.

[15] C. Silviu ja B. Eric, „Spelling correction as an iterative process that exploits the collective knowledge of web users,“*Proceedings of EMNLP 2004*, Barcelona, 2004.

[16] C. Whitelaw, B. Hutchinson, G. Y. Chung ja G. Ellis, „Using the Web for Language Independent Spellchecking and Autocorrection,“*Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore, 2009.

[17] J. Gao, X. Li, D. Micol, C. Quirk ja X. Sun, „A Large Scale Ranker-Based System for Search Query Spelling Correction,“*Proceedings of the 23rd International Conference on Computational Linguistics*, Beijing, 2010.

[18] S. Brin ja L. Page, „The Anatomy of a Large-Scale Hypertextual Web Search Engine,“ *Computer Networks and ISDN Systems*, kd. 30, nr 1-7, pp. 107-117, 1998.

[19] M. W. Evans, S. Brandle, R.-C. Chang, R. Freedman, M. Glass, Y. H. Lee, L. S. Shim, C. W. Woo, Y. Zhang, Y. Zhou, J. A. Michael ja A. A. Rovick, „CIRCSIM-Tutor: An Intelligent Tutoring System Using Natural Language Dialogue,“*12th Midwest AI and Cognitive Science Conference*, Oxford , 2001.

[20] B. Zimmer, „Language Log,“ 9 Märts 2009. [Võrgumaterjal]. Available: http://itre.cis.upenn.edu/~myl/languagelog/archives/002911.html. [Kasutatud 5 Mai 2015].

[21] Guidelines, „Guidelines for Electronic Text Encoding and Interchange (TEI P3),“ Oxford, 1994.

[22] W. W. Cohen, P. Ravikumar ja S. E. Fienberg, „A Comparison of String Distance Metrics for Name-Matching Tasks,“*Proceedings of KDD-2003 Workshop on Data Cleaning and Object Consolidation*, 2003.

[23] P. E. Black, „National Institute of Standards and Technology (NIST),“*Dictionary of Algorithms and Data Structures*, 2005.

[24] M. D. Kernighan, K. W. Church ja W. A. Gale, „A Spelling Correction Program Based on a Noisy Channel Model,“*Proceedings of the 13th conference on Computational linguistics*, Stroudsburg, 1990.

[25] E. Brill ja R. C. Moore, „An Improved Error Model for Noisy Channel Spelling Correction,“*Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, Stroudsburg, 2000.

[26] F. J. Damerau, „A technique for computer detection and correction of spelling errors,“ *Communications of the ACM,* kd. 7, nr 3, pp. 171-176, 1964.

[27] T. Bants, A. C. Popat, P. Xu, F. J. Och ja J. Dean, „Large Language Models in Machine Translation,“*Association for Computational Linguistics*, Prague, 2007.

[28] B. Martins ja M. J. Silva, „Spelling Correction for Search Engine Queries,“*Advances in Natural Language Processing*, Alicante, Springer Berlin Heidelberg, 2004, pp. 372-383.

Lisad

I. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina **Jane Jürgenson** (sünnikuupäev: 29.03.1993)

annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose

Meetodeite võrdlus automaatses õigekirjavigade paranduses, mille juhendaja on Margus Treumuth, reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

1.1. üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace´i kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.
3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, **14.05.2015**