Tallinna Reaalkool

Eesti keele kirjavahemärkide automaatkorrektuuri mudeli treenimine

Uurimistöö

Kaur Ilison

141.a, reaal-programmeerimise õppesuund

Juhendaja: õp Jaagup Kippar

Tallinn 2025

Sisukord

Sissejuhatus....3
1. Grammatika automaatkorrektuur....5
1.1. Korrektuuride tüübid....5
1.2. Kasutusvaldkonnad....6
1.3. Korrektuurivahendite arendus Eestis ja hetkeolukord....6
1.4. GLEU skoor....7
2. Korrektuuri lahendus....9
2.1. Metoodika....9
2.2. Töökäik....9
2.3. Treeningandmed ja nende puhastamine....11
2.4. Treenimine ja esinenud piirangud....13
2.4.1. Andmete eeltöötlemine....13
2.4.2. Mudeli treenimine....14
2.4.3. Treenimisel esinenud probleemid....15
3. Tulemused....17
3.1. Eksperimentide ülevaade....17
3.2. Keeletase A2 tulemused....18
3.3. Keeletase B1 tulemused....19
3.4. Keeletase B2 tulemused....21
3.5. Keeletase C1 tulemused....22
3.6. Koondtulemused....23
3.7. Edasised kasutus- ja arendusvaldkonnad....24
Kokkuvõte....26
Kasutatud allikad....27
Lisad....29
LISA 1. Andmete puhastamise kood....29
LISA 2. Fairseqi keskkonna seadistamine....32
Resümee....34
Abstract....35
Kinnitusleht....36

Sissejuhatus

Käesolev töö käsitleb eesti keele kirjavahemärkide automaatkontrolli rakendamise võimalusi, kasutades selleks masinõpet. Viimaste aastate jooksul on toimunud suur läbimurre keeletehnoloogia valdkonnas. Seda tõendavad suured keelemudelid, mis võimaldavad inimestel algoritmi abil luua uut teksti soovitud teemal, vastata küsimustele, saavutada õppeedukusi, parandada teksti ning jäljendada muid inimmõtlemist vajavaid tegevusi. Näitena võib tuua teksti automaattõlke, mis võimaldab sirvida reaalajas tõlgitud võõrkeelseid veebilehti. Laialdaselt on levinud ka kõnetuvastus, mis võimaldab filmidele lisada automaatseid subtiitreid, reaalajas transkribeerida keelt või aidata vaegkuuljatel lugeda teiste räägitud teksti.

Eesti keel on maailmas unikaalne, sest sellel on väga vähe rääkijaid. 2021. aasta rahvaloenduse põhjal kõneleb kõikidest Eesti elanikest (vanust ja kodakondsust arvestamata) 83,3% eesti keelt ehk 1 145 114 inimest (Tiit 2024). Maailma rahvastikust moodustub see 0,0145%. On selge, et eesti keele arengu eest vastutavad peamiselt Eesti kodanikud. Eesti keele ja kultuuri säilimine on ka lisatud Eesti põhiseaduse üheks põhiliseks eesmärgiks (Põhiseadus).

Käesoleva töö eesmärk on treenida masinõppe mudeleid, mis suudavad korrigeerida erinevate keeletasemete lausetes esinevaid kirjavahemärgivigu, ning hinnata selle täpsust, kasutades GLEU skoori.

Uurimistöös on püstitatud kolm uurimisküsimust: “Kui efektiivne on eesti keele kirjavahemärkide korrigeerimise mudeli treenimine automaattõlke algoritmi põhjal?”, “Millised raskused tekivad grammatikakorrektuuri mudeli treenimisel?”, “Kas spetsiifiliselt treenitud grammatikakorrektuuri mudel sooritab grammatikavigade parandamist suurema täpsusega, kui suur keelemudel ChatGPT 4o?”

Uurimistöö koosneb kolmest osast. Esimeses osas kirjeldatakse kirjavahemärkide automaatkorrektuuri meetodeid ja kasutusvaldkondi. Teises osas käsitletakse mudeli treenimiseks kasutatavate andmete töötlemist, mudeli treenimist ning GLEU skoori. Kolmandas osas analüüsitakse treenitud mudeli täpsust GLEU skoori alusel.

Käesoleva töö allikateks on kasutatud erinevaid teaduslikke artikleid, eelnevaid teadustöid grammatikakorrektuuri valdkonnas ja avalikke koodinäiteid. Kõik tarkvara osad põhinevad avalikult kättesaadavatel koodinäidetel, millele autor on lisanud täiendused.

1. Grammatika automaatkorrektuur

Grammatiliste vigade parandamise (GEC *ingl. k Grammatical Error Correction*) ülesandeks on automaatselt tuvastada ja parandada grammatilised vead tekstis. GEC ei hõlma ainult grammatiliste vigade parandamist, näiteks morfoloogilisi vigu, aga ka ortograafilisi ja semantilisi vigu. Grammatikakorrektuuri valdkond on viimasel kümnendil näinud märkimisväärseid edusamme. Seda protsessi toetavad reeglipõhiste meetodite, statistiliste klassifikatsioonide, statistiliste masintõlke ja ka närvivõrkudel põhinevad masintõlkesüsteemid. (Bryant *et al.* 2022)

1.1. Korrektuuride tüübid

Esimesed grammatikavigu parandavad programmid ilmusid 1960-ndatel ning olid reeglipõhised ja kontrollisid vaid lihtsaid vigu, mis tulenesid koma, punkti või mõne muu kirjavahemärgi valesse kohta paigutamisest. Esimesed automaatkorrektuuri lahendused olid inglise keeles ja ei põhinenud tekstis toodud kontekstil (Yuan 2017). Reeglipõhised lahendused on küll lihtsamad ning on ka tänapäeval laialdaselt kasutuses, kuid olukorras, kus tekib kompleksne kirjakeelne probleem, võivad reeglipõhised meetodid pakkuda vääraid tulemusi. Esimesed kasutatavad grammatikavigu parandavad programmid ilmusid alates 2000-ndatest näiteks ETS's Criterion ja Microsofti ESL Assistant, mis põhinesid käsitsi koostatud reeglitel (Bryant *et al.* 2022).

Järgmise suure hüppe GEC programmides tõi esile närvivõrkudel põhinev 2017. aastal ilmunud Transformeri arhitektuur. Transformer arhitektuur on eriline selle poolest, et tõlkimisel võetakse arvesse igat sõna. Iga sõna arvesse võtmine säilitab lause algse konteksti, näiteks sõna "tee" saab eesti keeles tõlgendada mitut moodi, mis kõik sõltuvad lause kontekstist ning just transformeri arhitektuuriga mudel suudab seda konteksti hoida.

Hetkeseisuga (märts 2025) on kõige populaarsem kasutada GEC mudeli treenimiseks Transformers arhitektuuri ning selle põhjal ehitatud masintõlke algoritmi, kus masin õpib teksti korrigeerima nii nimetatud vigasest eesti keelest korrektsesse eesti keelde. Selline meetod on

efektiivne viis, et sooritada grammatilist vigadeparandust, kuna meetod suudab võtta arvesse kogu lause konteksti lause korrigeerimisel.

1.2. Kasutusvaldkonnad

Grammatika vigade parandamise mudelid on leidnud laialdast kasutust erinevates valdkondades üle kogu maailma. Need mudelid on spetsiaalselt loodud tuvastama ja parandama keelelisi vigu tekstides ning nende rakendusvõimalused ulatuvad haridusest kuni professionaalsete kirjastusteni. Keeleõppes, eriti võõrkeele õppimisel, aitavad need mudelid õppijatel arendada oma kirjutamisoskust, pakkudes kiiret ja täpset tagasisidet grammatiliste vigade kohta. Populaarsed rakendused, näiteks Grammarly ja Google Docs, kasutavad samuti GEC mudeleid, et pakkuda kasutajatele reaalajas abi, parandades grammatilisi- ja stiilivigu kirjutamise ajal. Kirjastustes ja sisuloome ettevõtetes kasutatakse neid mudeleid, et tagada loodud tekstide kõrge kvaliteet ja loetavus, vähendades toimetajate töökoormust. Sotsiaalmeedia platvormidel ja suhtlusrakendustes aitavad need mudelid parandada kasutajate kirjutamise täpsust, samas kui teadustekstide ja akadeemiliste väljaannete puhul tagavad need selge ja vigadeta esitluse. GEC mudelite laialdane rakendamine aitab oluliselt kaasa tekstide kvaliteedi tõstmisele ja suhtluse parandamisele erinevates kontekstides, kus keeleline täpsus on oluline.

1.3. Korrektuurivahendite arendus Eestis ja hetkeolukord

Esimesed eestikeelsed grammatikakorrektuuri lahendused ilmusid 2000-ndate alguses ning nende eesmärgiks oli kontrollida lausete korrektsust reeglite põhjal. Reeglipõhised lahendused olid kirjutatud Java programmeerimiskeeles ning peamiselt kontrollisid rangete reeglitega nii kirjavahemärkide kui ka ortograafilisi vigu. Kirjavahemärgivigade olemust kontrolliti sidesõna ees ja järel. Koma mittenõudvate sidesõnade puhul keelati koma nii sõna ees kui järel ja koma

nõudva sõna ette nõuti koma, välja arvatud juhul, kui ühes kahest sidesõnaga eraldatud osalausest puudus öeldis. (Liin 2008)

Alates 2017. aastast on hakatud kasutama grammatikakorrektuuri programmide arendamisel neurovõrkude põhiseid Transformers arhitektuuri kasutavadi mudeleid. Järgnevalt on esitatud enam levinud grammatika vigade parandamise meetodid:

1. Zero-Shot Machine translation (Johnson *et al.* 2017) on meetod, kus masinõppe mudel treenitakse tõlkima kahe keele vahel, näiteks eesti keelest soome keelde, kuid mudeli sisendiks antakse eestikeelne lause, mis tõlgitakse soomekeelse mudeliga tagasi eesti keelde (Korotkova *et al.* 2019).
2. Jadast-jadasse teisienduste (*seq2seq* ehk *Sequence to sequence*) mudel on närvivõrgu põhine masinõppe mudel, mida kasutatakse järjestikuste andmete töötlemiseks (Luhtaru *et al.* 2024). Algoritmi arhitektuuri eristab *encoder* ja *decoder* kiht. *Encoder* kihile antakse lause ning selle ülesanne on muuta lause fikseeritud suurusega kontekstivektoriks, mis muudetakse teise keele kontekstivektoriks transformer arhitektuuri põhjal. *Decoder* muudab kontekstivektori tagasi loetavaks tekstiks. (Geeks for Geeks 2024)
3. Grammatika kontrolli saab ka teostada suurte keelemudelite abil. See meetod on küllaltki täpne tänu väga mahukale andmemahule. Kahjuks on see meetod aga ebaefektiivne, kuna suur keelemudel on loodud teksti genereerimiseks ning mudeli kasutamine tekstis olevate grammatika vigade parandamise ajal sooritatakse ka palju muid arvutusi.

Eestis tegeleb keeletehnoloogiliste lahendustega Tartu ülikooli arvutiteaduste instituudi keeletehnoloogia õppetool. Keeletehnoloogia lahenduste populariseerimiseks on loodud koostöögrupp TartuNLP, mis on tegutsenud aastast 2017. TartuNLP tegeleb laialdaselt nelja kategooriaga: kõnesüntees, keele tõlge väikekeelte jaoks, nimeolemite tuvastaja ja grammatikakorrektuur, mis on kõige uuem keeletehnoloogiline valdkond. (TartuNLP 2025)

Keeletehnoloogiat uuritakse ka Tallinna Tehnikaülikoolis tarkvarateaduse istituudi keeltehnoloogia laboratooriumis, kus põhifookus on automaatne kõnetuvastus. Tallinna Ülikoolis tegeletakse keeletehnoloogiaga Digitehnoloogiate instituudis.

1.4. GLEU skoor

Käesolevas töös kasutatakse grammatikavigade parandamise mudelite täpsuse hindamiseks GLEU (*ingl. Generalized Language Evaluation Understanding*) skoori, mis on võetud avalikust koodinäitest (Sakaguchi 2018). GLEU skoor on välja arenenud BLEU (*ingl. Bilingual Evaluation Understudy*) skoorist, mis on mõeldud peamiselt masintõlke mudelite hindamiseks. BLEU skoori hindamise kasutamise mittesobivus on osaliselt tingitud väikesest, kuid olulisest erinevusest masintõlke ja ükskeelse teksti ümberkirjutamise vahel. Masintõlkes on tõlkimata sõna või fraas peaaegu alati masina poolt tehtud viga, kuid grammatiliste vigade parandamises peavad jääma algsed sõnad alles. Näiteks masintõlkes lause "*The sky is blue.*" tõlgitakse lauseks "Taevas on sinine." – kõik sõnad on tähekujult erinevad. Samas aga grammatika vigade tõlkimisel lause "Pood on kinni sest täna on pühapäev" tõlgitakse "Pood on kinni, sest täna on pühapäev." – suurem osa lauses esinenud sõnu on täpselt või peaaegu täpselt sama sõnastusega. Seetõttu on BLEU skoori algoritmi täiendatud, mis arvutab n-grammi täpsused sihtlausete peal, aga mis annab kõrgema kaalu nendele n-grammidele, mis on õigesti muudetud lähtelausete seast. Seetõttu eelistab GLEU skoor parandatud vigu, samas andes muutmata sõnadele õiged kaalud. (Napoles *et al.* 2015)

2. Korrektuuri lahendus

2.1. Metoodika

Käesolevas uurimistöös on kasutatud Facebooki emafirma Meta tehisintellektilabori poolt tõlkemudelite treenimiseks loodud teeki Fairseq (Fairseq 2025). Fairseqi abil treenitakse mudelit tuvastama erinevusi korrektse ja vigase eesti keele vahel, võimaldades sel viisil masinal õppida, kuidas kirjavahemärkide puudumine mõjutab lause struktuuri ja tähendust. Mudeli eesmärk on analüüsida vigast teksti ja taastada selles puuduvad kirjavahemärgid, et muuta see grammatiliselt korrektseks ja keelereeglitele vastavaks.

Fairseqi teek on väga mitmekülgne tööriist, mida saab kohandada paljude ülesannete jaoks, nagu teksti genereerimine, masintõlge ja teksti kokkuvõte. Fairseq toetab mitmeid mudeliarhitektuure, näiteks Transformer ja LSTM (*Long Short Term Memory*). Fairseqi üks peamisi eeliseid on asjaolu, et see pakub mitmesuguseid eelseadistusi ja mitmekeelseid töötlemise võimalusi, võimaldades teadlastel ja arendajatel keskenduda rohkem oma konkreetsetele rakendustele ja õppimisele, selle asemel, et kulutada aega andmetöötluse infrastruktuuri loomisele.

2.2. Töökäik

Töö autori esmane kokkupuude grammatikkorrektuuri valdkonnas toimus esmalt 2021. aastal, kui Tallinnas toimus Hüppelaua suvelaager, kus üks osalenud meeskond proovis lahendada grammatikakorrektuuri ülesannet reeglipõhise lahendustega. Kahjuks tehniliste piirangute tõttu ei arenenud nende lahendus piisavalt kiiresti. Inspireerituna varasemast kokkupuutest sai autor juhendajalt viite Tallinna Ülikooli bakalaureusetööle (C.E Hindremäe, 2024), mis lahendas sarnast ülesannet grammatikakorrektuuri valdkonnas. Inspireerituna väljapakutud lahendusest jäljendas autor Hindremäe töös kasutatud metoodikat ja Fairseqi põhist lähenemist.

Esmaste katsete käigus kasutati eesti keele koondkorpusest ühe korpuse andmeid, mille põhjal tehti esmane testimine (Tartu Ülikool 2018). Selleks võeti eesti keele koondkorpusest ligikaudu 100 tuhat lauset, mida kasutati mudeli treenimiseks. Teise katse käigus suurendati andmete mahtu 500 tuhande lauseni, kuid see põhjustas treenimiseks kasutatud videokaardi mälu üleksautuse. Seejärel vähendati andmemahtu 200 tuhande lauseni, mida kasutati esimese suurema mudeli treenimiseks.

Järgnevate katsetustel võeti kasutusele Estonian National Corpus 2019 korpus (Estonian national corpus 2019 2020). Peamiseks põhjuseks oli andmete korrektsus ja keele dialekti parem sobivus - varasem korpus sisaldas peamiselt ajaleheväljaannete tekste, mille dialekti ei kasutata igapäevaselt, ning kommentaare, mille grammatiline korrektsus oli madalam kui keskmisel keskkooliõpilasel.

Uuete andmetega mudeli treenimine parandas tulemusi oluliselt ning seda mudelit kasutati ka käesoleva töö järgnevates ettappides. Kuigi mudel saavutas varasemaga võrreldes paremaid tulemused, esines selles veel ulatuslikult vigu. Mudel sobis lihtsama taseme lausete parandamiseks. Mudeli täpsuse suurendamiseks puudus siiski sobiv lahendus: andmemahu suurendamine oleks nõudnud oluliselt suurema mälu ja arvutusvõmekusega graafikakaarti.

Järgnevalt teostati mitmeid katsetusi Fairseqi dokumentatsioonis välja pakutud lahendustena. Näiteks prooviti andmete jagamist juppideks. See võimaldas treenida mitmeid kordi sama mudelit, asendades iga kord andmeid. Treeningprotsessi jooksul jõuti olukorrani, kus mudelit treeniti vaid ühe alamhulga piires, pärast mida masin treenimise pooleli jättis. Probleemi lahendamisel selgus, et rohkem kui ühe alamhulgaga treenimine polnud võimalik, kuna iga alamhulgaga loodi mudelile uus kuju, mis jäi konstantseks terve treenimise protsessi jooksul.

Uurimistöö parimaks riistvara piirangutega kokkusobivaks meetodiks osutus treenimise parameetrite muutmine. Mille puhul muudeti treenimiseks lubatud iga etapi mälu kasutust ja andmete mahtu.

Peale mudelite treenimist analüüsiti mudelite täpsust valitud test andmetega ja võrreldi ChatGPT 4o ja TartuNLP mudelitega, mille tulemused on kirjeldatud järgmistes peatükkides.

2.3. Treeningandmed ja nende puhastamine

Urimistöös treenitud mudelid kasutavad Estonian National Corpus 2019 võetud andmeid, mis koosnevad erinevatest keeletasemetega keeleliselt korrektsetest lausetest (Metashare 2020). Mudeli treenimise eelduseks on korrektsed treeningandmed. Suurest teksti korpusest võetud andmetega tekkis kaks peamist probleemi.

1. Kogu korpuse andmemaht oli 1,6 Gb teksti. Kahjuks ei olnud võimalik panna kogu teksti korraga treenima, kuna selleks oleks olnud vaja suurema mälumahuga graafikakaarti, mida töö kirjutamise hetkel ei olnud saadaval. Selleks, et korpuse andmeid kasutada, oli vaja andmeid töödelda väiksemate osadena. Algselt alustati testimist 50 Mb ning edasi prooviti ka kasutada 100 Mb, mis veel mahtus graafikakaardi mällu. Järgmine katsetus tehti 200 Mb, mis kahjuks ei mahtunud enam graafikakaardi mällu.
2. Palju raskust tekitas ka andmete õige kuju. Kõik andmed pidid olema õige kujuga ja asuma õigel kohal, mis tähendas seda, et nii õige tekstiga failis kui ka vale tekstiga failis pidi asuma lause mõlemas failis samal real. Selleks, et saada lauseid õigetele kohtadele, tuli andmeid puhastada, mis hõlmas järgnevaid samme:
    a. Korpusest võetud teksti algne kuju oli järgnev:

        <s>

        Piirkonnaarhitekt Alice Laanemägi ütles, et OÜ Maranello Vara on taotlenud linnalt selle ala detailplaneeringu koostamist, et ehitada sinna tulevikus Grand Hotel Tallinna laiendus, veepark ja parkla.

        </s>

Iga lause oli pandud <*s*><*/s*> vahele. Selleks et lausete korrigeerimisel vähendada tundmatute sõnade määra, tuli eemaldada need märgendid lausetest.

b. Eelmainitud süntaksi eemaldamisel tekkis olukord, kus tekstifaili tekkisid tühjad read. Nendest vabanemiseks tehti uus Pythoni kood, mis vabanes neist ja kirjutas iga lause eraldi reale.
c. Kõikidelt lausetelt oli vaja veel eemaldada kõik kirjavahemärgid, selleks koostati uus Pythoni programm.

Enne mudeli treenimist tuleb andmed eeltöödelda. Eeltöötlemise korral arvutab masin <*unk*> korduste arvu protsendi kogu teksti ulatuses. See näitab, kui palju leiti andmetes tundmatuid sõnu. Kuna tegemist on tõlke algoritmiga, siis tundmatute sõnade tekkimine pole optimaalne, kuna see võib tekitada olukorra, kus masin hakkab pakkuma <*unk*> asemele lause konteksti välist sõna. Kahjuks on tundmatute sõnade tekkimine vältimatu, kuid nende arvu saab vähendada, võttes kõik andmed samast andmestikust ja jaotades need treening, test ja valideerimise andmete vahele. Kogu andmete puhastamist hõlmav kood on toodud uurimistöö lisas (LISA 1).

Mudelite sisendi ja väljundi täpsustamiseks muudeti teksti vastavalt vajadusele: sisendtekstile lisati tühikud iga kirjavahemärgi ette ja taha; tühikute lisamine aitas vähendada mudeli jaoks tundmatuid sõnu. Näiteks kui treeningandmetes leidus sõna “kodu” ühes lauses ja “kodu.” teises lauses, tuvastab mudel neid täiesti erinevate sõnadena, mis viis konteksti kadumiseni. Tühikute lisamine aitas vähendada selliste olukordade mõju.

Treenitud mudeli väljundit korrigeeriti peamiselt kahel viisil:

1. Esimesena kontrolliti, et kõik tühikud oleksid ühekordsed ning eemaldati liigsed tühikud, näiteks sõna algusest või lõpust.
2. Teisena kontrolliti, et kõik kirjavahemärgid oleksid ühekordsed. Mudel võis tundmatute sõnade asemel lisada jada “? ? ? ? ? ? ? ?” või asendada sõna muude järjestikuste

kirjavahemärkidega. Need kohad asendati algse sõnaga, et säilitada lause korrektsust ja loetavus.

2.4. Treenimine ja esinenud piirangud

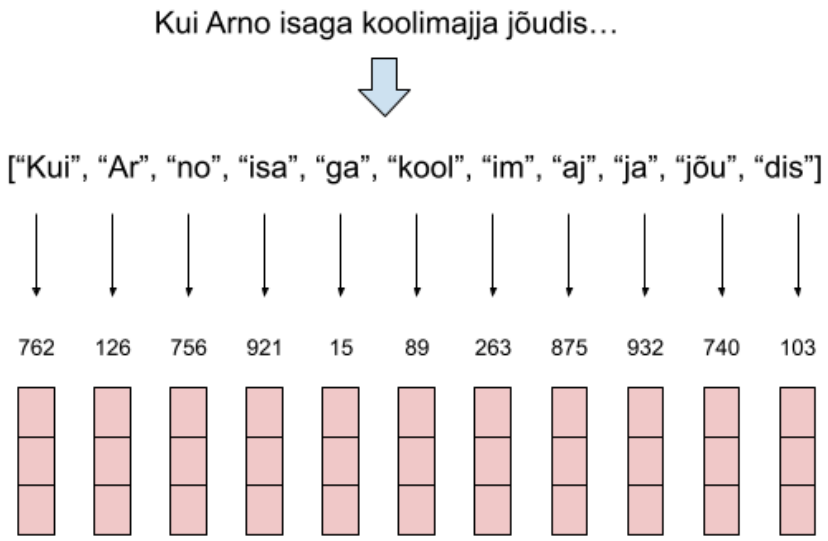
Käesolevas töös on kõik mudelid treenitud, kasutades graafikakaarti Nvidia RTX Quadro 5000 graafikakaarti, millel on 16 GB VRAM'i. Mudeli treenimine koosneb kahest etapist: andmete eeltöötlemine ning eeltöödeldud andmete põhjal treenimine.

2.4.1. Andmete eeltöötlemine

Selleks, et mudelit treenida, on vaja viia treeningandmed sobivasse formaati. Andmete eeltöötluse käigus teostatakse järgmised sammud:

1. Esmalt luuakse tokeniteks tehtud sõnade andmebaas, kus sõnad tehakse juppideks ning sarnastele sõna osadele antakse sarnased väärtused.
2. Järgnevalt koostatakse lähte- ja sihtsõnavara sõnastikud binaarsel kujul, selleks et suurendada arvutuskiirust.
3. Eeltöötlemise protsessi käigus luuakse treenimise, valideerimise ja testimise kataloogid.

Fairseqi teek pakub efektiivset lahendust treeningprotsessi läbiviimiseks kasutades eeltöötluses loodud tokenite andmebaasi. Tõlkeprotsessis kasutatakse sõna osade kaudu lähtekeelest sihtkeelde üleviimist tagades sihtkeele struktuuri. Allolev joonis on klassikaline näide sõnade tokeniseerimisest (joonis 1)

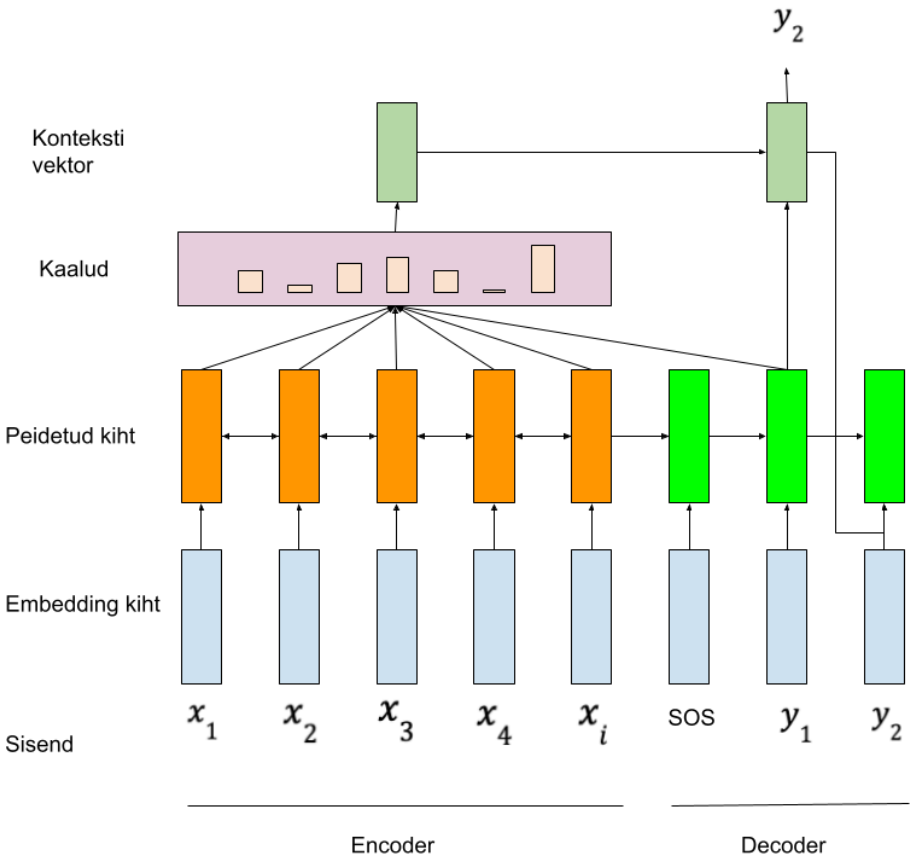


Joonis 1. Tokeniseerimise protsess

Allikas: Wisdom ML, 2022. Autori poolt täiendatud.

2.4.2. Mudeli treenimine

Treenimise käigus õpib mudel, kuidas sisendjadad muuta väljundjadaks ehk sisendteksti tokenid muudetakse väljundteksti tokeniteks ja nendest koostatakse tekst. Kuna tõlkimisel kasutatakse sõnaosasid, mitte sõnu, on võimalik tagada lähte- ja sihtkeele vastavus ja struktuuriline keerukus. Treenimiseks kasutatakse neurovõrke, mille käigus optimeeritakse mudeli kaalud, et minimeerida väljundteksti vea määra. Treenimine koosneb tsüklitest ehk epohhidest, mille käigus arvutatakse mudelile uued kaalud ja uus mudeli veamäär ning selle põhjal leitakse, kas mudeli täpsust suurendaks uue epohhi teostamine ehk edasi treenimine. (Joonis 2)



Joonis 2. Treenimise protsessi kirjeldus

Allikas: Abacus.AI, 2020. Autori poolt täiendatud.

2.4.3. Treenimisel esinenud probleemid

Treenimisel esines mitmeid probleeme, kuid kõige rohkem esines CUDA OOM (*ingl. Compute Unified Device Architecture Out Of Memory*) vigu. Tegemist on graafikakaardi mälu ülekasutamisega, mis oli tingitud andmete suurest mahust.

Mälu probleeme võib mõjutada väga mitmeid tegureid:

- Mudelit on võimalik treenida erineva täpsusega, mis määrab kui palju mälu jagatakse iga numbri väljendamise jaoks. Tavapäraselt arvutatakse mudeleid FP32 (*ingl. Floating Point 32*) täpsusega, mis näitab, et iga numbri väljendamiseks kasutatakse 32 bitti. Töös on kasutatud FP16 ehk pooliku täpsusega numbrite väljendamist (Exxact 2024). FP16 täpsus jaotub 3 osaks:
    - 1 bitt positiivse/negatiivse märgi jaoks
    - 5 bitti eksponendi esitamiseks alusega 2.
    - 10 bitti murdosa jaoks, st väärtus pärast koma.
- Veel on ka võimalik määrata maksimaalsete tokenite arv, mida masin kasutab ühe alamosade (*ingl. Batch*) jaoks. Eraldi on ka võimalik määrata alamosade arv, mida masin arvutamisel kasutab.
- Veel saab määrata ka *encoder* ja *decoder* kontekstivektorite suurused ehk mitme dimensionaalsed ruumivektorite suurused, mis omakorda määrab, kui palju mälu lubatakse kasutada *encoder* ja *decoder* etapis.

Mälu probleemide lahendamiseks kasutati erinevaid treenimise konfiguratsioone ning ka *data sharding* meetodit, mis jagab treenimise andmed väiksematesse osadesse, selleks et treenimine suurema andmemahuga mahuks kogunisti GPU mällu ära. Mudelite täpsed treenimise parameetrid on toodud lisades (LISA 2).

3. Tulemused

3.1. Eksperimentide ülevaade

Mudelite hindamiseks kasutatakse Tallinna Ülikooli poolt koostatud grammatilise veaparanduse korpust “Estonian L2 Grammatical Error Correction Corpus” (EstGEC-L2) (Tallinna Ülikool 2024), kust on võetud A2, B1, B2, C1 keeletaset hõlmavaid lauseid. Igast keeletaseme korpusest on võetud 500 näitelauset. Laused lähtuvad e-kirja tekstidest, varieerudes 5 kuni 15 lauseni ühe e-kirja kohta. Sellest lähtuvalt on valitud andmehulgad varieeruvad ning vastavad keeletasemele. Andmete analüüsimiseks hinnati luseid eraldi.

Töö käigus treenitud mudeleid võrreldi TartuNLP poolt loodud mudeli ja ChatGPT 4o mudeli skooridega. Autori poolt on treenitud 3 mudelit, mis on tähistatud ühtselt: M1, M2 ja M3. Mudelite efektiivsuse hindamiseks on uuritud kolme mudelit, mis erinevad treenimise andmete mahu poolest. Tabelis 1 on toodud kõikide töö käigus treenitud mudelite tähemärkide arv ja maht Mb’des.

Tabel 1. Treenitud mudelite tähemärkide arv ja maht

Mudel	Tähemärkide arv	Maht Mb’des
M1	112,8 miljonit	107,5
M2	72,8 miljonit	69,5
M3	101,1 miljonit	96,4

Mudeli täpsuse mõõtmiseks kasutatakse GLEU skoori, mis on skaalal nullist üheni, hinnates korrektse ja masina poolt prognoositud kirjavahemärkide paigutuse erinevust. Mudeli sisendtekstiks on võetud kõik laused, millelt on eemaldatud kõik kirjavahemärgid. Tabelis 2 on välja toodud näide, kuidas on lauset muudetud enne ja pärast korrigeerimist.

Tabel 2. Näited korrektsest tekstist, mudeli sisend ja väljundtekstist.

Korrektne tekst	Ma tean veel, et ta on puhas ja vaikne loom.
Mudeli sisendtekst	Ma tean veel et ta on puhas ja vaikne loom
Mudeli väljundtekst	Ma tean veel, et ta on puhas ja vaikne loom

Tabelis 3 on välja toodud näitelaused iga keeletaseme kohta. Tähelepanuväärne asjaolu on see, et hindamise andmetes on e-kirjade keeletase määratud ühtse tervikuna, mistõttu sisaldavad C1 taseme hindamise andmed osaliselt ka lihtlauseid ja muid sellele keeletasemele mittevastava struktuuriga lauseid.

Tabel 3.A2, B1, B2, C1 Keeletasemele vastavad näitelaused.

Keeletase	Näitelause
A2	Ma sõidan sellel laupäeval autoga.
B1	Ma olen ammu lemmikloomast unistanud, aga minu vanemad ei ole seda lubanud.
B2	Me võiksime minna koos, kuna minu abikaasa sõidab kontserdi ajaks välismaale.
C1	Ei tohi eirata ka seda, et isegi väike kogus alkoholi võib juhtimise ajal teie tähelepanu tunduvalt vähendada ning et alkoholi mõju all kasvab inimesel ohutustunne, mis võib teda tappa.

3.2. Keeletase A2 tulemused

Tabelis 4 on esitatud kõikide mudelite A2 keeletaseme GLEU skoorid. Kõrgeima tulemuse saavutas M3 mudel skooriga 0,624. M2 mudel jääb teistest mudelitest vähesel määral alla, saavutades skoori 0,428. Selline tulemus on ilmselt tingitud sellest, et mudeli treeningandmetes

on vähe isikunimesid ning mudel kipub neid valeks lugema, asendades tihtipeale nime mitme jutumärgiga või algse nime mõne muu nimega. Tähelepanuväärne on asjaolu, et ChatGPT 4o mudel jääb märkimisväärselt tulemusega alla M3 mudelile, skooriga 0,531. ChatGPT 4o mudeli vead seisnevad lauselõpumärkides, kus mudel ennustas lause lõppu punkti, kuid lause vajas kas hüüumärki või küsimärki.

Tabel 4. Keeletaseme A2 GLEU skoorid

Mudel	GLEU skoor
M1	0,455
M2	0,428
M3	0,624
ChatGPT 4o	0,531
TartuNLP	0,468

Tabelis 5 on esitatud A2 keeletasemest üks näitelause ja sellele lausele vastavalt iga mudeli poolt tehtud parandused. Tabelist on näha, et mitte ükski mudel ei korrigeerinud teksti ülima täpsusega.

Tabel 5. Mudelite korrigeeritud keeletaseme A2 näitelaused.

Mudel	Näitelause
Originaal	Ma loodan, et kõik on hästi!
M1	Ma loodan, et kõik on hästi.
M2	Ma loodan, et kõik on hästi.
M3	Ma loodan, et kõik on
ChatGPT 4o	Ma loodan, et kõik on hästi.
TartuNLP	Ma loodan,et kõik on hästi.

3.3. Keeletase B1 tulemused

Tabelis 6 on esitatud kõikide mudelite B1 taseme lausete GLEU skoorid. Kõrgeima tulemuse saavutas TartuNLP mudel, skooriga 0,592. Vähima tulemuse saavutas M2 mudel skooriga 0,404, mis on peaaegu 0,2 punkti võrra madalam, kui TartuNLP mudelil. Märkimisväärne on see, et M3 ja ChatGPT 4o mudeli erinevus on väike. M1, M2 ja M3 mudelite suurimaks vea allikaks on treeningandmetes esinevate sõnade puudumine – mudelid ei tunne ära kindlat sõna ning muudavad selle mõneks muuks sõnaks, mida mudel tunneb. Vigu analüüsides täheldati, et M2 mudel ennustas lausesse keskmiselt rohkem komasid ja jutumärke. Täpsemalt olid need pandud üksteise järgi ritta, asendades paljud lauseosad jutumärkidega. M3 mudeli iseärasuseks tuli esile olukord, kus puudusid täielikult kõik lauselõpumärgid.

Tabel 6. B1 keeletaseme GLEU skoorid.

Mudel	GLEU Skoor
M1	0,448
M2	0,404
M3	0,554
ChatGPT 4o	0,564
TartuNLP	0,592

Tabelis 7 on toodud B1 keeletasemest üks näitelause ja sellele lausele vastavalt iga mudeli poolt tehtud parandused. Lausetest on näha, et M1, M2 ja M3 ei suuda enam väljastada sisendiga sarnast lause struktuuri ning vähese andmemahu tõttu on lause kontekst kaduma läinud.

Tabel 7. Mudelite korrigeeritud B1 keeletaseme laused.

Mudel	Näitelause
Originaal	Ta ei osanud eesti keelt, aga ta oli väga sõbralik ja mõistev tüdruk.
M1	Ta ei osanud eesti keelt, aga ta oli väga sõbralik ja võib tüdruk.
M2	Ta ei osanud eesti keelt aga ta oli väga kenasti ja nime usub,
M3	Ta ei osanud eesti keelt, aga ta oli väga sõbralik ja
ChatGPT 4o	Ta ei osanud eesti keelt, aga ta oli väga sõbralik ja mõistev tüdruk.
TartuNLP	Ta ei osanud eesti keelt, aga ta oli väga sõbralik ja mõistev tüdruk.

3.4. Keeletase B2 tulemused

Tabelis 8 on esitatud kõikide mudelite GLEU skoorid B2 keeletasemega. Kõrgeima skoori saavutas ChatGPT 4o, skooriga 0,606. Madalaima skoori saavutas TartuNLP mudel, skooriga

0,371, mis on tingitud sellest, et Tartu mudel hakkab muutma sõnade järjestust ning kasutab ka sünonüüme. Kuna lause sõnastuse muutmine põhjustab kirjavahemärkide sattumist teisele kohale, võrreldes korrektse versiooniga, siis selle võrra väheneb ka vastava mudeli skoor. M1 mudel ennustab paljude vähetuntud sõnade asemele muid sõnu. Samuti nagu ka B1 tasemel, kipub M2 mudel panema üleliigseid kirjavahemärke. M3 mudeli treeningandmetes on palju rohkem isikunimesid, kuid skoori madala tulemuse põhjustab lauselõpumärkide täieliku puudumise.

Tabel 8. B2 keeletaseme GLEU skoorid.

Mudel	GLEU skoor
M1	0,427
M2	0,410
M3	0,470
ChatGPT 4o	0,606
TartuNLP	0,371

Tabelis 9 on toodud B2 keeletasemest üks näitelause ja sellele lausele vastavalt iga mudeli poolt tehtud parandused. Samuti nagu oli B1 keeletasemel on ka B2 keeletasemel: M1, M2 ja M3 ei suuda enam hoida lause struktuuri, eriti M2 mudel.

Tabel 9. Mudelite korrigeeritud B2 keeletaseme laused.

Mudel	Näitelause
Originaal	Mul on kahju , aga olen sunnitud otsima lahendusi väljaspool meie ettevõtet , kus ma olen veetnud kümme aastat .
M1	Mul on kahju,aga olen sunnitud otsima lahendusi väljaspool meie ettevõtet,kus ma olen veetnud kümme aastat.
M2	Mul on kahju aga olen sunnitud otsima
M3	Mul on kahju,aga olen sunnitud otsima lahendusi väljaspool meie ettevõtet,kus ma olen
ChatGPT 4o	Mul on kahju, aga olen sunnitud otsima lahendusi väljaspool meie ettevõtet, kus ma olen veetnud kümme aastat.
TartuNLP	Mul on kahju, aga olen sunnitud otsima lahendusi väljaspool meie ettevõtet, kus ma veetsin kümme aastat.

3.5. Keeletase C1 tulemused

Tabel 10 sisaldab kõikide mudelite C1 taseme skoore. Kõrgeima tulemuse saavutas ChatGPT 4o, skooriga 0,625. Madalaima tulemuse saavutas M2 mudel, skooriga 0,358. Märkimisväärse tulemuse saavutas M3 mudel skooriga 0,485, mis on ligikaudu 0,1 võrra madalam ChatGPT 4o ja TartuNLP mudelite skooridest. Nagu ka eelmiste keeletasemetega kipub M2 mudel lisama üleliigseid kirjavahemärke tundmatute sõnade asemele.

Tabel 10. C1 keeletaseme GLEU skoorid.

Mudel	GLEU Skoor
M1	0,422
M2	0,358
M3	0,485
ChatGPT 4o	0,625
TartuNLP	0,619

Tabelis 11 on toodud B2 keeletasemest üks näitelause ja sellele lausele vastavalt iga mudeli poolt tehtud parandused.

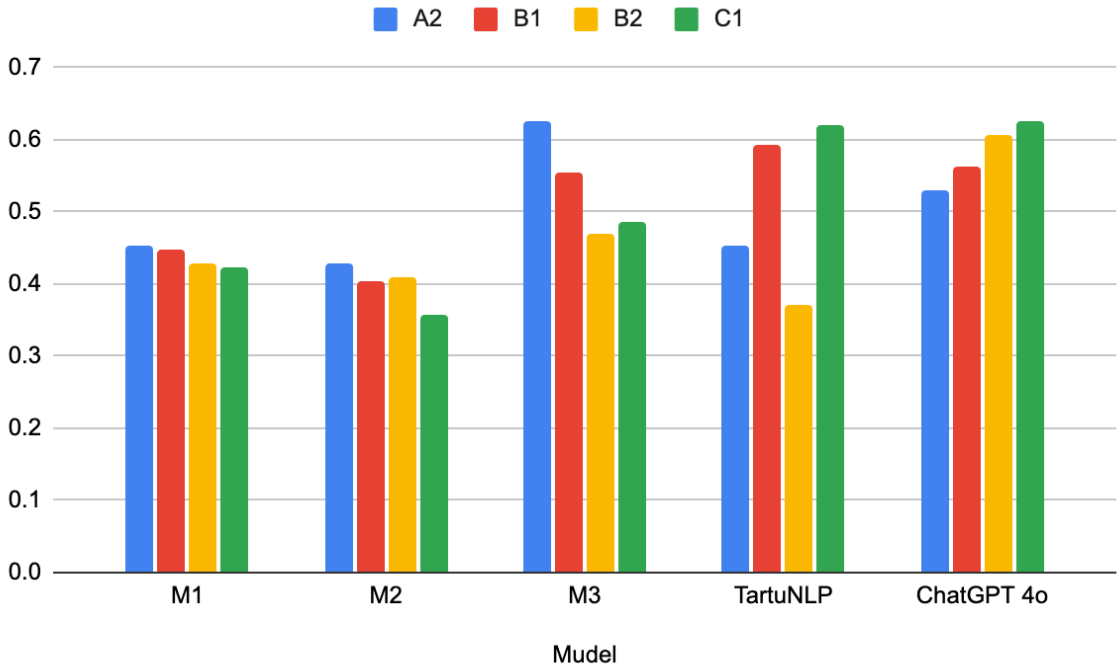
Tabel 11. Mudelite korrigeeritud C1 keeletaseme laused.

Mudel	Näitelause
Originaal	Üha rohkem juhivad päästjad tähelepanu sellele , et liigne alkoholi tarbimine kahjustab tervist ning häirib teiste inimeste rahulolekut .
M1	Üha rohkem juhivad päästjad tähelepanu sellele,et liigne alkoholi tarbimine kahjustab tervist ning häirib teiste inimeste kohalike.
M2	Üha rohkem tund töötav tähelepanu sellele,et liigne alkoholi lahendamiseks kuuluvat võimalikku ning täielik teiste inimeste saanud
M3	Üha rohkem juhivad päästjad tähelepanu sellele,et liigne alkoholi tarbimine kahjustab tervist ning häirib teiste inimeste
ChatGPT 4o	Üha rohkem juhivad päästjad tähelepanu sellele, et liigne alkoholi tarbimine kahjustab tervist ning häirib teiste inimeste rahulolekut.
TartuNLP	Üha rohkem juhivad päästjad tähelepanu sellele, et liigne alkoholi tarbimine kahjustab tervist ning häirib teiste inimeste rahulolu.

3.6. Koondtulemused

Tulemusi analüüsides, selgus, et mudelid pole veel täiuslikud ja vajavad edasiarendamist. Üldiselt on M1, M2 ja M3 mudelite tulemused head, võttes arvesse uurimistöös kasutatud treeningandmete mahtu ja arvutusvõimekust. Treenitud mudelite skoorid A2, B1 ja B2 keeletasemetele ühtivad ning ka mõnes olukorras ületavad TartuNLP ja ChatGPT 4o mudelite skoore.

Joonisel 3 on toodud kõikide keeletasemete mudelite koondtulemused. Jooniselt on näha, et mudelite M1, M2 ja M3 puhul on näha et, mida keerulisemaks muutub lause, seda väiksem on mudeli GLEU skoor.



Joonis 3. Mudelite M1, M2, M3, TartuNLP ja ChatGPT 4o GLEU skoorid vastavalt A2, B1, B2, C1 keeletasemetele.

3.7. Edasised kasutus- ja arendusvaldkonnad

Töö käigus selgus mitmeid viise, kuidas suurendada mudeli täpsust eesti keele kirjavahemärgivigade parandamisel. Järgnevalt on kirjeldatud neid viise:

1. Esmane ja ka kõige efektiivsem viis, kuidas suurendada mudeli täpsust, on suurendada arvutusvõimekust. Tähtsaim tegur, mis mõjutab mudeli arvutamist on graafikakaardi mälu. Siinses töös on kasutatud 16 Gb mälu, mis seab piirangu ühe epohhi käigus treenimisel kasutatavatele andmemahule. Graafikakaardi mälu suurendamisel saaks treenimise käigus kasutada rohkem andmeid. Näiteks kasutab ChatGPT 4o mudelite treenimisel Nvidia graafikakaarte H100, mille mälu maht on 80 Gb (Techpowerup 2024). Lisaks sellel kasutab ChatGPT 4o selliseid kaarte paralleelselt üle 10 000 tüki korraga.
2. Suurema täpsuse saavutamiseks ja rohkemate grammatikareeglite hõlmamiseks tuleks suurendada andmete mahtu ja muuta treeningandmete loogikat. Treeningandmetesse tuleks lisada nii grammatika vigu hõlmavaid lauseid, kui ka morfoloogilisi ja ortograafilisi vigu.

Kokkuvõte

Esimesed grammatikakorrektuuri hõlmavad programmid ilmusid eelmise sajandi keskel, kuid need on alles viimaste kümnendite jooksul muutunud konkurentsivõimeliseks võrreldes keskmise inimese oskustega. Grammatikakorrektuuri mudelid on eesti keeletehnoloogiale väga oluline valdkond. Eestis tegeleb eesti keele arendamisega kõige mahulisemalt TartuNLP.

Käesoleva uurimistöö eesmärk oli treenida tõlke algoritmi põhjal kirjavahemärkide korrektuuri mudel ning hinnata selle täpsus erinevate keeletasemete lausete korrigeerimisel, kasutades GLEU skoori. Selleks kasutati neurovõrgupõhist masintõlke algoritmi Fairseq teegi põhjal, et treenida masintõlke mudel, mis parandab lausetes kirjavahemärgivigu.

Esimese uurimisküsimuse vastuseks leiti, et käesolevas töös treenitud kirjavahemärgi mudelid on heal tasemel võrreldes TartuNLP ja ChatGPT 4o mudelitega. Parimaid tulemusi saavutas treenitud M3 mudel, mille täpsus madala keeletaseme puhul ületab TartuNLP ja ChatGPT 4o mudeleid. Kõrgema keeletaseme puhul on nendega samaväärne või jääb alla.

Teise uurimisküsimuse vastuseks leiti, et mudeli treenimisel on mitmeid erinevaid väljakutseid. Kõige suuremaid raskusi tekitas arvutusressurss ja graafikakaardi mälu mahtuvus. Probleeme pakkusid veel ka andmete ühtsele kujule viimine ja mudeli treenimise parameetrite parima konfiguratsiooni leidmine. Mudeli treenimisel oli võimalik määrata mitmeid parameetreid, mille muutmine mõjutas olulisel määral treenimise aega ning ka mälu kasutamist.

Kolmanda uurimisküsimuse vastuseks leiti, et uurimistöös treenitud mudelid suudavad võrdväärselt TartuNLP ja ChatGPT 4o mudelitega parandada tekstis kirjavahemärgivigu.

Antud uurimistöö valdkond on väga huvitav ning paljulubav, sest matemaatiliste arvutuste ja keele kokkupanemisel on võimalik luua inimvõimekuse tasemel teksti parandamise ja kontrollimise lahendusi. Autori arvates tuleks antud uurimistööd jätkata, kasutades suuremat andmete mahtu ja arvutusvõimekust.

Kasutatud allikad

Abacus.AI. Understanding Seq2Seq Models (2020). Loetud: https://blog.abacus.ai/blog/2020/07/10/understanding-seq2seq-models/, 10.03.2025.

Bryant, C. Yuan, Z. Briscoe, T. Qorib, M.R. Ng, H.T. Cao, T. (2022) Grammatical Error Correction: A Survey of the State of the Art[Artikkel]. Loetud: https://arxiv.org/pdf/2211.05166, 22.11.2024.

Eesti põhiseadus (RT 1992, 26, 349) preambul.

Exxact. (2020) Defining Floating Point Precision - What is FP64, FP32, FP16?. Loetud: https://www.exxactcorp.com/blog/hpc/what-is-fp64-fp32-fp16, 9.11.2024.

Fairseq. Fairseq dokumentatsioon (2025). Loetud: https://fairseq.readthedocs.io/en/latest/, 23.07.2024.

Geeks for Geeks. (2024) seq2seq Model in Machine Learning. Loetud: https://www.geeksforgeeks.org/seq2seq-model-in-machine-learning/, 10.01.2025.

Hindremäe, C.E. (2024) Kirjavahemärgivigade sünteesimine eesti keele grammatikakontrolliks [bakalaureusetöö]. Tallinn: Tallinna Ülikool

Johnson, M., Schuster, M., Le, Q., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., Hughes, M., Dean, J. (2017) Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation [Artikkel]. Loetud: https://aclanthology.org/Q17-1024.pdf, 2.02.2025.

Korotkova, E., Luhtaru, A., Liin, K., Del, M., Deksne, D., Fishel, M., (2019) Grammatical Error Correction and Style Transfer via Zero-shot Monolingual Translation [Teadusartikkel]. Loetud: https://arxiv.org/pdf/1903.11283, 10.01.2025.

Liin, K. (2008) Reeglipõhine komavigade tuvastaja eestikeelsetele tekstidele [magistritöö]. Loetud: https://lepo.it.da.ut.ee/~kaili/juhendamised/kliinmag.pdf, 10.01.2025.

Luhtaru, A., Vainikko, M., Liin, K., Fišel, M., Allkivi-Metsoja, K., Kippar, J., Eslon, P. (2024) Eestikeelse teksti automaatkorrektuur: projekti EKTB25 lõpparuanne [Teadusartikkel]. Loetud: https://arxiv.org/pdf/2402.11671, 10.01.2025.

Metashare. (2020) Estonian national corpus 2019. Loetud: https://metashare.ut.ee/repository/browse/estonian-national-corpus-2019/500ab1dea0d411eebb4773db10791bcf11aaac6c42674b0d9202bb5b6fe4324d/, 13.8.2024

Napoles, C., Sakaguchi, K., Post, M., Tetreault, J. (2015) Ground Truth for Grammatical Error Correction Metrics[Teadusartikkel] Loetud: https://aclanthology.org/P15-2097.pdf, 12.01.2025.

Sakaguchi, K. (2018) Compute sentence-level GLEU score. Loetud: https://github.com/keisks/jfleg/blob/master/eval/gleu.py, 26.10.2024.

Tartu Ülikool. (2018) Eesti keele koondkorpus. Loetud: https://www.cl.ut.ee/korpused/segakorpus/index.php?lang=et, 28.07.2024

Tallinna Ülikool. (2024) Estonian L2 Grammatical Error Correction Corpus (EstGEC-L2). Loetud: https://github.com/tlu-dt-nlp/EstGEC-L2-Corpus, 15.12.2024

TartuNLP. TartuNLP (DEPS). Loetud: https://nlp.cs.ut.ee/, 22.10.2024

Techpowerup. NVIDIA H100 PCIe 80 GB (DEPS). Loetud: https://www.techpowerup.com/gpu-specs/h100-pcie-80-gb.c3899, 22.02.2025

Tiit E.M, Statistikaamet. (2024) Eesti rahvastik. Loendamata loendatud [aruanne]. Loetud: https://rahvaloendus.ee/sites/default/files/2024-04/Eesti_rahvastik_Loendamata_loendatud_ETiit.pdf, 1.11.2024.

Wisdom ML. Tokenization in Natural Language Processing (2022). Loetud: https://wisdomml.in/tokenization-in-natural-language-processing/, 10.03.2025.

Lisad

LISA 1. Andmete puhastamise kood

```
import string
import os
from sklearn.model_selection import train_test_split

# Funktsioon, mis eemaldab tekstist kirjavahemärgid
def remove_punctuation(text):
      translator = str.maketrans('', '', string.punctuation)
      return text.translate(translator)

# Sisendandmete faili asukoht
file_path = 'treening_andmed/train.txt.correct'

# Väljundkaustade defineerimine
output_dirs = {
      'train': 'train',
      'test': 'test',
      'valid': 'valid'
}

# Kontrollime, kas väljundkaustad eksisteerivad. Kui mitte, siis loome need
for directory in output_dirs.values():
      os.makedirs(directory, exist_ok=True)

# Loeme sisendfaili sisu
with open(file_path, 'r', encoding='utf-8') as file:
      content = file.read()

# Eemaldame tekstist kirjavahemärgid
punctuation_removed_text = remove_punctuation(content)

# Funktsioon teksti jagamiseks treening-, valideerimis- ja test andmeteks
def split_text(text):
```

```
      lines = text.split('\n')
      if len(lines) < 3:
      raise ValueError("Ei ole piisavalt ridu, et jagada treening-, valideerimis-
ja test andmeteks.")

      # Jagame teksti kolmeks osaks: 50% treeningandmed, 25% valideerimise
andmed, 25% test andmed
      train_texts, remaining_texts = train_test_split(lines, test_size=0.5,
random_state=42)
      val_texts, test_texts = train_test_split(remaining_texts, test_size=0.5,
random_state=42)
      return train_texts, val_texts, test_texts

# Jagame nii originaalteksti kui ka kirjavahemärkideta teksti
train_texts, val_texts, test_texts = split_text(content)
train_punct_removed, val_punct_removed, test_punct_removed =
split_text(punctuation_removed_text)

# Funktsioon tekstide kirjutamiseks failidesse
def write_texts(train_texts, val_texts, test_texts, correct_filename,
incorrect_filename):
      # Kirjutame õiged tekstid (correct) vastavatesse kaustadesse
      with open(f'{output_dirs["train"]}/{correct_filename}', 'w',
encoding='utf-8') as file:
      file.write("\n".join(train_texts))
      with open(f'{output_dirs["valid"]}/{correct_filename}', 'w',
encoding='utf-8') as file:
      file.write("\n".join(val_texts))
      with open(f'{output_dirs["test"]}/{correct_filename}', 'w',
encoding='utf-8') as file:
      file.write("\n".join(test_texts))

      # Kirjutame valed tekstid (incorrect) vastavatesse kaustadesse
      with open(f'{output_dirs["train"]}/{incorrect_filename}', 'w',
encoding='utf-8') as file:
      file.write("\n".join(train_texts))
      with open(f'{output_dirs["valid"]}/{incorrect_filename}', 'w',
encoding='utf-8') as file:
```

```
      file.write("\n".join(val_texts))
      with open(f'{output_dirs["test"]}/{incorrect_filename}', 'w',
encoding='utf-8') as file:
      file.write("\n".join(test_texts))

# Kirjutame originaalteksti jagatud osad failidesse
write_texts(train_texts, val_texts, test_texts, 'train.txt.correct',
'train.txt.incorrect')

# Kirjutame kirjavahemärkideta teksti jagatud osad failidesse
write_texts(train_punct_removed, val_punct_removed, test_punct_removed,
'train.txt.correct', 'train.txt.incorrect')

print("Töötlemine ja jagamine on lõpetatud.")
```

LISA 2. Fairseqi keskkonna seadistamine

Fairseqi keskkonna seadistamise aluseks on võetud C.E Hindremäe bakalaureusetöö seadistamise näide.

Arenduskeskkonnana on kasutatud välist Ubuntu 24.10 serverit, millesse on ssh'ga kaugelt ligi pääsetud ning see on omakorda ühendatud VS Code'iga.

Süsteemi vajadused:

- Python > 3.10
- Fairseq
- Nvidia graafikakaart, millel on rohkem, kui 12 Gb mälu
- Graafikakaardi *dirver*id

Kättesaadavad peavad olema nii vigaste kui ka korrektsete lausetega tekstifailid ("train", "valid", "test"), vigaste lõpus peab olema keele laiend .incorrect, korrektsete lõpus .correct. Käskudesse tuleb lisada korrektsed viited failidele.

Materjali eeltöötlemiseks tuleb kasutada Fairseqi käsku fairseq-preprocess:

- fairseq-preprocess -s incorrect -t correct --trainpref train --validpref valid –testpref test

Materjali treenimiseks tuleb kasutada käsku fairseq-train:

- fairseq-train --fp16 --patience 5 --log-interval 1 --arch transformer --source-lang incorrect --target-lang correct --max-tokens 2048 --encoder-layers 3 --decoder- layers 3 --encoder-embed-dim 256 --decoder-embed-dim 256 --criterion label_smoothed_cross_entropy --optimizer adam --adam-betas '(0.9, 0.98)' --adam-eps 1e-9 --lr 0.0001 --lr-scheduler inverse_sqrt --warmup-updates 4000 --dropout 0.3

--label-smoothing 0.1 --save-dir checkpoints --save-interval 1 --tensorboard-logdir tb_logs data-bin

Interaktiivse tõlke jaoks tuleb kasutada käsku fairseq-interactive:

Üksikute lausete kaupa:

- fairseq-interactive --path checkpoint8.pt --beam 8 --source-lang incorrect --target-lang correct --max-len-a 1 --max-len-b 0 data-bin

Tekstifail korraga:

- fairseq-interactive --input input.txt --path checkpoints/checkpoint100.pt --source-lang incorrect --target-lang correct --max-len-a 1 --max-len-b 0 data-bin/ | grep -P '^H-' | cut -f3- | sed 's/@@ //g' > output.txt

Resümee

Käesolev uurimistöö keskendub eesti keele kirjavahemärkide automaatse korrektuuri mudeli treenimisele, kasutades masintõlke algoritmi. Suurte keelemudelite ja transformers arhitektuuri kiire arenguga on tekstide automaatne parandamine muutunud oluliseks vahendiks kirjaliku suhtluse parandamiseks. Eesti keele eriline struktuur ja väike kõnelejaskond muudavad kvaliteetsete grammatikakorrektuuri tööriistade arendamise oluliseks keele kasvuks ja lihtsustamiseks.

Töös kasutatakse Fairseq'i algoritmi, mis võimaldab treenida masinõppe mudelit parandama kirjavahemärkide vigu eestikeelses tekstis. Töö käsitleb teksti muutmist korrektsesse vormi, selleks et selle põhjal treenida masinõppe mudel. Töös kirjeldatakse andmete puhastamist, eeltöötlemist ja treenimist. Mudeli täpsust hinnati GLEU skoori alusel, mis on täpsuse mõõtmise skoor, mille skaala jääb 0 ja 1 vahele.

Tulemused on analüüsitud läbi nelja erineva keeletaseme, milleks on A2, B1, B2 ja C1. Uuritud on mudelite täpsuseid erinevate keeletasemetega lausete peal. Autori poolt treenitud mudelid saavutavad võrdlusmudelitega (TartuNLP ja ChatGPT 4o) sarnaseväärse või parema tulemuse. Suurim erinevus tuleb esile keerulisemate keeletasemetega lausete korral, kus mudel teeb oluliselt rohkem vigu, kui võrdlusmudelid. Peamiseks piiranguks osutusid treenimisandmete vähesus ja arvutusressurside piiratus.

Töös on toodud esile parandusvõimalused, sealhulgas arvutusvõimekuse suurendamise ja treeningandmete laiendamise.

Käesolev töö panustab eesti keeletehnoloogia arengusse, näidates masintõlke põhise kirjavahemärkide korrigeerimise teostatavust ning pakkudes suundi edasiseks arendamiseks.

Abstract

Training an automatic punctuation correction model for the Estonian language

This research focuses on training an automatic punctuation correction model for the Estonian language using a machine translation algorithm. With the rapid development of large language models and transformer architectures, automatic text correction has become an essential tool for improving written communication. The unique structure of the Estonian language and its small speaker community make the development of high-quality grammar correction tools crucial for the language's growth and simplification.

The study applies the Fairseq toolkit, which enables the training of a machine learning model to correct punctuation errors in Estonian text. The work addresses the transformation of text into a correct form to train the machine learning model accordingly. It describes data cleaning, preprocessing, and the training processes. The model's accuracy was evaluated using the GLEU score, a precision metric ranging between 0 and 1.

The results were analyzed across four different language proficiency levels: A2, B1, B2, and C1. The study investigated the models' accuracy on sentences corresponding to these language levels. The models trained by the author achieve performance comparable to or better than the reference models (TartuNLP and ChatGPT 4o). The most significant difference emerges with sentences at more complex language levels, where the model makes considerably more errors than the reference models. The primary limitations were identified as the scarcity of training data and limited computational resources.

The study highlights potential improvements, including increasing computational capacity and expanding training data.

This work contributes to the development of Estonian language technology by demonstrating the feasibility of machine translation-based punctuation correction and providing directions for further advancement.

Kinnitusleht

Uurimistöö autorina kinnitan, et

• koostasin uurimistöö iseseisvalt ning kõigile töös kasutatud teiste autorite töödele ja andmeallikatele on viidatud;

• uurimistöö vastab Tallinna Reaalkooli uurimistöö juhendile;

• olen teadlik, et uurimistööd ei edastata teistele tulu teenimise eesmärgil ega jagata teadlikult plagieerimiseks.

…….……

kuupäev/nimi/allkiri

Juhendajana kinnitan, et uurimistöö vastab Tallinna Reaalkooli uurimistöö juhendile ja lubatakse kaitsmisele.

Juhendaja

…….……..……..

kuupäev/nimi/allkiri