TARTU ÜLIKOOL
Arvutiteaduse Instituut
Informaatika õppekava

Tanel Pastarus

Tekstandmete ettevalmistamine suurte keelemudelite treenimiseks

Bakalaureusetöö (9 EAP)

Juhendaja: Mark Fišel, PhD

Tartu 2024

Tekstandmete ettevalmistamine suurte keelemudelite treenimiseks

Lühikokkuvõte:

Käesolevas bakalaureusetöö käigus taastati tõlgitud tekstandmete originaalne lausete järjekord algsete tekstikorpuste dokumentide põhjal. Mõned laused sisaldasid tõlkimise järel vigu, mida üritati töötlemisega parandada. Viidi läbi ka pilootkatse, kus peenhäälestati töödeldud andmete peal kolm GPT-2 keelemudelit, et teada saada kas tõlgitud tekstandmete kasutamine on elujõuline keelemudelite treenimiseks.

Võtmesõnad:

Keelemudelid, tekstikorpus, tekstiandmestik, peenhäälestamine, töötlemine

CERCS:

P176 Tehisintellekt

Preparing text data for training large language models

Abstract:

This bachelor’s thesis focuses on restoring the original order of translated text data by referencing the original text corpus documents. After the translation process, some sentences contained errors, which the author tried to fix by processing them. Additionally, a pilot test was conducted by fine-tuning three GPT-2 models on the processed data to assess the viability of using translated text data for training language models.

Keywords:

Language models, text corpus, tekstiandmestik, fine-tuning, processing

CERCS:

P176 Artificial intelligence

Sissejuhatus

Suured keelemudelid on tehisintellekti valdkonna üks suurimaid arenguid. Mudelid on treenitud suurel hulgal tekstiandmetel, mis võimaldab neil genereerida inimtekstile sarnast teksti saadud sisendi põhjal. Lisaks saab suuri keelemudeleid peenhäälestada kindlate ülesannete jaoks, nagu tõlkimine, küsimustele vastamine või koodilõikude genereerimine, muutes need mitmekülgseks.

Nagu eelnevalt mainitud, on keelemudelite treenimiseks vaja suurt hulka andmeid, kuid kahjuks on eesti keele tekstikorpused, võrreldes teiste laialdasemalt kasutatavate keeltega nagu inglise keel, oluliselt väiksema mahuga. Selle probleemi lahendamiseks tõlkisid Korotkova ja Fishel [1] tekstikorpused OpenSubtitles ja United Nations Parallel Corpus (edaspidi lühendatult UNPC) inglise keelest eesti keelde. Samuti tõlgiti tekstiandmestik Estonian National Corpus (edaspidi lühidalt ENC) eesti keelest 12 keelde, et tekstikorpusega oleks võimalik treenida eesti keelde tõlkimise mudeleid.

Käesoleva lõputöö eesmärk on taastada tõlgitud tekstandmete originaalne lausete järjekord algsete dokumentide põhjal ning viia läbi pilootkatse, kus peenhäälestatakse töödeldud dokumentide peal kolm GPT-2 keelemudelit. Töötlemise jaoks uuritakse tekstikorpuseid ENC, UNPC ja OpenSubtitles, et tutvuda korpuste erinevate struktuuridega ning dokumentide formaadiga. Samuti proovitakse päästa lauseid, mis on tõlkimise käigus muutunud vigaseks. Töö käigus otsitakse vastust järgmistele küsimustele:

- Kas tõlgitud lausete töötlemisega saab päästa vigaseid lauseid?
- Kas tõlgitud andmete peal treenitud mudel on parem kui tõlkimata eestikeelsete andmete peal treenitud mudel?
- Kas tõlgitud andmete kasutamine on üldse elujõuline keelemudelite treenimiseks?

Töö jaguneb neljaks suuremaks peatükiks. Esimeses peatükis antakse lühiülevaade tõlgitud korpuse loomisel kasutatud masintõlke mudelitest ja suurematest tekstiandmestikest, mis sisaldavad eestikeelseid andmeid. Teises peatükis antakse lühiülevaade korpustest ENC, OpenSubtitles ja UNPC ning kirjeldatakse tööprotsessi. Kolmandas peatükis viiakse läbi pilootkatse, kus peenhäälestatakse GPT-2 keelemudel töödeldud andmetel ning analüüsitakse mudeleid. Neljandas peatükis antakse ülevaade töö tulemustest ning tuuakse välja võimalikud edasiarendused.

1. Taust

See peatükk annab ülevaate suurematest mitmekeelsetest masintõlke mudelitest, levinumatest keelemudelitest ja suurematest tekstikorpustest, mis sisaldavad eestikeelseid andmeid.

1.1 Masintõlke mudelid

Järgnevalt tutvustatakse suuremaid mitmekeelseid masintõlke mudeleid.

1.1.1 M2M100

Meta kodulehelt [2] saadud info järgi on M2M100 avatud lähtekoodiga mitmekeelne masintõlke mudel, mis töötab lausepõhiselt ja võimaldab tõlkida mis tahes kahe keele vahel sajast. Mudeli teeb eriliseks see, et see ei toetu tõlkimisel ingliskeelsetele andmetele, vaid tõlgitakse otse valitud algkeelest valitud lõppkeelde ilma vahepeal tõlkimata inglise keelde. See aitab paremini säilitada tõlgitava lause tähendust või mõtet.

Angela Fan jt [3] Meta AI (varasemalt Facebook AI) tiimist lõid mitu-mitmele (ingl *Many-to-Many*) mitmekeelse tõlkemudeli. Tõlkemudeli treenimiseks lõid nad suuremahulise mitu-mitmele andmehulga, mis võimaldab neil tõlkida otse ühest keelest teise. Valitavate keelte jaoks oli mitu kriteeriumi: keel pidi olema laialt kasutatud, hindamise andmed pidid olema avalikud (mudeli headuse hindamiseks) ja pidid eksisteerima ükskeelsed andmed. Tulemused näitasid, et M2M100 mudel edestab teisi mudeleid, mis on treenitud peamiselt ingliskeelsetel andmetel ja ka inimkohtunikud, kes olid uurimisrühmas, väitsid, et mudeli tõlge on sorav.

Korotkova ja Fishel [1] kasutasid tõlgitud korpuse loomisel M2M100 mudeli 1,2 miljardilise parameetriga versiooni, et tõlkida tekstandmeid, mis ei ole inglise, saksa või vene keeles.

1.1.2 NLLB

Meta AI lehekülje [4] järgi on NLLB mitmekeelne masintõlke mudel, mis on mõeldud keelte jaoks, millel ei leidu palju andmeid. Meta AI tiimi eesmärgiks oli teha mudel, mis toetaks tõlkimist 200 erineval väheste andmetega keelel, keskendudes Aafrika keeltele, sest neid leidub vähe erinevates masintõlke mudelites.

Kuna andmeid leidub mõningate keelte korral vähe, lõid NLLB tiim jt [5] uue tekstikorpuse nimega FLORES-200, kus on dokumendid tõlgitud professionaalsete inimtõlkide poolt, mis on uuem versioon eelmisest andmestikust FLORES-101. Samuti tehti ka korpus NLLB Seed,

kuhu lisati professionaalsete inimtõlkide poolt tõlgitud laused 39 keeles. Andmestik loodi sellistele keeltele, kus ei eksisteerinud kvaliteetset teksti treenimise jaoks. NLLB tiim leidis, et mudel töötab paremini kui eelnevad mudelid (k.a M2M100), vaatamata sellele, et mudel hõlmab palju rohkem keeli.

1.2 Keelemudelid

Antud alapeatükis antakse ülevaade levinumatest keelemudelitest.

1.2.1 BERT

Devlini jt [6] järgi on BERT keele esitusmudel, mis kasutab transformereid ja nende kahesuunalist treenimist. Võrreldes teiste keele esitusmudelitega, loeb BERT tervet lauset korraga, mitte vasakult paremale või paremalt vasakule nagu teevad teised mudelid. Seda omadust kutsutakse kahesuunaliseks teksti vaatluseks. Tänu sellele, saavutas BERT erinevate loomulike keele töötlemise ülesannetes väga head tulemused. Eeltreenitud BERT mudelit saab ka peenhäälestada mitmesuguste ülesannete jaoks ühe kihi lisamisega, ilma mudeli arhitektuuri muutmata.

1.2.2 XLM-R

Conneau jt [7] järgi on XLM-R avatud lähtekoodiga mitmekeelne keelemudel, mis on loodud eesmärgiga arendada keelte vahelist mõistmist. Tähendab, et mudelit saab treenida ühes keeles ja kasutada teises keeles, ilma täiendavate treeningandmeteta. Mudeli treenimisel kasutati ka rohkem keeli, millel puuduvad suured andmekogumid. Treenimise käigus kasutati üle kahe terabaidi veebisorimise andmeid, mis saadi leheküljelt Common Crawl.
Conneau jt [7] leidsid, et XLM-R saavutas paremaid tulemusi kui teised mitmekeelsed mudelid, näiteks mBERT ja XLM, klassifitseerimise, järjestuse märgistamise ja küsimustele vastamise ülesannete korral.

1.2.3 GPT

GPT (ingl *Generative Pre-trained Transformer*) on süvaõppe keelemudel, mis kasutab ise juhendatud (ingl *self-supervised*) õppimist, et eeltreenida ennast suurel hulgal tekstandmetel ning mida saab peenhäälestada kindlate ülesannete jaoks nagu teksti genereerimine,

masintõlge ja teksti klassifitseerimine [8]. GPT esimene versioon, GPT-1, loodi OpenAI poolt aastal 2018 [9].

Mudelid põhinevad transformeritel, mis on närvivõrgu arhitektuur, mis töötleb sisendiks antud loomulikku keelt. Erinevalt BERT mudelist, ei kasuta GPT keelemudel transformerite kodeerija osa, vaid koosneb dekodeerija plokkidest. Transformerite enesetähelepanu mehhanism võimaldab arvestada sõna kaalu lauses ning tänu sellele arvestada kogu lause konteksti järgmise sõna genereerimisel, mis omakorda aitab mudelil paremini keelt mõista ning genereerida [8] .

Yenduri jt [8] järgi kasutab keelemudel eeltreenimise käigus juhendamata õppe metoodikat, et treenida ennast märkimisväärsel hulgal tekstandmetel. Eeltreenimise protsess hõlmab keele modelleerimist, mis on järgmise sõna ennustamine jadas, arvestades eelmisi sõnu. Protsessi tulemusena tulemusena õpib mudel ennustama järgnevat sõna, põhinedes eelnevatel sõnadel. Keelemudelit saab peale eeltreenimist peenhäälestada kindlate ülesannete jaoks. Peenhäälestamise jaoks treenitakse mudelit ülesandega seotud andmete peal, mille käigus muutuvad ka mudeli parameetrid, et tagada parim tulemus.

Võrreldes BERTiga, mis ka kasutab transformeri arhitektuuri, ei ole GPT kahesuunaline. GPT mudelil pole vaja kahesuunalisust, kuna GPT keelemudelit kasutatakse ainult eelnevate sõnade põhjal järgmise sõna ennustamiseks. Ennustatud sõna lisatakse ennem sisendiks antud sõnade jadale ning sisestatakse uuesti mudelile, genereerides järgmise sõna. Genereerimine lõpeb, kui laused on täielikud [10].

1.3 Tekstikorpused

Järgnevalt tutvustatakse suuremaid tekstikorpuseid, mis sisaldavad eestikeelseid andmeid. Tekstikorpus on EKI teatmiku [11] järgi suur hulk elektroonilisi andmeid, mis koosnevad kirjalikest või suulistest tekstidest.

1.3.1 MADLAD-400

MADLAD-400 on suur mitmekeelne dokumendi-tasemel andmestik. MADLAD-400 andmed on saadud leheküljelt Common Crawl[1]. Common Crawl haldab avatud hoidlat, mille andmed on saadud veebisorimise (ingl *web crawl*) käigus [12]. Saadud andmete peal kasutati LangID keeletuvastuse mudelit, mis annoteeris teksti dokumendi tasemel. Saadud dokumentidest

[1] https://commoncrawl.org/

filtreeriti välja laused, mis ei olnud sobivad ja ka dokumendid, milles oli vähem kui viis lauset. Tekstikorpusest eksisteerib kaks versiooni: puhastamata versioon, mis on saadud enne LangID kasutamist, ja puhastatud versioon, millele on rakendatud mitmeid erinevaid filtreid. Lõpuks saadi andmestik, kus on kokku 419 erinevat keelt. Puhastatud versioon koosneb 4 miljardist dokumendist ja 105 miljardist lausest, millest 5,5 miljonit dokumenti ja 176,3 miljonit lauset on eestikeelsed. Puhastamata versioon koosneb 7,8 miljardist dokumendist ja 148,4 miljardist lausest, millest 8,8 miljonit dokumenti ja 223,8 miljonit lauset on eestikeelsed. MADLAD-400 eeliseks teiste andmestike ees on seal olevate keelte koguarv ja andmete kvaliteet [13].

Testimise jaoks treenisid Kudugunta jt [13] MADLAD-400 andmestikul kolm mudelit erinevate suurustega: 3 miljardi, 7 miljardi ja 10 miljardi parameetriga. Treenitud mudeleid võrdlesid nad teiste mudelitega, sealhulgas ka mudeliga NLLB. Võrdlemise käigus leidsid Kudungunta jt, et kaks väiksemat MADLAD-400 tekstikorpusel treenitud mudelit olid konkurentsivõimelised nendest mitu korda suurema NLLB mudeliga, millel on 54 miljardit parameetrit.

1.3.2 CulturaX

CulturaX on Nguyeni jt [14] poolt loodud mitmekeelne andmestik, mis on mõeldud suurte keelemudelite treenimiseks. Korpus koosneb 167 keelest ja ühendab mC4 andmestiku versiooni 3.1.0 kõigi olemasolevate OSCARi korpuse väljaannetega, hõlmates versioone 20.19, 21.09, ja 23.01. Ühendamise tulemusena saadi 13,5 miljardit dokumenti. Saadud andmed läbisid töötlemisprotsessi, kus eemaldati mürarikkad, halva sisuga ja kahjulike allikatega dokumendid, läbiti meetriline puhastamine, kus jällegi eemaldati halva kvaliteediga dokumendid ja puhastati säilitatud dokumente veelgi, et parandada kvaliteeti. Peale töötlemist jäi alles 7,2 miljardit dokumenti koos 6308 miljardi tokeniga, millest 8 miljonit dokumenti koos 8,81 miljardi tokeniga on eestikeelsed.

2. Tööprotsess

See peatükk annab ülevaate töödeltavatest tekstikorpustest ning tööprotsessist. Praktilise osa jaoks kasutati Pythoni programmeerimiskeelt [15] kõrgjõudlustöötluse klastris nimega Rocket [16].

2.1 HPC klaster Rocket

Lõputöös kasutati UTHPC klastrit nimega Rocket, mis on Linuxil põhinev kõrgjõudlustöötluse arvutiklaster, mis kasutab Slurmi järjekorrasüsteemi. Rocket koosneb 64 arvutussõlmest, sisaldades kokku umbes 13 tuhat tuuma, üle 50 terabaidi mälu ja 64 graafikakaarti [17].

2.2 Andmed

Tõlgitud andmestik saadi kolmest tekstikorpusest: OpenSubtitles, UNPC ja ENC. OpenSubtitles ja UNPC andmestikud olid inglise keeles ja tõlgitud eesti keelde, ENC oli eesti keeles ja tõlgitud kaheteistkümnesse erinevasse keelde. Tõlgitud tekstikorpused asuvad UTHPC Rocket klastris, kuid mitte sama struktuuriga nagu originaalsed andmestikud. Iga korpuse tõlgitud andmed asusid vastavas failis, kujul alglause, tabulaator, tõlkelause, tabulaator ning tõlkimise hinnang. Erandiks on ENC tõlkefail, kus iga rea alguses oli lisaks fraas „__et__”. OpenSubtitlesi tõlkefaili väljavõtet on näha joonisel 1.

```
I'm going to wait.      Ma jään ootama.      -0.8006728887557983
Here we go.      Siit me läheme.      -1.4138981103897095
```

Joonis 1. Väljavõte tekstikorpuse OpenSubtitles tõlgitud korpuse failist.

2.2.1 OpenSubtitles

OpenSubtitles on filmide ja telesarjade subtiitritest koosnev mitmekeelne andmestik, selle töö käigus kasutati ainult ingliskeelset osa. Korpus koosneb 446 tuhandest dokumendist ja 441 miljonist lausest [18]. Filmid ja telesarjad on organiseeritud väljalaskeaasta, IMDB poolt määratud filmi identifikaatori ja OpenSubtitlesi poolt määratud subtiitrite identifikaatori järgi. Subtiitrite failid on XML-formaadis, kus kogu sisu asub „<document>” sildi sees ning subtiitri tekstiread asuvad on „<s>” siltide sees koos „<time>” siltidega, mis näitavad subtiitrite ekraanile ilmumise ja kadumise ajahetke. Dokumendi lõpus asuvad ka metaandmed

„<meta>” sildi sees [19]. Korpus saadi leheküljelt OPUS[2]. Joonisel 2 on näha väljavõtet ühest dokumendist.

```
<?xml version="1.0" encoding="utf-8"?>
<document id="5993964">
  <s id="1">
    <time id="T1S" value="00:00:01,688" />
Previously on "the Roosevelts, "
    <time id="T1E" value="00:00:03,723" />
  </s>
  <s id="2">
    <time id="T2S" value="00:00:04,171" />
a sickly child roused himself into a life of action.
    <time id="T2E" value="00:00:07,845" />
  </s>
```

Joonis 2. Väljavõte OpenSubtitles tekstikorpuse dokumendist.

2.2.2 UNPC

UNPC on Ziemski jt [20] poolt loodud keelekorpus, mis koosneb ÜRO ametlikest dokumentidest välja antud aastatel 1990-2014. UNPC loodi osana ÜRO pühendumusest mitmekeelsusele. Andmestiku eesmärk on hõlbustada uurimistööd ja teha edusamme mitmesugustes loomuliku keele töötlemise ülesannetes, sealhulgas masintõlkes. Tekstikorpus on käsitsi tõlgitud ÜRO peaassamblee ja konverentside juhtimise osakonna (ingl *UN Department for General Assembly and Conference Management*) poolt. Korpus on tõlgitud araabia, hiina, inglise, prantsuse, vene ja hispaania keelde. Korpus on saadaval leheküljelt OPUS[2].

Kokku on andmestikus 34,5 miljonit lauset ja 159 tuhat dokumenti [21]. Kõik dokumendid on sorteeritud kaustadesse keele, väljalaskeaasta ja väljaande sümboli alusel. Igas dokumendis asuvad andmed „<body>” sildi sees, mis omakorda kuulub „<text>” sildi alla. Tekstandmed jagunevad omakorda lõikudeks, mille alla kuuluvad laused, vastavad sildid on „<p>” ja „<s>” [20]. Joonisel 3 on näha osa ühest UNPC dokumendist.

[2] https://opus.nlpl.eu/

```
<text>
   <body>
     <p id="1">
       <s id="1:1" lang="en">AMCEN/SS/IV/3</s>
     </p>
     <p id="2">
       <s id="2:1" lang="en">AMCEN AU</s>
     </p>
     <p id="3">
       <s id="3:1" lang="en">African Ministerial Conference on the Environment</s>
     </p>
     <p id="4">
       <s id="4:1" lang="en">Distr.: General</s>
     </p>
     <p id="5">
       <s id="5:1" lang="en">19 September 2011</s>
```

Joonis 3. Väljavõte UNPC andmestiku dokumendist.

2.2.3 ENC

Eesti keele ühendkorpus (ENC) on mahult suurim eesti keele korpus. ENC koosneb mitmetest erinevatest allikatest kogutud tekstidest, suurema osa andmetest on saadud veebist. Korpus on loodud Eesti Keele Instituudi ja tarkvarafirma Lexical Computing Ltd. koostöö käigus [22].

Korpusest on väljas neli versiooni: eesti keele ühendkorpus 2013, 2017, 2019 ja 2021. Selles töös kasutati kõige uuemat, ehk aastal 2021 ilmunud andmestikku. Korpus koosneb 11 alamkorpusest ehk kokku 197 miljonit lauset ja 12 miljonit dokumenti, mis teeb sellest suurima mahuga ENC versiooni [22]. Kõik alamkorpused on saadaval Entu keeleressursside[3] veebilehelt.

Alamkorpused asuvad eraldi failides ning igas failis on dokumendid eraldatud vastavate „<doc>" siltidega. Dokumenti alustavas sildis on olemas ka metaandmed dokumendi kohta. Igas dokumendis on laused eraldatud „<s>" siltidega. Joonisel 4 on näha väljavõtet ühest ENC alamkorpuse dokumendist.

[3] https://entu.keeleressursid.ee/

```
<doc id="738253" src="Web 2013" length="100k-1M" crawl_date="2013-01-10"
<p heading="1">
<s>
Õigusõpetus
</s>
</p>
<p heading="0">
<s>
Siin on peamised teemad.
</s>
</p>
```

Joonis 4. Väljavõte ENC alamkorpuse dokumendist.

2.3 Andmete töötlemine

Andmete töötlemiseks kasutati Pythoni programmeerimiskeelt ja Bashi (Unixi käsukeel) skripte. Suurem osa tööst toimus UTHPC klastris Rocket, kuhu sai siseneda läbi SSH (Secure Shell, turvaline kest). Kood läbis mitu iteratsiooni, kuna andmete töötlemise käigus leiti vigaseid tõlgitud lauseid. Iga iteratsiooni käigus tehti täiustusi, mis lahendasid leitud probleemid. Töötlemist alustati korpusest OpenSubtitles.

2.3.1 OpenSubtitles

Esimene versioon koodist tehti peamiselt katsetamise jaoks ja et näha, millised vead leiduvad tekstikorpuses. Tõlgitud lausete failist salvestati alglause ja tõlgitud lause Pythoni sõnastikku (ingl *dictionary*), kus võtmeks on alglause ja väärtuseks tõlgitud lause. Seejärel läbitakse OpenSubtitlesi kataloogi failipuu, kus avatakse iga filmi dokument ning loetakse ridahaaval selle sisu, samal ajal luuakse ning avatakse uus dokument samasse kausta. Kui rida algab sümboliga „<” ja lõpeb sümboliga „>”, siis kirjutatakse rida otse faili, eeldades, et see on XML-silt. Muul juhul otsitakse rida sõnastikust, võtme leidmise korral kirjutatakse väärtus uude faili. Kui võtit ei eksisteeri sõnastikus, lisatakse rida ikkagi uude faili, aga lisatakse juurde sõne „|MingiError|”, et kergemini probleemne rida üles leida. Esimese versiooni koodi on näha joonisel 5.

Koodi jooksutamiseks loodi ka Bashi skript, kus oli kirjas vajalikud ressursid Slurmi järjekorrasüsteemi jaoks. Algselt oli Slurmi skriptis küsitud ülesande täitmiseks 16 gigabaiti mälu, kuid sellest ei piisanud, kuna protsessil sai mälu otsa. Lahenduseks küsiti ülesande jaoks 32 gigabaiti mälu. Joonisel 6 on näha Slurmi skripti.

```
import os
folder_path = '/gpfs/space/home/tann123/Open/OpenSubtitles/raw/en'
file_path2 = '/gpfs/space/home/tann123/OpenSubtitles.en-et.tsv'

with open(file_path2, 'r', encoding="UTF-8") as file:
    li = {}
    for line in file:
        phrases = line.strip().split("\t")
        li[phrases[0]] = phrases[1]

for root, dirs, files in os.walk(folder_path):
        for file_name in files:
            if file_name.endswith(".xml"):
                file_path = os.path.join(root, file_name)
                with open(file_path, 'r', encoding="UTF-8") as file:
                    newfile = open(root+"/new"+file_name, "w", encoding="UTF-8")
                    for line in file:
                        if len(line.strip()) >= 1:
                            if line.strip()[0] == "<" and line.strip()[-1] == ">":
                                newfile.write(line)
                            elif line.strip() in li.keys():
                                newfile.write(li[line.strip()]+"\n")
                            else:
                                newfile.write(line.strip()+" |MingiError|"+"\n")
                        else:
                            newfile.write(line)
                    newfile.close()
```

Joonis 5. Esimene iteratsioon koodist.

```
#!/bin/bash

#SBATCH --partition=main
#SBATCH --time=12:00:00
#SBATCH --ntasks=1
#SBATCH --job-name="Translation moving"
#SBATCH --cpus-per-task=1
#SBATCH --mem=32g
module load python

srun python ./proov.py
```

Joonis 6. Bashi skript esimese iteratsiooni koodile.

Peale koodi tööd leiti, et tekstikorpuste tõlkimise käigus oli masintõlke mudel asendanud tema jaoks tundmatud tähed ja sümbolid fraasiga „<unk>”, vigast rida on näha joonisel 7. Esimese versiooni kood ei töödelnud algset lauset, vaid lisas lause otse sõnastikku võtmena ehk dokumenti läbides ei olnud võimalik mõningaid lauseid sõnastikust üles leida, kuna alglaused olid erinevad. Koodi teine iteratsioon eemaldas tõlgitud lausete failist kõik laused, kus oli „<unk>“ fraas tõlgitud lauses, ning lisati kustutatud lause hulka (ingl *set*), et asendamise ajal kontrollida kas lause on kustutatud. Alglausetes asendati „<unk>” fraasid ning kõik tähemärgid mis ei kuulunud ASCII tähtede ega numbrite hulka tühisõnega. Korpuste dokumente läbides tehti lausetes samasugused asendused, et sõnastikust otsides oleksid laused samal kujul. Enam ei lisatud faili rea juurde fraasi „|MingiError|” kui lauset ei leitud sõnastikust, vaid kirjutati lause asemel „__PUUDU__” koos originaalse, tõlkimata lausega. Kui leiti, et tõlgitud lause on kustutatud, kirjutati faili lause asemele „__KUSTUTATUD__” koos originaalse lausega. Samuti ei tehtud enam uut dokumenti samasse kausta, kus originaalne dokument oli, vaid loodi uus kataloogi failipuu. Joonisel 8 on näha teist iteratsiooni koodist.

```
What<unk>s a flank?     Mis on tiib?     -1.230376124382019
```

Joonis 7. Lause, kus masintõlke mudel on asendanud tähemärgi fraasiga „<unk>”.

```
import os
import re
import string

translation_file = "/gpfs/space/home/tann123/OpenSubtitles.en-et.tsv"
original_files = "/gpfs/space/home/tann123/OpenSubs/OpenSubtitles/OpenSubtitles/raw/en"
end_path = "/gpfs/space/home/tann123/NewOpenSubs/"

with open(translation_file, 'r', encoding="UTF-8") as file1:
    kustutatud = set()
    li = {}
    for line in file1:
        phrases = line.strip().split("\t")
        inglk = phrases[0]
        eestik = phrases[1]
        if "<unk>" in inglk:
            inglk = inglk.replace("<unk>", "")
        inglk = ''.join([i if i in string.ascii_letters or i.isnumeric() else '' for i in inglk])

        if "<unk>" in eestik:
            kustutatud.add(inglk)
        else:
            li[inglk] = eestik
```

```
for root, dirs, files in os.walk(original_files):
    for file_name in files:
        file_path = os.path.join(root, file_name)
        with open(file_path, 'r', encoding="UTF-8") as file:
            path_v = end_path + root[62:]
            if not os.path.exists(path_v):
                os.makedirs(path_v)
            with open(path_v + "/" + file_name, "w", encoding="UTF-8") as newfile:
                for line in file:
                    line = line.strip()
                    if len(line) >= 1:
                        if line[0] == "<" and line[-1] == ">":
                            newfile.write(line+"\n")
                        else:
                            key = line.strip()
                            key = ''.join([i if i in string.ascii_letters or i.isnumeric() else '' for i in key])
                            if key in li.keys():
                                newfile.write(li[key]+"\n")
                            elif key in kustutatud:
                                newfile.write("__KUSTUTATUD__ " +line+"\n")
                            else:
                                newfile.write("__PUUDU__ "+ line+"\n")
```

Joonis 8. Teine iteratsioon koodist.

Pärast andmete töötlemist leiti, et 161,9 tuhat vigast lauset kustutati ja 45,8 tuhat lauset oli puudu. Kokku oli originaalses andmestikus 441 miljonit lauset, ehk kaduma läks ligikaudu 0.05% andmestikust.

2.3.2 UNPC

UNPC andmestiku puhul kasutati Pythoni XML-teegi alammoodulit nimega *ElementTree* [23], et läbida XML-formaadis dokumendid. Dokumenti töödeldes läbiti ainult lausesilte („<s>”) kasutades samasuguseid meetodeid nagu OpenSubtitles andmestiku töötlemisel ning samat Slurmi skripti. Joonisel 9 on näha UNPC töötlemiseks kasutatud koodi.

```
import xml.etree.ElementTree as ET
import os
import string

path = '/gpfs/space/home/tann123/UNPC/UNPC/raw/en'
t_path = '/gpfs/space/home/tann123/UNPC.en-et.tsv'

with open(t_path, 'r', encoding="UTF-8") as file:
    kustutatud = set()
    li = {}
    for line in file:
        phrases = line.strip().split("\t")
        inglk = phrases[0]
        eestik = phrases[1]
```

```
        if "<unk>" in inglk:
            inglk = inglk.replace("<unk>", "")
        inglk = ''.join([i if i in string.ascii_letters or i.isnumeric() else '' for i in inglk])

        if "<unk>" in eestik:
            kustutatud.add(inglk)
        else:
            li[inglk] = eestik

for root, dirs, files in os.walk(path):
    for file_name in files:
        if file_name.endswith(".xml"):
            file_path = os.path.join(str(root), file_name)
            root2 = str(root).replace("UNPC","UusUNPC")
            newpath = root2+"/new_"+file_name

            if not os.path.exists(root2):
                os.makedirs(root2)
            tree = ET.parse(file_path)

            xml_root = tree.getroot()
            text = xml_root.find("text")
            body = text.find("body")

            for p in body:
                for s in p:
                    key = s.text
                    if key == None:
                        continue
                    else:
                        key = ''.join([i if i in string.ascii_letters or i.isnumeric() else '' for i in key])
                        if key in li.keys():
                            newtext = li[key]
                            s.text = newtext
                        elif key in kustutatud:
                            newtext = "__KUSTUTATUD__ " + s.text
                            s.text = newtext
                        else:
                            newtext = "__PUUDU__ " + s.text
                            s.text = newtext
            tree.write(newpath, xml_declaration=True, encoding="UTF-8", method="xml")
```

Joonis 9. UNPC tekstikorpuse töötlemiseks kasutatud kood.

Koodi jooksutamise tulemusena leiti vigu UNPC andmestikust - leidusid dokumendid, kus ei olnud ampersandi sümbolid õigesti paotud (ingl *escaped*). Kuna ampersandi sümbolid tähistavad XML-elemendi algust, ei saa neid ilma pagumata kasutada. Sümbolite olemasolu XML-failides tõi kaasa krahhid XML-puu parsimisel. Vigaste ridade arvu leidmiseks kasutati regulaaravaldist, mis kontrollib, et kas leidub failis sümbol „&” mis ei ole õigesti paotud. Regulaaravaldist kasutati koos Linuxi grep käsuga, mille tulemusena avastati neli vigast rida kahe faili peale. Kuna kokku oli vigaseid ridu vähe, oli kõige lihtsam need käsitsi parandada. Dokumentides asendati „&” sümbol fraasiga „&”, mis XML-failides vastab

ampersandile. Samuti leiti ning eemaldati kolm täiesti tühja faili andmestikust. Joonisel 10 on näha valesti paotud ampersandiga rida UNPC korpusest.

```
<title>The United Nations Parallel Corpus v1.0 - CCPR/C/107/D/1835&1837/2008</title>
```

Joonis 10. Valesti paotud ampersandiga rida UNPC korpusest.

Andmestikust eemaldati 196,2 tuhat vigast lauset ja puudu oli 632,7 tuhat lauset. Kokku oli originaalses andmestikus 34,5 miljonit lauset, kaduma läks ligikaudu 2,4% andmestikust.

2.3.3 ENC

ENC tekstikorpuse töötlemiseks võeti aluseks korpuse OpenSubtitlesi töötlemisel kasutatud kood. Kuna ENC tekstikorpuse tõlkefailides oli iga rea ees fraas „__et__”, pidi selle eemaldama. Eelnevate andmekogude korral tõlgiti failid eesti keelest inglise keelde, ENC korral tõlgiti dokumendid eesti keelest 12 keelde: araabia, hiina, inglise, soome, prantsuse, saksa, läti, leedu, vene, hispaania, rootsi ja ukraina. Kokku oli 132 tõlgitud alamkorpuse faili. Alamkorpuse failid sorteeriti tõlgitud keelte alusel kaustadesse.
Kokku eemaldati kõikide keelte ENC tõlgitud alamkorpuste peale 18,5 mln lauset ja puudu oli 161,2 mln lauset, kaduma läks ligikaudu 8,2% andmestikust.

2.4 Tulemused

Töödeldud korpused on kättesaadavad leheküljelt Metashare[4] ning andmed töötlemise kohta on alltoodud tabelis 1. Kokku on algsete andmestike peale 2,6 mld lauset, millest kaotati 180,8 mln lauset, mis on 7% lausete koguarvust. Ilma asendamiseta oli 277,7 mln lauset puudu. Asendamisega oli puuduolevate lausete arv 161,9 mln, tänu töötlemisele saadi juurde 115,8 mln lauset ehk 41,7% puuduolevatest lausetest päästeti.

[4] https://metashare.ut.ee/repository/search/?q=SynEst

Tabel 1. Töötlemise tulemusena saadud korpuste andmed.

Korpus	Lauseid algkorpuses	Lauseid kustutatud	Enne asendamist puudu lauseid	Peale asendamist puudu lauseid	Puuduole-vatest andmetest päästetud	Kaotatud lausete osakaal korpusest	Lauseid kokku tõlgitud korpuses
OpenSubtitles	441,4 mln	161,9 tuh	1,6 mln	45,8 tuh	97,1%	0,05%	441,2 mln
UNPC	34,5 mln	196,2 tuh	1,3 mln	632,7 tuh	51,3%	2,4%	33,7 mln
ENC	2,2 mld	18,5 mln	175,8 mln	161,2 mln	8,3%	8,2%	2 mld
Kokku	**2,6 mld**	**18,9 mln**	**277,7 mln**	**161,9 mln**	**41,7%**	**7%**	**2,4 mld**

Esimeses veerus on korpuse nimi, välja arvatud viimane rida, kus on kõikide andmestike andmed kokku. ENC puhul on 12 keele andmed kokku pandud. Teises veerus on lausete koguarv algses tekstikorpuses, mida tõlgiti. Kolmandas veerus on kustutatud lausete arv. Neljandas ja viiendas veerus on vastavalt enne ja peale asendamise protsessi puuduolevate lausete arv. Kuuendas veerus on välja toodud osakaal, mis näitab kui palju puuduolevatest lausetest päästeti tänu lausete töötlemisele. Seitsmendas veerus on kaotatud andmete osakaal algsest, tõlkimata tekstikorpusest. Kaotatud andmete all mõeldakse puuduolevate ja kustutatud lausete koguarvu. Viimases veerus on lausete arv tõlgitud korpuses.

3. Pilootkatse

Lõputöö käigus viidi läbi pilootkatse, kus peenhäälestati kolm OpenAI GPT-2 keelemudelit kolmel erineval treeninghulgal, kus iga treeninghulk koosneb dokumentidest. Katse eesmärk on näha, et kas tõlgitud eestikeelsetel andmetel treenitud mudel on sama hea kui eestikeelsetel andmetel treenitud mudel ja kas tõlgitud andmete kasutamine on üldsegi mõttekas.

Esimene treeninghulk koosneb ühest miljonist lausest, kus 800 tuhat lauset on ENC andmestikust ja 200 tuhat lauset on tõlgitud OpenSubtitlesi andmestikust, vastavalt kasutati 29603 dokumenti ja 169 dokumenti. Teise treeninghulga suurus on 200 tuhat lauset, laused saadi ENC andmestikust ning kasutati 7608 dokumenti. Kolmas treeninghulk koosneb ühest miljonist ENC lausest ning sisaldab 36605 dokumenti. Samuti loodi ka testimishulk, mis koosnes poolest miljonist ENC lausest ning sisaldas 46886 dokumenti.

3.1 GPT-2 peenhäälestamine

GPT-2 kolme keelemudelit peenhäälestati Hugging Face poolt ette antud näidete järgi. Kasutati Hugging Face hoidlas nimega „transformers" olevat faili „run_clm.py", mille abil saab treenida nullist või peenhäälestada keelemudeleid [24]. Keelemudelite headuse hindamiseks kasutati perplexity mõõdikut.

Igat mudelit treeniti HPC Rocket klastris ühe NVIDIA Tesla V100 graafikakaardi peal. Esimese treeninghulga peal võttis mudeli treenimine aega 7 tundi ja 6 minutit, teise treeninghulga peal võttis treenimine aega 1 tund ja 24 minutit, kolmanda puhul 5 tundi ja 39 minutit.

3.2 Tulemused

Mudelite hindamiseks kasutati perplexity skoori, mis mõõdab kui kindel on keelemudel oma ennustatud sõnas. Suurem tulemus näitab, et mudel on oma ennustuses ebakindlam ehk ta on segaduses. Väiksem tulemus näitab, et mudel on vähem üllatunud, kindlam oma vastuses ja seetõttu ka parem oodatud sõnade ennustamisel [25].

Parima tulemuse saavutas kolmas mudel, mille treeninghulk koosnes ühest miljonist ENC lausest, perplexity skooriga 14,267. Sellest veidi halvema tulemuse sai teine treeninghulk, mis oli sama suur kui eelnevalt mainitud treeninghulk, aga andmed olid ENC ja tõlgitud

OpenSubtitles andmestikest, perplexity skooriga 14,979. Halvima tulemuse sai 200 tuhande ENC lause peal treenitud mudel skooriga 22,502.

Tulemused viitavad sellele, et tõlgitud andmete lisamine aitab kaasa mudeli ennustusvõime parandamisele, kuid ei anna nii häid tulemusi kui tõlkimata eesti keele peal treenitud mudel. Kuna keelemudelite treenimiseks kasutatakse kordades rohkem andmeid kui kasutati pilootkatses, ei ole mudelite tulemused kõige täpsemad, kuid annavad ikkagi hea eelduse tõlgitud andmete kasutamisest.

4. Kokkuvõte

Käesoleva bakalaureusetöö eesmärk oli taastada tõlgitud tekstandmete originaalne lausete järjekord algsete dokumentide põhjal ning treenida kolm GPT-2 keelemudelit. Töö käigus pidi leidma vastused järgnevatele küsimustele:

- Kas tõlgitud lausete töötlemisega on võimalik päästa vigaseid lauseid?
- Kas tõlgitud andmete peal treenitud mudel on parem kui tõlkimata eestikeelsete andmete peal treenitud mudel?
- Kas tõlgitud andmete kasutamine on üldse elujõuline keelemudelite treenimiseks?

Lõputöö käigus taastati tekstikorpuste OpenSubtitles, UNPC ja ENC tõlgitud dokumentide originaalne formaat. Andmete töötlemise käigus asendati lausetes mitte-ASCII tähed ja numbrid tühisõnega, lootuses vähendada puuduvate lausete arvu. Töötlemismeetodi tulemusena päästeti 115,8 mln lauset ehk 41,7% puuduolevatest lausetest, mis demonstreerib töötlemise kasulikkust.

Pilootkatse käigus peenhäälestati kolm GPT-2 keelemudelit kolmel erineval andmestikul, et näha, kas tõlgitud andmete peal mudelite treenimine on mõttekas ning kas tõlgitud andmete peal treenimine on parem kui tõlkimata andmete kasutamine. Küsimustele vastuse leidmiseks loodi kolm erinevat andmestikku, mis koosnesid vastavalt tõlgitud, tõlgitud ja tõlkimata ning tõlkimata andmetest. Parima tulemuse saavutas tõlkimata andmete peal treenitud mudel, kuid üpriski sarnase skoori saavutas tõlgitud ja tõlkimata andmete kooslus. Tulemustest saab järeldada, et andmestiku suurendamine tõlgitud andmetega parandab mudeli ennustusvõimet.

4.1 Edasiareng

Antud lõputööd on võimalik edasi arendada mitmel erineval viisil. Andmete töötlemisel kustutati kõik laused, kus masintõlke käigus on tõlgitud lausesse tekkinud fraas „<unk>” sisse. Kasulik oleks leida töötlemisviis, mille käigus oleks võimalik päästa eelmainitud laused. Selle tulemusena oleks võimalik saada tõlgitud korpusesse juurde 18,9 mln lauset mis selle töö käigus kustutati.

Pilootkatses koosnes kõige suurem andmestik ühest miljonist lausest, mis ei ole mudelite treenimise jaoks piisav. Üks edasi arendamise viis oleks läbi viia pilootkatse kasutades kõiki korpuse dokumente, mitte ainult väikest osa. Saadud mudelite tulemused oleksid täpsemad ning saaksime parema ülevaate kuidas tõlgitud andmete kasutamine mõjutaks mudelite suutlikkust.

Viidatud kirjandus

[1] Korotkova E, Fishel M. „Estonian-Centric Machine Translation: Data, Models, and Challenges" in *Proceedings of the 25th Annual Conference of The European Association for Machine Translation*. Sheffield, UK. 2024.

[2] Fan A. Introducing the First AI Model That Translates 100 Languages Without Relying on English. Meta homepage. 2020. https://about.fb.com/news/2020/10/first-multilingual-machine-translation-model/ (8.12.2023)

[3] Fan A, Bhosale S, Schwenk H, Ma Z, El-Kishky A, Goyal S, Baines M, Celebi O, Wenzek G, Chaudhary V, Goyal N, Birch T, Liptchinsky V, Edunov S, Grave E, Auli M, Joulin A. Beyond English-Centric Multilingual Machine Translation. 2020. https://arxiv.org/abs/2010.11125 (9.12.2023)

[4] Meta AI. No Language Left Behind. https://ai.meta.com/research/no-language-left-behind/ (10.12.2023)

[5] NLLB Team, Costa-Jussà MR, Cross J, Çelebi O, Elbayad M, Heafield K, Heffernan K, Kalbassi E, Lam J, Licht D, Maillard J, Sun A, Wang S, Wenzek G, Youngblood A, Akula B, Barrault L, Gonzalez GM, Hansanti P, Hoffman J, Jarrett S, Sadagopan KR, Rowe D, Spruit S, Tran C, Andrews P, Ayan N.F, Bhosale S, Edunov S, Fan A, Gao C, Goswami V, Guzmán F, Koehn P, Mourachko A, Ropers C, Saleem S, Schwenk H, Wang J. No Language Left Behind: Scaling Human-Centered Machine Translation. 2022. https://arxiv.org/abs/2207.04672 (10.12.2023)

[6] Devlin J, Chang M-W, Lee K, Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2019. https://arxiv.org/abs/1810.04805 (10.12.2023)

[7] Conneau A, Khandelwal K, Goyal N, Chaudhary V, Wenzek G, Guzmán F, Grave E, Ott M, Zettlemoyer L, Stoyanov V. Unsupervised Cross-lingual Representation Learning at Scale. 2019. https://arxiv.org/abs/1911.02116 (11.12.2023)

[8] Yenduri G, Ramalingam M, Chemmalar Selvi G, Supriya Y, Srivastava G, Maddikunta PKR, Raj GD, Jhaveri RH, Prabadevi B, Wang W, Vasilakos AV, Gadekallu TR. Generative Pre-trained Transformer: A Comprehensive Review on Enabling Technologies, Potential Applications, Emerging Challenges, and Future Directions. 2023. https://arxiv.org/abs/2305.10435 (11.05.2024)

[9] Radford A, Karthik N, Salimans T, Sutskever I. Improving Language Understanding by Generative Pre-Training. 2018. https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf (12.05.2024)

[10] Ghojogh B, Ghodsi A. Attention Mechanism, Transformers, BERT, and GPT: Tutorial and Survey. 2020. https://www.researchgate.net/publication/347623569_Attention_Mechanism_Transformers_BERT_and_GPT_Tutorial_and_Survey (12.05.2024)

[11] Eki teatmik. https://teatmik.eki.ee/teatmik/mis-on-tekstikorpus/ (15.05.2024)

[12] Common Crawl. https://commoncrawl.org/ (11.12.2023)

[13] Kudugunta S, Caswell I, Zhang B, Garcia X, Choquette-Choo CA, Lee K, Xin D, Kusupati A, Stella R, Bapna A, Firat O. MADLAD-400: A Multilingual And Document-Level Large Audited Dataset. 2023. https://arxiv.org/abs/2309.04662 (11.12.2023)

[14] Nguyen T, Nguyen CV, Lai VD, Man H, Ngo NT, Dernoncourt F, Rossi RA, Nguyen TH. CulturaX: A Cleaned, Enormous, and Multilingual Dataset for Large Language Models in 167 Languages. 2023. https://arxiv.org/abs/2309.09400 (08.05.2024)

[15] Rossum GV, Drake F. Python 3 Reference Manual. Scotts Valley, CA: CreateSpace. 2009.

[16] University of Tartu. UT Rocket. https://share.neic.no/#/marketplace-public-offering/c8107e145e0d41f7a016b72825072287/ (06.04.2024)

[17] HPC Center. https://hpc.ut.ee/services/HPC-services/Rocket (06.04.2024)

[18] OPUS - Corpora. OpenSubtitles. https://opus.nlpl.eu/OpenSubtitles.php (viimati töötas 22.01.2024)

[19] Lison P, Tiedemann J. „OpenSubtitles2016: Extracting Large Parallel Corpora from Movie and TV Subtitles" in *Proceedings of the 10th International Conference on Language Resources and Evaluation*. Portorož, Slovenia. 2016. http://www.lrec-conf.org/proceedings/lrec2016/pdf/947_Paper.pdf (08.03.2024)

[20] Ziemski M, Junczys-Dowmunt M, Pouliquen B. „The United Nations Parallel Corpus v1.0" in *Proceedings of the 10th International Conference on Language Resources and Evaluation*. Portorož, Slovenia. 2016. https://conferences.unite.un.org/UNCorpus (09.03.2024)

[21] OPUS - Corpora. UNPC. https://opus.nlpl.eu/UNPC.php (viimati töötas 22.01.2024)

[22] Koppel K, Kallas J. Eesti keele ühendkorpuste sari 2013–2021: mahukaim eestikeelsete digitekstide kogu. 2022. http://dx.doi.org/10.5128/ERYa18.12

[23] Python Software Foundation. The ElementTree XML API. https://docs.python.org/3/library/xml.etree.elementtree.html (12.05.2024)

[24] Hugging Face. Transformers hoidla. https://github.com/huggingface/transformers/blob/main/examples/pytorch/language-modeling/run_clm.py (07.05.2024)

[25] University of Nevada Las Vegas. Use of Generative AI in Research: Perplexity and Burstiness. 2024. https://guides.library.unlv.edu/c.php?g=1361336&p=10054021 (07.05.2024)

Lisad

I. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Tanel Pastarus,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose „Tekstandmete ettevalmistamine suurte keelemudelite treenimiseks", mille juhendaja on Mark Fišel, reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni.

2. Annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 4.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni.

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

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

Tanel Pastarus

14.05.2024