TARTU ÜLIKOOL
Arvutiteaduse instituut
Informaatika õppekava

Ando Haldna

Emotikonide tuvastamine tekstis

Bakalaureusetöö (9 EAP)

Juhendaja: Heiki-Jaan Kaalep

Tartu 2016

Emotikonide tuvastamine tekstis

Lühikokkuvõte:

Käesolevas töös uuritakse emotikonide tuvastamist tekstis eestikeelse uue meedia tekstikorpuse põhjal. Teema uurimiseks koostati programmeerimiskeeles *Python* programm, mis on koostöös kasutajaga võimeline teksti põhjal erinevaid emotikone tuvastama. Samuti koostab nimetatud programm oma tööprotsessi käigus hulganisti analüüsi abistavaid faile, mis võiksid kasulikud olla ka antud teema edasistele uurijatele, et mõista paremini emotikonide olemust.

Võtmesõnad:

netikeel, emotikon, uus meedia, *Python*, tekstitöötlus

CERCS: P175, Informaatika, süsteemiteooria

Recognizing emoticons in text

Abstract:

In this Bachelor's Thesis, the main purpose was to study emoticon recognition in context of the Estonian Mixed Corpse of New Media. For this purpose, a program was written in programming language Python to recognize different emoticons with a little help from the end user. The program also creates several detailed files during its runtime, which might be most useful for the further researchers of the topic in order to gain better understanding of emoticons.

Keywords:

Internet language, Emoticons, New Media, *Python*, Text Processing

CERCS: P175, Informatics, systems theory

Sisukord

Sissejuhatus

Inimesed suhtlevad teineteisega kasutades ühist keelt – olgu see kõne või kirjutamine, viibe või miimika. Info vahetamine läbi erinevate suhtlemiskanalite on igapäevaselt tavaline ning kõigile harjumuspärane. Üheks viimastel aastakümnetel lisandunud suhtluskanaliks on Internet, mis võimaldab inimestel omavahel suhelda nii asünkroonselt foorumite vahendusel kui ka reaalajas jututubades. Koos uute võimalustega info vahetamisel on üles kerkinud ka emotsioonide väljendamine teisiti, kui sirgjooneliselt välja kirjutades, et ollakse vihane.

Analoogi võiksime võtta vahetust suhtlemisest, kus lause mõttele annab oma varjundi väljendamisel kaasnenud emotsioon. Näiteks läbi naeru öeldud „Lõpeta ära!“ ei ole ju ometi sama, mis nähvates pillatud „Lõpeta ära!“. Järelikult aitab esitluse viis sageli mõista, mida tegelikult on soovitud edasi anda. Emotsioonide tuvastamine inimesega vahetult, näost-näkku suheldes, on üpriski lihtne. Inimesele on abiks käed, silmad ja suu. Aga kui on kirjutatud näiteks Delfi kommentaarides „Jää vait!“, siis kuidas seda mõista? Mis oli kasutaja emotsioon seda öeldes või mõeldes?

Õnneks on inimesed leidnud viisi väljendada oma emotsioone ka kirjalikult. Tegu on üpriski loomuliku arenguga, eriti teades, kui suurel hulgal on erineva kujuga sümboleid meie klaviatuuril ja kui loovad on inimesed. Eri sümboleid kombineerides on tänaseks paljudel õnnestunud oma emotsioonid teksti kujule panna teisiti kui „OLEN PRAEGU VÄGA ÕNNELIK“. Siinkohal tulevadki appi emotsiooni ikoonid ehk emotikonid, mille on inimesed peaasjalikult Internetis suhtlemisel kasutusele võtnud, et oma tundeid või emotsioone teistele paremini edasi anda. Näiteks on viha väljendamiseks kasutusel emotikon ***:@*** ja naeru/positiivsuse jaoks ülimalt levinud ***:-)***.(siin ja edaspidi on punasega parema ülevaate huvides tähistatud teksti sees olevad emotikonide osad või terved emotikonid)

Eelpool on toodud mõned näited enimlevinud emotikonidest, mille kohta on info olemas. Kui palju on aga neid, mille kohta täna info puudub? Kuidas saada teada, milliseid emotikone on veel olemas ning kas võime üldse väita, et tegu on lõpliku hulgaga?

Käesoleva tööga soovitakse kaasa aidata emotikonide uurimisele ja tuvastamisele tekstides. Selle eesmärgi täitmiseks koostati programm, milles läheneti emotikonidele kui anomaaliatele tekstis, mille vormid on lõplikult teadmata. Programmis koondatakse sisendist sammhaaval erinevaid analüüsi toetavaid faile, mille abil on võimalik kogu emotikonide tuvastamise protsessi täpsemaks muuta ning üldist teadmust emotikonide kohta suurendada.

Käesoleva töö esimeses peatükis antakse ülevaade emotikonidest kui nähtusest ning nende peamistest tunnustest. Teises peatükis tutvustatakse valminud programmile sisendiks olnud uue meedia netikeele korpusi ning kolmandas kirjeldatakse programmi ja selle võimalikke edasiarendusi. Neljas peatükk annab ülevaate teistest emotikonide tuvastamist puudutavatest programmidest ning nende sobitumisest antud töö konteksti.

1. Emotikonid

Järgnevas peatükis anname ülevaate emotikonide eesmärgist nende kasutajate jaoks ja täna teadaolevast esinemise vormist Interneti eneseväljenduses.

1.1 Ajalugu

Emotikone ehk emotsiooni ikoone on kasutatud internetisuhtluses emotsioonide väljendamiseks tänaseks juba üle 33 aasta. Esmakasutajaks loetakse Scott Fahlmanni, kes 19. septembril 1982 Carnegie Mellon ülikooli arvutiteaduste üldises foorumis tegi ettepaneku nalju märkida sümbolite järjendiga ***:-)***. Kuna tegu oli huumori teemaga antud foorumis, siis tegi ta ka teise pakkumise, et ökonoomsem oleks ehk märkida mitte nalju kasutades **:-(**. [1]

1.2 Avaldumise vormid

Emotikonid moodustatakse kõige esimeselt ilmingult järeldades kirjamärkidest ning nende peamiseks eesmärgiks on edasi anda mingit emotsiooni või tundevarjundit. Kõige tuntumad on naeratus **:)** ning kurbus **:(** , kuid loomingulised internetikasutajad moodustavad ka keerukamaid tekstikunsti pilte, et vestlust värvikamaks muuta [2, p. 260]:

- **~(_8^(|)** Homer Simpson [3]
- (ﾉ゜0゜)ﾉ Üllatunud [4]
- ('')-.-('') Karu [5]

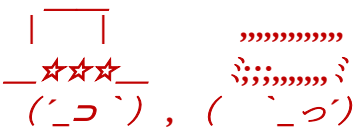
Oluliseks emotikonide hulga suurenemise põhjuseks võib lugeda Interneti levikut ning kasutajate soovi enda tundeid edasi anda rohkem, kui sõnad seda võimaldavad. Kõige selle tagajärjel on tänaseks kasutajatel emotsioonide väljendamiseks tuhandeid erinevaid sümbolite järjendeid. Kuid täna teadaolev hulk on muutmas ning täienemas, sest iga kasutaja võib luua emotikoni, mis veel kasutusel pole. Samuti on aidanud emotikonide hulga suurenemisele kaasa tekstikodeeringute areng, mis võimaldab kasutada teksti sees erinevaid sümboleid, mis pole varasemalt ASCII kodeeringus võimalik olnud. Näiteks Unicode võimaldab meil väljendada end järgnevatelt: [6]

- ☺
- ♥
- (ノ ʊ 益 ʊ)ノ彡┻━┻

1.3 Liigitamine

Emotikoni eri vormide tõttu on mõistlik emotikone liigitada ning sarnaselt Automaatsele Emotikonide tuvastamise programmile(CAO) [7] jagame käesolevas töös emotikonid kasutuse ja päritolu järgi kolmeks:

- üherealised lääne emotikonid *:-)* , *:-(* ,
- üherealised ida emotikonid *(-‿‿-)* , *(⌒ω⌒)* ,
- mitmerealised



1.4 Paiknemine tekstis

Emotikonide paiknemisel tekstis saab välja tuua tiheda seotuse kõnekeele ja lausetega. Kui kõnekeeles on fraaside ja lausete vahel sageli erinevate emotsioonide väljendamine (näiteks kõkutav naer enne või pärast lauset „See on nii naljakas“) , siis analoogne on ka emotsioonide väljendus netikeeles, kus kasutatakse vastavaid emotikone fraaside või lausete lõpus. Lisaks seotusele lausetega on üpriski tavaline, et emotikone on kasutatud eraldiseisvana, näiteks vastusena mõnele eelpool kirjutatud tekstile. [8]

1.5 Sarnasused emotikonide vahel

Emotikonide juures on samuti täheldatud, et ühe tähendusega emotikon võib sageli küll erineda vormilt, kuid seda väga vähesel määral. [9] Näiteks silmade väljendamine kooloni asemel võrdusmärgiga või naerunäo kirjutamise suund *(:* versus *:)*. Teadmine emotikonide vormide sarnasuse võimalikkusest on ka käesolevas töös seatud üheks eelduseks ning koostatud programm tegeleb sarnasuste otsimisega leitud tulemuste seas. Ennatlikult võikski ära mainida, et programmi ühe osa eesmärgiks on iga emotikoni lähendamise mõnele teisele ja protsessi kordamise tagajärjel leida, et üks emotikon on teise alus.

Käesolevas töös tuvastatakse sisendtekstist üherealiseid emotikone ning programmis on seatud eeldus, et emotikon on vähemalt pikkusega 2 tähemärki. Samuti on programmis realiseeritud hüpotees, et üks emotikon võib enda vormile väga sarnaseid emotikone.

2. Uue meedia tekstikorpus

Antud peatüki eesmärgiks on ülevaate andmine töö käigus valminud programmi sisendiks olnud tekstikorpustest.

Käesoleva töö raames koostatud programmiga analüüsitavad tekstikorpused on netikeele ehk uue meedia korpused, milles sisalduvad hetkel nelja erineva interneti tekstiklassi tekstid [10]. Uue meedia korpus koosneb järgnevatest alamkorpustest (korpuse päritolud ning mahud sulgudes):

- foorumid (Planet foorumid 2000-2008; 10,8 miljonit sõna),
- uudisgrupid (eesti uudisgrupid 2000-2004; 7,3 miljonit sõna),
- kommentaarid (Delfi kommentaarid 26.01.2004-03.2004; 2 miljonit sõna),
- jututoad.

Töödeldud tekstikorpused on ette nähtud vabaks kasutamiseks mitteärilistel eesmärkidel ning lõpptulemused on kättesaadavad Tartu Ülikooli arvutilingvistika uurimisrühma kodulehel. [11]

Tekstikorpusi on töödeldud lõplikule kujule jõudmiseks tavalise tekstifailina, kus rakendatakse analüüsis erinevaid reegleid, näiteks asendatakse regulaaravaldistega või tuvastatakse sisu keelsust. Kogu töötlusprotsessi kood on üles ehitatud UNIX skriptina.

Lõplikult märgendatud ja töödeldud XML valiidsed tekstikorpused sisaldavad paljusid erinevaid märgendeid, millest iga täpsem kirjeldus on samuti ära toodud arvutilingvistika uurimisrühma kodulehel [11]. Tekstikorpuste edasise analüüsi hõlbustamiseks on kätte saadavad kaks erinevat varianti – tsitaatide kordustega ja kordusteta. Korduste näol on tegu tagasiulatuvate viidetega mõnele eelnevale tekstile, mis on samuti uue tekstiga kaasas. Näiteks foorumites on väga sage, et kellelgi vastusena mõeldud tekst on samuti postituse osas koos autoriga märgitud. Tänu foorumite taolise protsessi toetamisele on võimalik neid eraldi märgendite vahelt leida (näiteks <tsitaat>Tsitaadi sisu</tsitaat>)

Korpuste analüüsi protsess on jaotatud mitmeks erinevaks osaks ning käesoleva töö eesmärgiks oli koostada programm, mis täiendaks ja arendaks töötluses emotikonide märgendamise osa. Emotikonide märgendamine on osa lausestamise protsessist, mille käigus proovitakse olemasolevate lausevahemärkide abil jagada morfoloogiliselt korrektseteks lauseteks ning kus probleem emotikonide näol esmalt esile kerkis.

Emotikonid moodustavad täna arvatavalt jututubade tekstides 3,2% tekstisõnadest, foorumites 0,16% tekstisõnadest, kommentaaride tekstides 0,28% ja uudisgruppides 0,37% tekstisõnadest.

[12, p. 116] Antud arvamus on koostatud käesoleva töö juhendaja ning algallika poolt märgendatud emotikonide sageduse järgi tekstikorpustes.

Käesolevas töös tegeletakse foorumite, uudisgruppide ning kommentaaride tekstides emotikonide tuvastamisega, kuna nende näol on tegu asünkroonsete suhtluskanalitega, mille märgendamine tänaseks antud täiendust enim vajaks. Töö sisend on väljavõte tekstikorpuste töötlusest, täpsemalt samast kohast, kus praeguses lausestamise protsessis emotikone on märgendatud.

3. Lõputöö käigus koostatud programm

Antud peatükis vaadeldakse lõputöö käigus koostatud programmi peamist ideed ning analüüsitakse tööprotsessi kitsaskohti ja nende põhjusi. Programm on koostatud programmeerimiskeeles *Python*, mis osutuks valituks kergesti õpitavuse, laia lisandteekide hulga ning erinevate platvormide toetamise tõttu. [13]

3.1 Peamine idee

Käesoleva töö raames valminud programm on autori teine meetod emotikonide tuvastamist tekstis uurida ning selgitatakse valitud lahendust. Seejärel tutvustatakse esialgselt proovitud lahendust ja selle käigus ilmnenud takistusi.

Töö käigus valminud programmi kandvaks ideeks on teadmatus - asjaolu, et siiani on üheselt defineerimata, mis täpselt on emotikonid ja kuidas inimesed neid kasutavad. Üks seatav määratlus neile võiks olla, et tegu on kirjamärgi kunstiga, mida kasutatakse emotsioonide väljendamiseks. Antud määratlusest lähtub ka töö autor.

Taolise kunsti alla võiksime seega lugeda järgnevad kirjamärkide järjendid:

- =>
- <8-)

Teadmatuse faktori tõttu võeti käesolevas töös suund teatava kaudu tuvastada emotikone ja koos nendega paratamatult ka kõike muud anomaalselt. Antud meetod on üpriski levinud lähenemine olukorras, kus informatsioon uuritavast on napp, aga samas on uuritav ümbritsetud teadaolevaga. Taoline kontseptsioon on tuttav matemaatikast, näiteks tarvis on leida tundmatu x, kui on antud, et 20=10+2x. Antud töö konteksti asetades saame, et eemaldades kõik tavalise ja teatava, saab tööd jätkata hulga anomaaliatega, millest omakorda tuleb leida potentsiaalsed emotikonid.

Kõige tavalise eemaldamise järel leitavate anomaaliate seast emotikonide määramise protsessi võiks käsitleda käesolevast tööst eraldiseisvalt, sest see vajab lisaks tehnilisele lahendusele ka oluliselt süstemaatilisemat ja ka taustteadmisi vajavat lähenemist. Töö käigus koostati programm, mille abil on võimalik anomaaliate seast emotikone määrata, kuid antud töö eesmärgiks pole nende määramine.

Käesoleva töö raames prooviti esmalt emotikonide tuvastamist etteantud mustrite rakendamise meetodil, analoogselt Heiki-Jaan Kaalepi tehnilise lahendusega, mida on kirjeldatud pikemalt peatükis 4. Kaalepi poolt *Shell* skriptidesse koostatu *Python*'isse kohaldamisel tekkinud

takistused viisid arusaamani, et keerukas on tuvastada midagi, mille olemus on väga abstraktselt kirjeldatud ning üheselt defineerimata. Kokkuvõtvalt jõuti arusaamale, et Kaalepi lähenemine antud probleemi lahendamiseks polnud parim ning otsustati proovida teisiti.

Peamiste probleemidena Kaalepi lahenduses võiks siinkohal välja tuua sõltuvuse täna teadaolevatest ja ehk ka levinuimatest emotikonidest. Samuti on nii lähenedes oht keerukate regulaaravaldisega märgendamise käigus rikkuda ära kogu teksti kuju või eemaldada midagi tavalist.

Teadmatuse ja ka ohtude tõttu otsustati antud töös kasutusele võtta peatüki alguses kirjeldatud lähenemine, mis on oluliselt keerukam nii programmi lähtekoodi kui ka tuvastamiseks vajaliku teadmuse osas. Lahenduse peamist ideed ja töö käigus koostatud programmist antakse detailne ülevaade peatükkides 3.2 - 3.4.

3.2 Sisendi puhastamine

Bakalaureuse töö käigus koostatud programmi oluliseks osaks ja ka erinevuseks Kaalepi poolt loodu suhtes on töötluse protsessi jälgimine igal selle sammul. Sisendi puhastamiseks loeme koodifailis *_anomaaliateLeidja.py* toimuvat selle *main* funktsiooni käivitamisel.

Sisendi puhastamise käigus koostatakse järgnevad failid (kõik tulemusfailid on nummerdatud, et oleks võimalik paremini jälgida nende loogilist järgnevust):

- *1.XML-eemaldatud.txt*
- *2.tavaline-asendatud.txt*
- *3.anomaaliad-oma-real.txt*
- *4.anomaaliad-suuremast-vaiksemaks-tyhikuteta.tsv*
- *4.1.loendatud-anomaaliad-suuremast-vaiksemaks-tyhikutega.tsv*

Kõik loetletud failid on koondatud tulemus etteantud sisendist, mille kohta anti detailsem ülevaade käesoleva töö teises peatükis. Koondatud tähendab, et tulemusfailide arv on konstantne ning sisendfailide arvust sõltumatu.

1.XML-eemaldatud.txt sisuks on kogu kaustades leidunud failidest koondatud sisend, millest on eemaldatud võimalikult palju XML märgendeid. Toimingu eesmärk on leida kasutaja poolt algselt sisestatud tekst, mis on vastavalt ümbritsetud kas XML märgenditega või on mõned tähemärgid nende kuju säilitamiseks asendatud. Nimelt on HTML-is kasutusel eritähendusega sümbolid, mis salvestamise hetkel nende originaalse tähenduse säilitamiseks asendatakse.

Tuntuim asendus võiks olla ***&*** märgi salvestamine kui ***&***. Kui taolisi asendusi ei tehtaks, oleks see oht kogu XML struktuurile ning seega on nende kasutamine üpriski levinud.

1.XML-eemaldatud.txt faili sisu loomist juhtivad regulaaravaldised on hoolikalt koostatud ning tulemuste salvestamine on vajalik, et hiljem kontrollida ega pole tehtud ennatlikke eemaldusi. Eraldi faili salvestamine annab võimaluse programmi viimases sammus leitavate tulemuste puhastamise järel olnud seisule, et saada tuge arusaamadele, mis seal tekivad või lahtisteks jäävad.

Näide reast enne töötlust, valitud Kultuuri foorumist, rida 3370:

<tuvasta_keel> a rühm, knight rider, magnum, macgyver jne jne jne...unustamatu klassika :') <lb/> bioloogia tunnis ikka ümisetakse (kogu klassiga :D) aeg-ajalt mõnda tuntud viisi sealt... </sp></tuvasta_keel>

Sama rida pärast puhastamise operatsioone failis *1.XML-eemaldatud.txt*:

a rühm, knight rider, magnum, macgyver jne jne jne...unustamatu klassika :') bioloogia tunnis ikka ümisetakse (kogu klassiga :D) aeg-ajalt mõnda tuntud viisi sealt...

1.XML-eemaldatud.txt on sisendiks järgmisele programmi sammule, mille nähtavaks tulemiks on fail *2.tavaline-asendatud.txt*. Faili sisu on *1.XML-eemaldatud.txt* lihtsustatud kuju, kus kõik tavaline on asendatud hoolega valitud sümbolitega, mida algtekstis ei esinenud. Asendamise abil on hiljem võimalik kasutatud sümbolit jälgides leida kõik, mida kasutatud regulaaravaldised ei mõjutanud.

Esmalt asendatakse jeeni(¥) sümboliga kirjamärkide järjendid, mis on pikemad kui 1 tähemärk ja võivad koosnevad järgnevatest tähemärkidest:

ABCDEFGHIJKLMNOPQRSŠZŽTUVWXYÜÕÄÖabcdefghijklmnopqrstuvwxyzšžöüõä268'"-

Üksikud numbrid on lubatud, kuna eestikeelsetes netikorpustes on 2,6,8 levinud asendused Ä,Õ ja Ö jaoks. Jutumärgid ning sidekriips on loetud sõna osaks, kui nad asuvad selle juures. Näiteks märgitakse jeeni sümboliga järgnevad kirjamärkide järjendid:

- 2ge,
- ja,
- wolf'i,
- "suur".

Woni tähisega (₩) märgitakse kas üksikut numbrit, mis on eraldatud tühikutega, või numbreid koos punkti, koma või sidekriipsuga**,** kui kogupikkus on rohkem kui 1 tähemärk. Näiteks asendatakse järgnevad numbreid sisaldavad kirjamärkide järjendid:

- 1
- 1.2
- 1,2
- 1-2

Samuti viiakse üheks jeeni tähiseks kokku kõik jeeni tähised, mille vahel on vaid tühikud ja komad, eeldusel, et kogu järjendi ees ning järel on mõni lausevahemärk(.?!). Seejärel eemaldatakse kõik sulu ja jutumärgi tähised, mis asuvad ümber jeeni tähise. Näidis läbiviidavast toimingust:

¥,¥ (¥,¥). => ¥ (¥). => ¥ ¥.

Järgnevalt eemaldatakse kõik lihtlausele sarnanevad rea osad, et kogu edasise protsessi juures lause osadeks olevate kirjamärkide osakaalu vähendada. Näiteks eemaldatakse punkt koos jeeni märgiga, kui see näib olevat lause osa:

¥. :(=> :(

Tavalise asendamise protsessi tulemuse näide, kui kasutatakse sisendiks järgnevaid ridu *1.XML-eemaldatud.txt* failist:

a rühm, knight rider, magnum, macgyver jne jne jne...unustamatu klassika :') bioloogia tunnis ikka ümisetakse (kogu klassiga :D) aeg-ajalt mõnda tuntud viisi sealt...

< ^_^ > Vähemasti minu Hollywood+ ja CTX-i kooslus küll ei nurise miskite regioonide või nende puudumise üle ...

Faili *2.tavaline-asendatud.txt* salvestatakse read:

a ¥ ¥₩¥ ¥ :') ¥ (¥ ¥ :D) ¥ ¥ …

< ^_^ >¥+¥ …

2.tavaline-asendatud.txt on heaks emotikonide uurimist toetavaks materjaliks, kuna tegu on viimase sammuga enne anomaaliate eraldamist algtekstist ning selle kompaktsus aitab vajadusel saada kiire ülevaate nii sisendteksti lausekujudest kui ka teksti ümber paiknevatest anomaaliatest.

Järgnevaks tavalise eemaldamise protsessi sisendiks on *2.tavaline-asendatud.txt*. Üksikute anomaaliate leidmiseks jagatakse read osadeks Jeeni ja Won sümbolite kohalt ning salvestatakse kirjamärkide järjendid faili *3.anomaaliad-oma-real.txt*.

Lisatingimuseks faili *3.anomaaliad-oma-real.txt* kirjutamisel on, et ilma tühikute ja kordusteta peab olema anomaalia pikkuseks vähemalt 2 tähemärki. Kuigi seetõttu võib kaduma minna emotikoni ühest sümbolist kujuline väljendusvorm, on tingimus vajalik hilisema märgendamise juures, kus tegelikult ühe kirjamärgi järgi otsustada ei saa.

Näiteks jagades osadeks *2.tavaline-asendatud.txt* rea:

a ¥ ¥₩¥ ¥ :') ¥ (¥ ¥ :D) ¥ ¥₩

Kirjutatakse *3.anomaaliad-oma-real.txt* ühe rea asemel 2 rida:

:')

:D)

Ühte sel viisil reale paigutatud kirjamärkide rida nimetame **anomaaliaks**, mille hulgast programmi edasise töö käigus soovitakse leida võimalikult suurel hulgal emotikone.

Programm koostab anomaaliatest sagedusloendi, kuhu loetakse kokku anomaaliate kõik esinemised, ning kirjutab saadud tulemused faili *4.1.anomaaliad-suuremast-vaiksemaks-tyhikutega.tsv.* Näited ridadest nimetatud failis:

13442 :)

103 $1_1@

101 , :

Lisaks koostatakse sagedusloend *4.anomaaliad-suuremast-vaiksemaks-tyhikuteta.tsv*, kus loendatud anomaaliates on tühjad osad eemaldatud ning ka tulemus on salvestatud faili ilma nendeta. Näited ridadest:

15393 :)

116 ,:

103 $1_1@

Nagu juba näidetest ilmneb, siis sageduste arv võib olla suurem tühikute kaotamise korral, näiteks ***:)*** ja ***:)*** on nüüd mõlemad ***:)***. Samas kasv tuleb tühikutega emotikonide arvelt, seega anomaaliate koguarv algtekstis siinkohal mõjutatud ei saa.

Käesolevas töös käsitletakse tühikuid emotikonide sees ja ümber juhuslikena, sest eelnevad uue meedia korpuse töötlemise algoritmid võivad olla neid juurde tekitanud ning üldjuhul ei muuda tühiku olemasolu emotikoni tähendust. Seega on töö järgnevates etappides kasutusel *4.anomaaliad-suuremast-vaiksemaks-tyhikuteta.tsv*, et oleks võimalik kasutada lihtsamaid algoritme ning vähendada anomaaliate erinemise võimalust vaid tühiku paiknemise osas.

Kokkuvõttes võib öelda, et programmi esimese osa eesmärgiks on kogu sisendi koondamine, puhastamine ja kõige tavalise elimineerimine. Tavalise elimineerimise järel loendatakse erinevad anomaaliad, et jätkata uurimist efektiivsemalt ja kasutada nende sagedusi järgnevates sammudes.

3.3 Anomaaliate lähendamine

Programmi teises osas on peamiseks eesmärgiks lähendada erinevaid anomaaliaid, et oleks võimalik leitud sarnasuste põhjal hiljem võtta vastu otsuseid nende kuuluvuse kohta emotikonide sekka. Lähendamine on töö autori tõlgendus emotikonide eri vormide esinemisest kasutajate seas. Antud peatükis kirjeldatu on *_anomaaliateL2hendamine.py main* funktsiooni käivitamisel toimuva ülevaade.

Lähendamise etapi nõutud sisendiks on programmi esimeses osas koostatud fail *4.anomaaliad-suuremast-vaiksemaks-tyhikuteta.tsv*. Read faili algusest:

15378 :)

9141).

7629 ..

7302 ??

5819 !!

Anomaaliate lähendamise programmi töö lõpuks on valminud järgnevad failid:

- *5.anomaaliad-alusega.tsv*
- *6.anomaaliad-kõigi-alustega.tsv*
- *6.1.anomaaliad-ilma-alusteta.txt*

Esimese sammuna võetakse tühikuteta sagedusloendi algusest alates iga anomaalia nii-öelda **aluseks**, mida võrreldakse Levenshteini algoritmi järgi kõigi teiste failis temale järgnevatega. Võrdlemise toimingut korratakse kuni jõutakse sagedusloendi lõppu.

Levenshteini algoritm maksimaalse kauguse jaoks tagastab kahe teksti võrdlusel arvu, mis väljendab mitu asendust, kustutamist või lisamist tuleb teha, et saada ühest teine. Levenshteini algoritmi kasutatakse käesolevas töös koostatud programmis maksimaalse kaugusega 1, millega defineerime, et erinevus võib olla vaid ühes elemendis. [14] Antud algoritmi kasutamine lõputöö kontekstis tundus sobilik, sest selle abil on võimalik leida üles tehtud üksikud asendused või muudatused emotikonide kujutamise juures.

Näiteks Levenshteini kaugused kahe anomaalia võrdlusel (siit ja edaspidi on punasega kursiivis märgitud anomaalia, millega alust kõrvutati):

_:)_ ja). korral 2

_:)_ ja .. korral 2

_.)_ ja .. korral 1

Aluseks ja võrreldavaks anomaaliaks jagamise idee põhineb emotikonide peatükis kirjeldatud emotikonide esinemisest eri vormides. Samuti on käesoleva töö autoril hüpotees, et emotikonid on teineteisega mingis osas seotud. Alusena käsitletakse siis kõige algsemat emotikoni, aga kuna kasutada on sagedusloend, siis algseks loeks kõige levinuimat vormi. Ja kuna praeguses faasis on teadmata, millised anomaaliatest on emotikonid, siis levinuima järgi koondatakse kogu anomaaliate hulk.

Levenshteini järgi leitud tulemused salvestatakse faili *5.anomaalia-alusega.tsv*, kus on koos anomaalia esinemise sagedus, anomaalia ja alus:

3728 _;)_ :)

3403 _:(_ :)

2848 _.)_ :)

2219 _?)_ :)

2186 _:/_ :)

1615 _:D_ :)

Failis *5.anomaalia-alusega.tsv* lubatakse ühel anomaalial omada mitut alust ning järgmise sammuna salvestatakse faili *6.anomaaliad-kõigi-alustega* iga unikaalne anomaalia koos kõigi tema alustega. Näited ridadest:

_=:)_ !:) =)) =)):) =: ::) :) ?:) .:) ”:)

/". /S. /a. =". /s. />. /A. /K.)". /l. ". /h. /*.
+." /" /m. Ã³". /G. !". ?". /t. /F. .". /k. /D.
5". 3". >". /"> /.

.:)). **:)). .:))) d:)). .:)) .:).**

_p! **_! _!! _p: _p. _p@ _p _p, _a!**

Anomaaliate lähendamise detailsemaks tulemiks analüüsi jätkamisel on fail *6.anomaaliad-kõigi-alustega.tsv*, kus iga rida algab anomaaliaga, millele järgneb n-arv erinevaid aluseid, millest igaüks on seatud sinna Levenshteini kauguse 1 järgi.

Siinkohal on oluline märkida, et aluseid valitud algoritmi järgi ei leidu kõikidele anomaaliatele ning antud tingimustel lähendamise näol on tegu otseselt uuritava anomaaliate hulga vähendamisega. Uue meedia korpust sisendina kasutades saame praeguse programmi põhjal faili *6.anomaalia-kõigi-alustega.tsv* kokku 12013 anomaaliat. Tühikuteta sagedusloendis, mis oli anomaaliatele aluste leidmise protsessi sisendiks, oli 14898 anomaaliat. Seega on kaduma läinud 2885 anomaaliat, mis moodustab koguhulgast ca 19%. Kaduma läinud anomaaliad on uuritavad faili *6.1.anomaaliad-ilma-alusteta.txt* abil.

Anomaaliate kaotamist saaks vältida vahetades või täiendades lähendamise algoritmi, aga antud bakalaureuse töö kontekstis on oluline see kitsaskoht ära märkida. Teema järgnevate uurijate jaoks on tegu otsustamise kohaga. Esimeseks variandiks võiks olla eraldiseisva meetodi kasutamine, et katta anomaaliate osa, mida ei suudeta lähendada ühelegi teisele. Teiseks võiks näiteks proovida täiendada algoritmi määral, mis võimaldab iga anomaalia mõne teisega samastada. Kõige lihtsam täienduse näide võiks siinkohal ollagi Levenshteini algoritmi kauguse seadmise kahele.

Kokkuvõtteks võib öelda, et peatükis käsitletud programmi osa eesmärgiks on anomaaliate kogumi koondamine, et hiljem teha otsuseid kas kogumi iseloomustavate suuruste või nende sisu järgi.

3.4 Emotikonide määramine

Programmi kolmanda ja viimase osa peamiseks eesmärgiks on anomaaliate hulgast kasutaja abiga emotikonide määramine. Käesoleva peatüki koodifailiks on *_emotikonideMaaramine.py* ning kirjeldatakse *main* funktsioonil käivitamisel toimuvat.

Eelnevalt teostatud protsesside järel on tulemuseks terve hulk anomaaliaid ning seosed nende vahel, kuid puudub jätkuvalt teadmine, millised neist on emotikonid ja millised lihtsalt

kirjamärgiread. Kombinatsioonid on väga erinevad ning leidmaks käesoleva töö raames uuritavaid emotikone, jõuame punkti, kus on tarvilik igaühe kohta neist võtta vastu otsus, kas anomaalia väljendab kasutaja emotsiooni või mitte. Kuna uue meedia korpuse sisu on inimeste omavaheline suhtlus, seega on loogiline ka programmi jaoks koguda teavet inimeselt, et oleks võimalik mõista, mida on kirjamärkide abil soovitud tegelikult väljendada.

Oluliseks teabe allikaks saab selles etapis programmi kasutaja, kes astub dialoogi, mille abil programm oma teadmust emotikonide ja anomaaliate kohta kasvatab. Iga kasutaja alustab tuvastamist nullist, ehk määratud anomaaliate hulgas pole ühtki liiget. Siinkohal võiks ära märkida, et võimalik oleks ka teadmuse algbaasi koostamine, kus asuks tänaseks juba teadaolevad emotikonid, näiteks esimene ilming ***:-)*** või ka ***:-(***.

Programmis kuvatakse käsureal dialoogi vormis 20 kõige levinumat alust ning soovitakse, et kasutaja määraks ükshaaval nende kuulumise emotikoni hulka, kirjutades kas y või Y. Kui kirjutatakse midagi muud või jäetakse väli tühjaks, siis liigitatakse anomaalia mitte-emotikoniks. Õpitu salvestatakse faili *8.teadmus-anomaaliatest.tsv*:

:) 1

.. 0

Esmase teadmuse omandamise järel õpib programm kasutaja poolt sisestatud väärtuste järgi. Õppimise eeldusena seatakse iga anomaaliale koondatud alus failis *6.anomaalia-kõigi-alustega.tsv* emotikoniks või mitte-emotikoniks, kui selle kohta leidub teave failis *8.teadmus-anomaaliatest.tsv.*

Näiteks olgu kasutaja öelnud, et ***d:)).*** näol on tegu emotikoniga (mis töö autori arvates võiks olla nokamütsiga naerunägu). Punkt selles on küll kaheldavalt emotikoni osa, aga otsust anomaaliate sisu terviklikkuse kohta siinkohal ei tehta.

Rida failis *6.anomaalia-kõigi-alustega.tsv*:

.:)). **:)). .:))) d:)). .:)) .:).**

Asenduse järel salvestatakse faili *7.anomaaliad-alustega-asendustega.tsv* rida:

.:)). **:)). .:))) 1 .:)) .:).**

Nüüd on teadmuse asendamise järel näha, et ***.:)).*** alustest üks on emotikon ja nelja kohta teave veel puudub.

Programmi praeguses konfiguratsioonis on iga määratud aluse kohta lubatud 9 määramata alust, et vastu võtta otsus rea kohta. Seega kui praegusel juhul on ühel real teave ühe aluse kohta, et tegu on emotikoniga, ning ülejäänud viie kohta info puudub, siis saame valemisse asendades

„1“*1 > „0“*0 + 0.1*5

Võrratus on tõene ning iseõppimise algoritm asetab valemis olnud rea esimese elemendi samuti väärtusega 1 faili *8.teadmus.tsv*:

.:)). 1

Analoogne võrratus on kasutusel ka mitte-emotikonide määramise tarbeks, omavahel vahetavad kohad „1“ ja „0“.

Kui real oleks olnud ka 0, siis poleks saanud võtta vastu otsust ei emotikoni ega mitte emotikoni kohta, kuna aluste seast ei tulene ühest kuuluvust. Näiteks kui faili *8.teadmus.tsv* jõuab, et:

.:))) 0

Sellisel juhul on real failis *7.anomaaliad-alustega-asendustega.tsv*:

.:)). **:)). 0 1 .:)) .:).**

Ja siin otsust ***.:)).*** kohta vastu võtta ei saa, mõlemad järgnevad võrratused on väärad:

„1“*1 > „0“*1 + 0.1*5

„0“*1 > „1“*1 + 0.1*5

Teadmatuse määr 0.1 on seatud töö autori poolt subjektiivselt ning kasutajal on koodis muudatusi tehes võimalik lihtsalt protsessi soovi korral rangemaks seadistada. Silmas peaks pidama, et antud määra suurendamise korral muutub olulisemaks ka kasutaja sisend aluste hindamisel, kuna programmil on vaja iseõppimise jaoks oluliselt rohkem algteadmust. Protsessi võiks kiirendada, kui erinevatest allikatest kogutaks kokku teadmus ning kasutataks seda siis programmi iseõppimise algoritmi jaoks.

Programmi iseseisva õppimise järel saab kasutaja otsustada, kas soovib veel anomaaliaid hinnata või lõpetab programmi töö. Kogu anomaaliate emotikonideks ja mitte-emotikonideks jagamise tulemust on võimalik kasutajal jälgida failis *8.teadmus.tsv*.

3.5 Programmi kokkuvõte

Bakalaureuse töös koostatud programm on loodud eesmärgiga leida XML formaadis algtekstist võimalikult suurel hulgal anomaaliaid, lähendada leitud anomaaliaid teineteisele ning leida

nende seast koostöös kasutajaga emotikonid. Programm on loodud teatavatele eeldustele ja teadmistele emotikonide kohta ning kindlasti on ruumi protsessi täiendamiseks, et uurimisel saadavad tulemused oleksid usaldusväärsed.

Peamiseks tulemiks võiksimegi lugeda tööriista, mille abil on võimalik vastavalt kasutaja soovile kas analüüsida kõiki anomaaliaid Uue meedia tekstikorpustes või tegeleda ainult emotikonide tuvastamise protsessiga. Programmi töö käigus koostatud failid on mõeldud abistavateks vahenditeks emotikonide ja üldiselt netikeele korpuse uurijatele.

Programm on koostatud programmeerimiskeeles *Python*, kuna antud keele näol on tegu üpriski kergesti omandatava süntaksiga keelega, mis annab võimaluse ka algteadmisi omaval isikul käesoleva töö koodi mõista. Kindlasti on võimalik ka antud tööd jätkata ning täiendada, milleks *Python* samuti väga sobilik on, kuna see on juba täna keeletehnoloogia valdkonnas üpriski edukalt kasutuses. Samuti on *Python* esimeseks keeleks, mida õpetatakse programmeerimise kursusel Tartu Ülikoolis.

3.6 Võimalikud edasised arendused

Käesoleva töö autor on programmi koostamise ja analüüsi protsessis teinud nii mõnedki tähelepanekud. Järgnevad märkused on osaliselt bakalaureuse töö käigus koostatud programmi kitsaskohtade avamine, kuid ka ideed, mis võiksid teema järgnevaid uurijaid abistada.

1. Sisendi puhastamise käigus tehakse kitsendusi, mille sisu tänase netikeele korpuse teadmuse põhjal lingvistid kontrollima peaksid. Kuna tegu on programmi kogu edasise töö otsese sisendiga, siis sealt saadavad tulemused mõjutavad oluliselt emotikonide ja mitte-emotikonide hulka liigitatut.

2. Levenshteini algoritmi järgi lähendamise juures kaob ära märkimisväärne hulk anomaaliaid, mida peaks uurima kas eraldi või täiendama algoritmi. Samuti pole välistatud, et koostatud programmi ja ka kogu emotikonide uurimise teemas osutub otstarbekamaks mõni teine algoritm, näiteks pakub üpriski paindlikke lahendusi *Python'i* teek *difflib*. [15]

3. Programmi iseseisva õppimise juures on suhteliselt kõrge vea oht, kuna iga kasutaja hindab anomaaliat subjektiivselt, mis lõpptulemusena määrab erineva lõpliku emotikonide hulga. Subjektiivsust tingib juba kasutaja isiklik kokkupuude emotikonidega. Programmi võiks edasi arendada, et abistada otsustamise protsessi kasutaja jaoks. Näiteks võiks anomaaliale kogutud aluste pealt hinnata nende omavahelisi erinevusi. Võiks ju väita, et kui anomaalial on väga palju erinevaid aluseid ilma sarnasuseta aluste vahel, siis ei pea paika ka hüpotees sarnasusest

emotikonide vahel ning anomaalia võiks selle kaudu välistada. Näide reast, kus anomaaliate vahel näib sarnasus üpriski väike.

.P.C. .O.C. .U.C. .P.S.

Näide reast, kus on sarnasus oluliselt tajutavam:

;)_)_ :)_ ;_ ;)! ;), ;)) ;). ;)? ;)(;)

Samuti võiks proovida seada üles eeskirja, mis suudaks võrrelda anomaaliat oma alustega vaadeldes elementide kordusi, näiteks ***:)*** vs ***:))*** aga ka ***:::)***. Näiteks rida failist *6.anomaalia-kõigi-alustega.tsv*:

::)))) :)))) ?:)))) ,:)))) ::))) :))))) :::)))) .:)))) ”:))))

Näitest kordusi kokku võttes saaks oluliselt eemaldada väikseid vormilisi erinevusi, mis aitaksid kaasa selgemale lõpptulemusele anomaaliate vaatlemisel.

4. Teised probleemi lahendused

Käesoleva töö koostamisel tutvuti Internetis leiduvate emotikonide uurimisega tegelevate programmidega. Samuti analüüsiti põhjalikumalt Heiki-Jaan Kaalep poolt uue meedia korpuse emotikonide märgendamiseks koostatud lahendust.

4.1 Uue meedia tekstikorpuse emotikonide märgendamise lahendus

Käesoleva töö programmi oluliseks sisuliseks abiliseks oli juba täna kasutusel olev lahendus, mis on koostatud kogu korpuste analüüsi lausestamise protsessi jaoks Kaalepi poolt.

Hetkel avalikuks kasutuseks välja antud tekstikorpused on koostatud ainult UNIX-il baseeruvates arvutisüsteemides töötavate *Shell* skriptide abil ning ka emotikonide märgendamine on kirjutatud nendesse skriptidesse integreeritud osana.

Emotikonide märgendamine toimub hulga regulaaravaldiste abil, mis on lähtuvalt tekstikorpustes esinevatest emotikonidest õpitu põhjal Kaalepi poolt koostatud. Tuvastamine ei olnud korpuste töötlemise juures planeeritud protsess, vaid vajadus tulenes avaldunud vigadest lausestamisel, mille tingisid arvukad erinevaid kirjavahemärke kasutavad emotikonid. Nimelt täheldati, et emotikonides on kasutusel paljud kirjavahemärgid nagu **( ) . , : ;** .

Näiteks üks rida praegusest töötlemiseks kasutatud failist *lausesta_foorumid.sh*, parema ülevaate andmise huvides jagatud siin eraldi ridadele:

```
| sed '/<tuvasta_keel>/s/

\([:;][-=]*[][()DppoO}S][][()DPpoO}*>]*\)

/

<emotikon> \1 <\/emotikon>

/g' \
```

Ülaltoodud muster töötleb UNIX käsuga *sed* ridu, mis algavad sümboliga <*tuvasta_keel*>(näite esimene rida),. Sulgude vahel (näite teine rida) sisaldub muster, mis asendatakse hiljem emotikonide märgendi(<*emotikon*></*emotikon*>) vahele (neljas rida, tähistatud \1).

[] vahele sisestatakse sümbolid, millest üks peab leiduma, näiteks peab algama emotikon näite mustri kohaselt kas **:** või **;** sümboliga. Näites on sellist eesmärki täitvad kantsulud värvitud punaseks. ***** annab võimaluse **[]** vahel olevatel elementidel esineda 0 kuni mitu korda, ehk praeguse mustri järgi **:** või **;** esinemise korral võib, aga ei pea järgnema **–** või **=** (näitest välja

võttes *[-=]*, [][()DPpoO}*>]**). Mustri lõpuosas näidatakse emotikoni nö suu, milleks on järjend töö käigus antud rollis leitud sümbolitest.

Antud lahenduse suurimateks kitsaskohtadeks on keerukas regulaaravaldiste täiendamine ja valetuvastuste oht. Kogu korpuse märgendamiseks kasutatakse 14 erinevat regulaaravaldist, mis siis oma tulemina suudavad leida ja paigutada märgendi <*emotikon*> vahele leitud sümbolid. Siinkohal on aga tehtud palju tagasiulatuvaid parandusi, näiteks kui märgendati emotikonina **. :/**, siis võeti sealt punkt hiljem välja. Regulaaravaldiste üpriski väike hulk aga tähendab seda, et need on üpriski keerukad ülesehituselt nin mille asjatundmatu või hoolimatu muutmine võib kaasa tuua väga suure hälbe lõpptulemuses.

4.2 *Twitter*'i emotikonide kategoriseerija – *emote-cat*

Twitter'i emotikonide kategoriseerija [16] tuvastab väga kitsast emotikonide hulka. Koodiga tutvumise järel võib öelda, et lahendus on mõeldud pigem emotikonide taga peituvate emotsioonide leidmiseks ja määramiseks. Vihjeks on emotikoni osadeks jagamine ning kategoriseerimine emotsioonide järgi juba märgendamise eel. Osadeks jagamine on käesoleva töö autori silmis siiski hea idee, kuna vähendab emotikonide baasi kogumahtu ning võimaldab tagasi rakendamise juures võtta kokku andmeid. Samuti on mõnevõrra parem saada kompaktset ülevaadet olemasolevatest emotikonidest. Näiteks määratakse seal silmade, ninade ja suude hulgad, millest siis kombineeritakse kokku erinevad emotikonid.

Miinuseks võikski lugeda tugeva spetsialiseerituse platvormile, esmapilgul üpriski keeruka programmi ülesehituse ning ka väga kitsa emotikonide tuvastamise vahemiku.

Peamiseks sihiks programmile on vaid enim levinud lääne emotikonid ning nende tuvastamine sotsiaalmeedia platvormidest – nimest tulenevalt siis peaasjalikult *Twitter*'ist

4.3 Emotsioonide tuvastamine tekstianalüüsi kaudu

Töö tarbeks info kogumise käigus leiti Chad Atkinsoni töö Emotsioonide tuvastamisest tekstianalüüsi kaudu. Antud uurimus käsitleb põhjalikumalt emotikonidele emotsiooni määramist ning seetõttu sarnaselt *Twitter*'i emotikonide kategoriseerijale mõeldud väga kitsale ning teadaolevale emotikonide vahemikule. Seetõttu ka käesolevasse töösse väga hästi ei sobitu. Käesoleva töö kontekstis väärib Atkinson'i siiski uurimus ära märkimist, kuna tegu on suunaga, millel võiks tulevikus samuti uurimust emotikonidest jätkata. [17]

Kokkuvõte

Käesoleva bakalaureuse töö raames uuriti emotikonide tuvastamist tekstis uue meedia tekstikorpuste põhjal. Selle jaoks koostati programmeerimiskeeles *Pyhton* programm, mille abil leiti tekstist esmalt anomaaliad ning seejärel koostöös kasutajaga ka emotikonid. Programmi töö käigus koostatud failid aitavad teema tulevastel uurijatel leida ideid protsessi parandamiseks või ka analüüsida anomaaliaid emotikonide paremaks määramiseks.

Programmi koostamisel ongi pööratud enim rõhku tehnilisele lahendusele, mida võiksid emotikonide tekstist tuvastamise teema järgnevad uurijad kasutada emotikoni-alaste teadmiste suurendamiseks. Kogu käesolevat bakalaureuse tööd peakski käsitlema kui sissejuhatust teemasse ning probleemi avamist, kuna täna pole emotikonide olemasolu ning olulisust teksti vormi ja ka sisu juures väga tugevalt arvestatud. Selleks, et tulevikus tekstikorpuste analüüsi täiendada ning hakata tekstile omandama väärtusi vastavalt emotikonide tähendusele, on töö autori silmis tarvis antud teemat oluliselt rohkem avada ning läheneda käesolevas töös leitud emotikonide hulgale pigem keelelisest kui tehnilisest vaatenurgast.

Programmi juhend ja nõuded

Nõuded tarkvara käivitamiseks käesolevas töös kasutatud sisendil:

1. Vaba kõvaketta ruumi ca 400MB
    a. Vajalik peamiselt sisend- ja tulemusfailide hoiustamiseks
2. Vajalik vähemalt Python 3
3. Soovituslik *Python* 3.4.3. (arendus ja testimise versioon) [18]
4. Programm vajab *Python'*i lisandteeki *editdistance*, seadistatav *pip* abil [19]
5. Käivitamiseks ja arendamiseks soovituslik UNIX baseeruv operatsioonisüsteem, koostamisel oli kasutusel *Ubuntu* 12.04 LTS.

Programmi kogu protsessi käivitab fail *main.py*, milles järgemööda käivitatakse tekstiosas samuti mainitud *Python* koodifailide *main* funktsioonid:

1. *_anomaaliateTuvastaja.py*
2. *_anomaaliateL2hendaja.py*
3. *_emotikonideM22ramine.py*

Kogu projekti koos sisendfailidega saab alla laadida:

http://math.ut.ee/~ando/emotikonidetuvastaminetekstis/programm+sisend.tar.gz

Sisendkaust on eraldi kättesaadav:

http://math.ut.ee/~ando/emotikonidetuvastaminetekstis/sisend.tar.gz

Viited

[1] Microsoft Research, „The First Smiley :-),“ [Võrgumaterjal]. Kättesaadav:http://research.microsoft.com/en-us/um/people/mbj/smiley/smiley.html. [Viimati vaadatud 10 mai 2016].

[2] A. Oja, „Eesti keel internetis,“ *Keel ja Arvuti*, Tartu, Tartu Ülikooli Kirjastus, 2006, lk. 259-267.

[3] „Celebrity Emoticons and Smileys,“ [Võrgumaterjal]. Kättesaadav:http://www.muller-godschalk.com/celebrities.html. [Viimati vaadatud 10 mai 2016].

[4] D. Green, „17 Fun Text Emoticons So You Can Stop Shrugging All the Time,“ [Võrgumaterjal]. Kättesaadav: http://mashable.com/2014/05/23/fun-text-emoticons/#ZKvDUQkvr5qr. [Viimati vaadatud 10 mai 2016].

[5] „Cool Smileys,“ [Võrgumaterjal]. Kättesaadav: http://cool-smileys.com/text-emoticons. [Viimati vaadatud 10 mai 2016].

[6] „Unicode Emoticons,“ [Võrgumaterjal]. Kättesaadav: http://unicodeemoticons.com/. [Viimati vaadatud 10 mai 2016].

[7] M. Ptaszynski, J. Maciejewski, P. Dybala, R. Rzepka ja K. Araki, „CAO: A Fully Automatic Emoticon Analysis System,“ [Võrgumaterjal]. Kättesaadav:https://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/viewFile/1828/2140. [Viimati vaadatud 9 5 2015].

[8] R. R. Provine, R. J. Spencer ja D. L. Mandell, „Emotional Expression Online,“ *Journal of Language and Social Psychology,* kd. 26, nr 3, lk. 299–307, 2007.

[9] T. Schnoebelen, „Do You Smile with Your Nose? Stylistic Variation in Twitter Emoticons,“ *University of Pennsylvania Working Papers in Linguistics,* kd. 18, nr 2, 2012.

[10] „Uus meedia - segakorpus,“ Tartu Ülikooli arvutilingvistika uurimisrühm, [Võrgumaterjal].

Kättesaadav: http://www.cl.ut.ee/korpused/segakorpus/uusmeedia/index.php?lang=et. [Viimati vaadatud 10 mai 2016].

[11] „Uue meedia korpus: foorumid, uudisgrupid, kommentaarid,“ Tartu Ülikooli arvutilingvistika uurimisrühm, [Võrgumaterjal]. Kättesaadav:http://www.cl.ut.ee/korpused/segakorpus/uusmeedia/foorumid_uudisgrupid_kommentaarid.php?lang=et. [Viimati vaadatud 10 mai 2016].

[12] K. Muischnek, H.-J. Kaalep ja R. Sirel, Korpuslingvistiline lähenemine eesti internetikeele automaatsele morfoloogilisele analüüsile, 2007.

[13] „Python,“ [Võrgumaterjal]. Kättesaadav: https://www.python.org/about/. [Viimati vaadatud 10 mai 2016].

[14] „The Levenshtein-Algorithm,“ [Võrgumaterjal]. Kättesaadav: http://www.levenshtein.net/. [Viimati vaadatud 10 mai 2016].

[15] P. S. Foundation, „6.3.difflib - Helpes for computing deltas,“ [Võrgumaterjal]. Kättesaadav: https://docs.python.org/3.5/library/difflib.html. [Viimati vaadatud 10 mai 2016].

[16] zahanm. „emote-cat“ [Võrgumaterjal]. Kättesaadav: https://github.com/zahanm/emote-cat/blob/master/scripts/emoticons.py. [Viimati vaadatud 10 mai 2016].

[17] C. Atkinson, „Deciphering Emoticons for Text Analytics,“ [Võrgumaterjal]. Kättesaadav: http://support.sas.com/resources/papers/proceedings13/104-2013.pdf. [Viimati vaadatud 10 mai 2016].

[18] "Python Downloads," [Võrgumaterjal]. Kättesaadav: https://www.python.org/downloads/. [Viimati vaadatud 10 mai 2016].

[19] editdistance. [Võrgumaterjal]. Kättesaadav: https://github.com/aflc/editdistance. [Viimati vaadatud 10 mai 2016].

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, Ando Haldna(isikukood 39111142714),

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose **Emotikonide tuvastamine tekstis,** mille juhendaja on Heiki-Jaan Kaalep,

    1.1. reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

    1.2. 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, **11.05.2016**