Tampereen yliopisto

Selja Vanamo

PÄIVÄYSTEN POIMINTAA

Automaattisten ja manuaalisten menetelmien vertailua digitoidussa historiallisessa kirjeaineistossa

Informaatioteknologian ja viestinnän tiedekunta
Pro gradu -tutkielma
Maaliskuu 2019

TIIVISTELMÄ

Selja Vanamo: Päiväysten poimintaa. Automaattisten ja manuaalisten menetelmien vertailua digitoidussa historiallisessa kirjeaineistossa.
Pro gradu -tutkielma
Tampereen yliopisto
Informaatioteknologian ja viestinnän tiedekunta
Informaatiotutkimuksen ja interaktiivisen median tutkinto-ohjelma
Maaliskuu 2019

Tutkimuksessa tarkasteltiin sitä, miten historiallisesta digitoidusta aineistosta pystytään etsimään päivämääriä automaattisin menetelmin. Koska historiallisten dokumenttien digitointia tapahtuu jatkuvasti, ja enenevästi myös hyvin monenlaisia aineistotyyppejä muutetaan digitaaliseen muotoon, on samalla tarpeen kehittää erilaisia tietoteknisiä menetelmiä, joiden avulla digitoituja aineistoja pystytään käsittelemään. Tutkimuksen aineistona oli noin tuhat talvi- ja jatkosodan aikaista digitoitua kirjettä, jotka sisältävät viiden eri yksityishenkilön kirjeenvaihtoa. Kirjeitä tarkasteltiin niiden päiväysten pohjalta, sillä tarkoituksena oli selvittää, millä tavoin kahden eri automaattisen menetelmän avulla olisi mahdollista poimia koko tutkimusaineistosta tietyn ajanjakson kirjeet. Tutkimusta varten koko laajasta digitoitujen kirjeiden kokoelmasta muodostettiin tämä pienempi kokeellinen testikokoelma, johon suoritettiin kolme erilaista testihakua. Vertailukohtana toimi itse manuaalisesti läpikäyty tulkinta jokaisen kirjeen päiväyksestä, ja hakutuloksia arvioitiin tarkkuuden ja saannin osalta.

Tutkimuksessa päiväystä lähestyttiin osaltaan nimettyjen entiteettien kautta, sillä päivämäärä on yksi nimetyistä entiteeteistä, joka on entiteettitunnistimien avulla mahdollista merkitä tekstiin. Vertailtavana menetelmänä tutkimuksessa käytettiin suomen kielelle kehiteltyä nimettyjen entiteettien tunnistinta nimeltään FiNER, jonka avulla tutkimusaineistosta oli mahdollista poimia tarkasteluun ne kirjeet, joihin oli merkitty päiväysentiteettitunniste. Toisena vertailtavana menetelmänä oli itse kehitelty Python-ohjelmointikielinen hakukoneen kaltaisesti toimiva ohjelma, jonka avulla kirjeitä poimittiin koko tutkimusaineistosta. Myös FiNERin merkitsemille kirjeille oli tulosten saamiseksi tarpeen hyödyntää tätä itse kehiteltyä hakukonetta hieman muokattuna, jolloin tarkasteluun tulivat vain päiväysentiteetin saaneet kirjeet.

Tutkimuksessa havaittiin, että FiNER tunnistaa kirjeiden päiväyksiä varsin huonosti eli entiteettitunnisteita merkittiin koko aineistolle vain vähän, minkä lisäksi tunnisteista suurin osa sijaitsi muualla tekstissä kuin varsinaisen päiväyksen kohdalla. Tällä oli vaikutuksensa hakutuloksiin, sillä kahden eri menetelmän tarkastelemassa kohdeaineistossa oli varsin suuri ero. Kirjeitä etsittiin vuoden, vuoden ja kuukauden sekä tarkan päivämäärän avulla. Automaattisin keinoin kirjeiden päiväykset löytyivät varsin hyvin, ja itse kehitellyllä menetelmällä hakutulosten saanti pysyi kohtalaisen hyvänä eli relevantit kirjeet löytyivät. Tarkkuus vaihteli hakujen välillä ollen paikoitellen varsin huono johtuen mukaan tulleista epärelevanteista osumista. Kautta linjan FiNERin tulokset olivat niin tarkkuuden kuin saannin osalta huonommat, mikä johtui siitä, etteivät kaikki relevantit kirjeet olleet saaneet päiväysentiteettitunnistetta tekstiinsä. Päiväyksen merkintätavoissa oli varsin suurta vaihtelua, eikä FiNER tunnistanut kuin tietynlaisen päiväyksen.

Tutkimuksen perusteella tultiin siihen tulokseen, että tietoteknisiä menetelmiä olisi syytä parannella ja kehittää, jotta niiden avulla digitoidut aineistot olisivat mahdollisimman käytettäviä. Tietojen etsimisessä erilaiset tiedonlouhintamenetelmät ovat hyvä apu, minkä lisäksi päiväysten mieltäminen nimetyksi entiteetiksi voisi auttaa niiden etsimisessä, sillä tällöin entiteettitunnisteen avulla kirjetekstistä olisi helpompi saada päiväys poimittua. Menetelmien ja tunnistimien parantelu on kuitenkin tarpeen, jotta useammat erilaiset variantit tunnistettaisiin myös. Digitoitujen aineistojen käsittelyssä ja tietojen etsimisessä tietoteknisten menetelmien kehittely ja parantaminen helpottaisivat laajasti eri alojen tutkijoiden työtä ja aineistojen käytettävyyttä, minkä vuoksi siihen tulisi panostaa aina vain enemmän.

Avainsanat: nimetyt entiteetit, annotointi, digitoitu historiallinen aineisto, tiedonhaku, testikokoelma, sota-ajan kirjeet

Sisällysluettelo

1 JOHDANTO

Digitalisaatio on viimeisten vuosikymmenten ja erityisesti muutaman viime vuoden aikana lisääntynyt jatkuvasti. Se on soluttautunut osaksi jokapäiväistä elämäämme monessakin eri muodossa, ja arkielämän lisäksi digitalisaatiolla on ollut vaikutuksensa myös tieteelliseen tutkimukseen. Yhä useammalle mahdollistuu pääsy vanhoihin historiallisiin arkistoaineistoihin, sillä fyysinen sijainti tai kellonaika eivät enää muodostu ongelmaksi, kunhan käsillä on tietokone tai jokin muu laite ja siinä internet-yhteys. Perinteisesti historiallinen arkistoaineisto on sisältänyt erilaisia asiakirjoja, valokuvia tai muita virallisempia dokumentteja, joita viime vuosien aikana on digitoitu, mutta yhä enenevässä määrin myös muunlaista historiallista aineistoa, kuten kirjeitä, muutetaan digitaaliseen muotoon. Yksityishenkilöiden kirjoittamat kirjeet talvi- ja jatkosodan ajalta ovat yksi mielenkiintoinen aineistomuoto, sillä niiden avulla muun muassa historian- ja kulttuurintutkijoiden on mahdollista päästä tarkastelemaan paremmin myös tavallisten ihmisten kokemuksia sodasta.

Niin sanottujen digisyntyisten eli pelkästään sähköisessä muodossa olevien aineistojen lukumäärä lisääntyy jatkuvasti, mutta maailmanlaajuisesti on ollut jo varsin pitkään käynnissä erilaisia digitointihankkeita, joiden tavoitteena on siirtää vanhoja ei-digitaalisessa muodossa olevia aineistoja digitaaliseen muotoon. Digitoinnin tarkoituksena on varmistaa, että mahdollisesti herkkäkin paperinen alkuperäisaineisto säilyisi mahdollisimman hyvin, sillä digitaalisessa muodossa materiaali kestää paremmin kulutusta ja on monin tavoin saavutettavampaa, eikä alkuperäiseen versioon kohdistu enää niin suurta käyttötarvetta. Toisaalta teknologian ja ohjelmistojen jatkuva kehitys aiheuttaa sen, että digitaalisten aineistojen arkistoinnissa on kiinnitettävä huomiota pitkäaikaissäilytyksen keinoihin, jotta aineistot pysyvät käytettävinä myös tulevina vuosikymmeninä.

Kun yhä suurempi osa paperilähteistä on nykyään myös digitaalisessa muodossa, on esimerkiksi tutkijoiden entistä helpompaa saada erilaisia aineistoja käyttöönsä. Jotta näiden historiallisten aineistojen käyttö olisi mahdollisimman laajaa ja monipuolista, tulisi digitoinnin ohella kehittää myös nimenomaan digitaalisille aineistoille sopivia käsittely- ja analyysimenetelmiä, sillä suurten tutkimusaineistojen tarkastelussa tietokone on monin tavoin ihmistä tehokkaampi. Toisaalta kone ei kykene tulkitsemaan asioita ainakaan vielä samalla tavoin kuin ihminen, joten ihmisten tekemä analyysi ja tulkinta on edelleen

tärkeää. Myös dokumenttien ja niiden sisältämien tietojen haettavuuteen on syytä kiinnittää huomiota, sillä muutoin aineiston käyttö muuttuu ongelmalliseksi.

Historiallisten aineistojen yksi tärkeimpiä tehtäviä on pystyä sijoittamaan kyseinen lähdemateriaali johonkin tiettyyn ajalliseen kontekstiin, jotta tulkinta ja tarkastelu on ylipäätään kunnolla mahdollista, sillä aineisto on kuitenkin aina syntyhetkeensä sidoksissa. Erityisesti kirjeisiin on kautta aikain lähes poikkeuksetta merkitty näkyviin kirjoitushetken päivämäärä. Tässä tutkielmassa tarkastellaan pientä otosta Suomen talvi- ja jatkosodan aikaisten kirjeiden digitoidusta kokoelmasta. Tällä hetkellä vasta murto-osa kaikista paperisista sota-ajan kirjeistä on digitoitu. Digitoinnissa kirje on skannattu ja sen teksti on optisesti tunnistettu (OCR, Optical Character Recognition), minkä jälkeen digitoitu kirje on muutettu edelleen myös tekstitiedostoksi. Tämän tutkimuksen käytössä on ollut pelkästään kyseistä tekstitiedostona olevaa aineistoa, ja koska digitoituja kirjeitä on sen verran paljon, on niistä poimittu otos testikokoelmaksi tutkimusta varten. Tämä testikokoelma sisältää 992 yksittäistä kirjettä viidessä yksittäisen henkilön kirjeenvaihtoa sisältävässä osakokoelmassa. Testikokoelmaan suoritetaan automaattisia tiedonhakuja, joiden tuloksia sitten arvioidaan.

Tässä tutkielmassa tarkastellaan sitä, millä tavoin kaksi erilaista automaattista tietoteknistä menetelmää vertautuvat ihmisen tekemään tulkintaan siitä, miten historiallisesta digitoidusta kirjeaineistosta on mahdollista poimia kirjeen päivämäärä. Päiväysten tunnistaminen ja poiminta liittyy osaltaan nimettyjen entiteettien tunnistamiseen, jota varten on kehitelty erilaisia menetelmiä ja tunnistimia. Yhtenä automaattisena menetelmänä tässä tutkielmassa on tarkastelussa FiNER, joka on suomen kielelle kehitelty nimettyjen entiteettien tunnistin. FiNER tunnistaa ja merkitsee tekstiin paitsi henkilöiden, yritysten ja paikkojen nimiä, myös päivämääriä, jotka ovat tämän tutkimuksen kiinnostuksen kohteena. Toisena tietoteknisenä menetelmänä on itse kehitelty Python-ohjelmointikielellä luotu ohjelma, joka on suunniteltu tutkimusaineiston ominaisuudet mahdollisimman hyvin huomioiden. Tämän ohjelman avulla tutkimusaineistosta kootaan tulosdokumenttiin halutun päiväyksen sisältämät kirjeet. Muokattuna tätä itse kehiteltyä ohjelmaa hyödynnetään myös hakukoneen kaltaisesti FiNERin avulla merkatulle aineistolle, jotta siitäkin on mahdollista saada etsittyä tietoja. Menetelmien vertailussa hyödynnetään tekstin lukemiseen ja omaan tulkintaan perustuvaa annotointia, jossa kirjetekstille on pyritty luomaan tiedonhakua helpottavaa rakennetta ja jossa päiväykset on poimittu myös metadataksi selkeään yhdenmukaiseen muotoon tulosten helpompaa relevanssiarviointia varten.

Juuri tällaista tutkimusta aineisto ja menetelmät huomioiden historialliselle kirjeaineistolle ei ole aiemmin tehty, vaikka päiväys onkin olennainen tieto erityisesti juuri kirjeissä sekä laajemminkin historiallisessa aineistossa. Koska digitoitu kirjeaineisto on hyvin laaja, on yksittäisen ihmisen melkein mahdotonta käydä koko aineistoa läpi. Tältä pohjalta automaattisten menetelmien kehittäminen on tärkeää, sillä on oletettavaa, ettei mikään tällä hetkellä olemassa oleva automaattinen päiväysten poiminnan mahdollistava menetelmä ole vielä kovin tehokas ainakaan suomenkielisessä tekstissä ja suomalaisella tavalla ilmaistun päiväyksen suhteen. Myös tekstissä olevat mahdolliset tunnistus- eli OCR-virheet sekä päiväyksen kirjoituskäytäntöihin liittyvät seikat aiheuttavat oletettavasti omat hankaluutensa tiedon etsimiseen, joten on tärkeää tarkastella myös syitä, miksi haluttua tietoa ei aineistosta välttämättä löydy ja mitä asialle olisi mahdollista tulevaisuudessa tehdä.

Luvussa 2 tarkastellaan teoreettista viitekehystä eli tämän tutkimuksen näkökulmaa sekä keskeisimpiä käsitteitä, ja luvussa 3 perehdytään aiempaan aiheeseen liittyvään tai sitä sivuavaan tutkimukseen. Luvussa 4 esitellään tämän tutkimuksen kannalta olennaiset tutkimuskysymykset, käytössä oleva tutkimusaineisto sekä tutkimusmenetelmät. Luvussa 5 käydään läpi tutkimuksen tulokset ja vertaillaan eri menetelmiä keskenään. Luku 6 kokoaa yhteen johtopäätöksiä tehdystä tutkimuksesta, tuo esiin ajatuksia esiin nousseista ongelmista sekä tarjoaa ehdotuksia mahdollisia uusia tutkimus- ja kehityskohteita ajatellen.

2 TEOREETTINEN VIITEKEHYS

Tässä luvussa esitellään tutkimuksen teoreettinen viitekehys eli tutkimuksen näkökulma ja rajaukset sekä keskeisimmät käsitteet. Alaluvussa 2.1 käsiteltävät käsitteet metadata ja annotointi liittyvät olennaisesti siihen, miten aineistot ja dokumentit saadaan haettaviksi, sillä niiden avulla aineistoille saadaan muodostettua rakennetta ja niihin voidaan liittää erilaisia lisätietoja tiedonhakua helpottamaan. Tietojen hakemiseen liittyvät olennaisesti myös alaluvussa 2.2 esitellyt nimetyt entiteetit, joita ovat esimerkiksi nimet, paikat sekä päiväysten kaltaiset numeraaliset tiedot. Alaluvussa 2.3 tutustutaan yleisemmin historiallisiin digitoituihin aineistoihin. Alaluvussa 2.4 tarkastellaan testikokoelman rakentamista kokeellisempaa tutkimusasetelmaa varten sekä tulosten arviointia erityisesti tarkkuuden ja saannin osalta. Luvussa 2.5 puolestaan kiinnitetään hieman huomiota luonnollisen kielen ominaisuuksiin ja perustellaan Python-ohjelmointikielen valintaa.

2.1 Metadata ja annotointi

Tässä tutkielmassa *metadata* ja siihen hyvin pitkälti liittyvä *annotointi* ovat tärkeitä termejä, koska niiden avulla tutkimuksessa tarkasteltavalle aineistolle on pyritty antamaan tiedonhakua helpottavaa rakennetta, sillä sitä ei ollut ennestään olemassa. Metadatalle määritelty tärkein tehtävä on kuvata tietoresurssin eli dokumentin sisältöä tiivistetyllä ja strukturoidulla eli rakenteistetulla tavalla (Foulonneau & Riley 2008, 3). Annotointikin liittyy kiinteästi rakenteen luomiseen, joskin sen merkittävin tehtävä on pohjimmiltaan määritellä ja luokitella dokumentin sisältämiä palasia, kun taas metadatan avulla on mahdollista poimia esiin tärkeitä asioita ilman tarkempaa luokittelua. Lisäksi metadata sijaitsee käytännössä aina varsinaisen tekstin ulkopuolella tai jopa siitä erillään, kun taas annotointia voi lisätä niin tekstin sisään kuin sen ulkopuolellekin.

Metadatan määrittely on varsin haastavaa, sillä sen voi määritellä joko hyvin suppeasti tai erittäin laajasti ja ympäripyöreästi. Perinteisimmillään metadataa kuvaillaan ”datan dataksi” eli sen avulla pyritään kuvailemaan yleisluontoisesti jonkin dokumentin sisältöä. Metadatan avulla dokumentille voi luoda rakennetta, ja toisaalta tiettyjä dokumentin osasia pystyy nostamaan muita merkittävimpinä esille. Tällaisia kohteita ovat esimerkiksi dokumentin kirjoittaja tai luoja sekä päivämäärä. (Davies et al. 2009, 187.) Metadata liittyy nykyään keskeisesti internet-ympäristöön, jossa sen tarkoituksena on parantaa

relevanttien dokumenttien tarkkuutta eli esiintyvyyttä hakutuloksissa. Toisaalta myös esimerkiksi kirjastojen luettelointikäytännöt ovat pohjimmiltaan metadatan luomista, joten termi ei liity pelkästään digitaalisiin dokumentteihin, vaikka suurin hyöty sillä tässä yhteydessä onkin. (Nagao 2003, 61–62.)

Nagao (2003, 61) käyttää metadatasta mieluummin termiä *annotaatio*, jolla hän tarkoittaa sisällön kontekstiin tai tarkoitukseen liittyvää lisätietoa. Perinteisesti annotoinnilla tarkoitetaan lähinnä kommenttia tai selitystä, minkä lisäksi sillä voidaan viitata myös tekstin kielelliseen analyysiin. Yleisesti ottaen annotointi on kuitenkin varsin tarkemmin määrittelemätön ja monikäyttöinen käsite. (Nagao 2003, 61.) Erityisesti luonnollisen kielen prosessoinnissa annotaatiota pidetään usein synonyyminä metadatalle. Tekstidokumenttien annotoinnin voi jakaa kahteen osa-alueeseen, joista *dokumenttitason annotointi* viittaa koko dokumenttiin ja *merkkitason annotointi* puolestaan dokumentin palaseen, joka on merkitty selkein alku- ja loppumerkein. Yleisimmin annotoinnista puhuttaessa viitataan merkkitason annotaatioon. Toinen keino jaotella annotointia on se, kuinka ne liitetään tekstiin. Ensimmäinen tapa on sisäinen merkintä (embedded markup), jota käytetään erityisesti HTML-merkintäkielessä alku- ja lopputagien muodossa. Toinen tapa on ulkoinen viittaus, jossa annotoinnit pidetään erillään itse dokumentista ja kytketään siihen linkkien avulla. (Davies et al. 2009, 187–188.)

Annotointi voi olla luonteeltaan myös semanttista, jolloin se liittyy tavallisimmin semanttiseen webiin. Yleisesti ottaen *semanttista annotointia* voi hyödyntää erityisesti nimettyjen entiteettien suhteen (SANE, Semantic Annotation of Named Entities), jolloin spesifiä metadataa luodaan sillä ajatuksella, että entiteetti viittaa dokumentin semanttisesti tärkeään informaatiopalaseen. (Davies et al. 2009, 189–192.) Tällaisia palasia ovat esimerkiksi henkilöiden ja paikkojen nimet sekä päivämäärät.

Annotoinnin yksi osa-alue on *temporaalinen annotointi*, jossa erilaisia temporaalisia eli ajallisia ilmauksia pyritään annotoimaan tekstiin. Tällaisia ovat tarkat ajalliset ilmaukset, kuten päiväykset, kestot ja frekvenssit eli taajuudet, jotka ilmaisevat, kuinka usein jokin asia tapahtuu. Temporaaliset ilmaukset voivat olla luonteeltaan varsin yksiselitteisiä ja tarkkoja, muihin tapahtumiin liittyen suhteellisia tai sitten hyvinkin epämääräisiä, mikä saattaa hankaloittaa tulkintaa. Ajallisten ilmausten annotointi ei itsessään tuo juurikaan informaatiota tekstin sisällöstä, vaan huomio on enemmän ajallisten yhteyksien välisessä tarkastelussa ja aikajanan luomisessa. Temporaalisessa annotoinnissa huomio ei kohdistu

pelkästään tarkkoihin ajallisiin ilmauksiin, vaan myös esimerkiksi lauseiden verbeihin, jotka kertovat omalta osaltaan tarkempaa tietoa tapahtumista ja tilanteista, jotka kaikki tapahtuvat ajassa. Vaikka ihmisten tekemää annotointia pidetäänkin monin tavoin parhaana ja luotettavimpana tulkintana, on tehokkuuden kannalta kuitenkin tärkeää kehittää myös temporaalisia taggereita eli tunnistimia, jotka pystyvät käsittelemään nopeammin suurempia aineistomääriä. (Mani et al. 2005, 487–504.) Temporaalinen annotointi tarjoaakin parhaimmillaan hyvin monipuolisen tarkastelutavan tekstiin ja sen ajalliseen kontekstiin, ja monien muiden tekstityyppien lisäksi sitä olisi mahdollista soveltaa myös kirjeisiin, jotka sisältävät paljon ajallisia viittauksia menneisiin ja tuleviin tapahtumiin.

2.2 Nimetyt entiteetit ja niiden tunnistus

Nimetyt entiteetit (NE, Named Entities) liittyvät olennaisesti luonnollisen kielen käsittelyyn (NLP, Natural Language Processing) ja edelleen erityisesti informaation eristykseen ja poimintaan, joka on tiedonhaun perusta (Davies et al. 2009, 192). Nimettyjen entiteettien käyttö sekä niiden tunnistukseen ja luokitteluun (NERC, Named Entity Recognition and Classification) liittyvä tutkimus alkoivat vähitellen vuoden 1996 jälkeen, jolloin käsitteet lanseerattiin Yhdysvalloissa MUC-6-konferenssissa (The Sixth Message Understanding Conference) (Nadeau & Sekine 2007, 3). Alkuvaiheessa nimetyiksi entiteeteiksi laskettiin vain ihmisten, organisaatioiden ja paikkojen nimet, jotka ovatkin kaikkein tutkituimpia entiteettityyppejä. Niiden tunnisteena on ”Enamex”. Myöhemmissä konferenssissa entiteeteiksi määriteltiin myös numeraalista tietoa sisältävät ilmaukset, kuten päiväykset (”Timex”) sekä aikaan, rahaan ja prosenttiosuuksiin liittyvät ilmaukset (”Numex”). Vuonna 2003 lanseerattiin TIMEX2, joka sisältää standardin temporaalisten ilmausten annotoinnille ja normalisoinnille. Alkuvaiheessa kiinnostus kohdistui vain englanninkielisiin teksteihin, mutta vähitellen entiteettien tunnistusta on sovellettu ja kehitetty myös moniin muihin kieliin. (Nadeau & Sekine 2007, 4–6.)

Entiteettien tunnistaminen ja nimeäminen liittyy kiinteästi tiedonhaun muuttuneisiin tarpeisiin ja menetelmiin, kun merkkijonoajattelusta ja dokumenttilistauksista on menty viime vuosikymmeninä enemmän kohti selkeämpien objektien ja palasten etsimistä ja poimimista, sillä niihin on helpompi viitata. Vaikka entiteettiajattelu on tiedonhaussa muuttunut tärkeämmäksi viimeisen parin vuosikymmenen aikana, ulottuvat sen juuret kuitenkin 1970-luvulle, jolloin erilaisten tietokantojen yleistyessä tuli tarve järjestellä

informaatiota selkeiksi kokonaisuuksiksi. Nimetyt entiteetit viittaavat ennen kaikkea reaalimaailman selkeästi nimettäviin ja tunnistettaviin objekteihin, minkä vuoksi ne tulee erottaa abstraktimmista konsepteista siitä huolimatta, että niiden tekninen käsittely on hyvin samanlaista. (Balog 2018, 1–4.)

Nimettyjen entiteettien tunnistuksen eli NERCin perimmäisenä tavoitteena on pystyä tunnistamaan myös aiemmin tuntemattomia entiteettejä, joten siinä missä alkuvaiheen kehitystyö perustui pitkälti ihmisten käsityönä tekemiin säännöstöihin, hyödynnetään nykyään pääasiassa algoritmeihin perustuvaa koneoppimista, joka mahdollistaa tehokkaamman automaattisen käsittelyn. Automaattisia menetelmiä kehitetään pääasiassa erilaisten harjoituskokoelmien avulla, mutta viime kädessä tulosten arviointi pohjautuu toistaiseksi edelleenkin manuaalisiin ihmisten tekemiin ja tulkitsemiin menetelmiin. (Nadeau & Sekine 2007, 7–8.)

Nimettyjen entiteettien merkitseminen ja hyödyntäminen on mahdollistanut myös *entiteettiorientoituneiden tiedonhakumenetelmien* kehittelyn, joiden juuret ovat TREC-konferensseissa (Text Retrieval Conference) vuosituhannen vaihteessa kehitellyissä kysymys- ja vastausmenetelmissä (QA, Question Answering). Niiden tavoitteena on pyrkiä mahdollistamaan tiedonhaussa hakusanojen lisäksi myös kysymysten esittämisen, jolloin kysymyksessä mainittu asia määritellään käytännössä entiteetiksi, johon pyritään löytämään vastaus. Tässä entiteettien tunnistus ja merkitseminen on tärkeässä osassa, ja 2010-luvulle tultaessa *entiteettitiedonhaku* (entity retrieval) oli jo alkanut muotoutua. Entiteettiorientoitunut tiedonhaku keskittyy yksinkertaistetusti löytämään informaatiota, joka ilmenee ennen kaikkea entiteetteinä sekä niitä määrittelevinä attribuutteina ja näiden välisinä suhteina. Tässä muun muassa tiedonhakijoiden tiedontarpeiden huomioiminen ja erilaisten entiteettilinkkien ja -tietokantojen rakentaminen ovat merkittävässä roolissa. (Balog 2018, 7; 11.) On oletettavaa, että entiteetteihin keskittyminen niin informaatiopalasten merkkauksessa eli annotoinnissa kuin tiedonhaussakin tulee jatkossa vain lisääntymään, minkä vuoksi menetelmien testausta ja kehittelyä eri kielillä on tärkeää jatkaa.

2.3 Historiallinen digitoitu aineisto

Eri-ikäisten historiallisten aineistojen digitointia on tehty kiihtyvällä tahdilla niin Suomessa kuin muuallakin maailmassa. Pääasiassa eri maiden kansalliskirjastot ja -arkistot ovat olleet vastuussa näistä digitointiprojekteista. Koko Euroopan laajuisessa Europeana-

projektissa on digitoitu erityisesti vanhoja sanomalehtiä, joista vuonna 2012 oli arviolta digitoitu jo yli 128 miljoonaa sivua yhteensä noin 24 000 nimekkeestä. Gutenberg-projektissa on digitoitu viime vuosina yli 50 000 kirjaa. Suomessa Kansallisarkisto on digitoinut monentyyppisiä historiallisia aineistoja, joista oli digitoituna vuonna 2016 jo yli 42 miljoonaa yksittäistä dokumenttia. Suomen Kansalliskirjasto oli digitoinut sanomalehtiä vuonna 2016 yli 10 miljoonaa sivua, joista vuosien 1771–1910 välinen aineisto eli noin kolme miljoonaa sivua on vapaasti käytettävissä. (Kettunen et al. 2016a, 3.) Määrät ovat valtavia, mutta eivät siitä huolimatta kata kuin murto-osan paperimuodossa olevasta arkistomateriaalista.

Digitoitujen lähdemateriaalien mahdollisimman hyvän käytettävyyden takaamiseksi tietokoneavusteisia aineistojen louhinta- ja analysointimenetelmiäkin tulee kehittää samaa tahtia digitoinnin kanssa, ja kohdeaineiston ominaisuudet tulisi tässä myös huomioida mahdollisimman hyvin. Kimmo Elon (2016) mukaan tietokoneavusteiset menetelmät tarjoavat mahdollisuuden tutkia näitä historiallisia aineistoja myös sellaisista näkökulmista, joiden soveltaminen ei aiemmin ole ollut mahdollista. Tieteellisessä tutkimuksessa monitieteisyys ja eri tutkimusryhmien yhteistyö korostuvat, kun perinteisemmät humanistiset tieteet hyödyntävät entistä enemmän myös tietokoneavusteisia tutkimusmenetelmiä. Toisaalta on kuitenkin muistettava, että tietokoneavusteiset menetelmät ovat vain välineitä, ja tulosten tulkinnassa tarvitaan edelleenkin ihmisen näkemystä. (Elo 2016, 8.)

Erilaiset digitointihankkeet ovat yleistyneet voimakkaasti, ja nämä mahdollistavat paremman ja helpomman pääsyn historiallisen lähdeaineiston pariin lähestulkoon missä ja milloin vain, kun ei enää ole välttämätöntä mennä fyysiseen arkistoon selaamaan kiinnostuksen kohteena olevaa materiaalia. Toisaalta täytyy pitää mielessä, että kaikesta vanhasta arkistoaineistosta on edelleenkin digitoitu vain pieni murto-osa (Elo 2016, 12), joten digitaalisessa muodossa olevat vanhat aineistot eivät vielä ole kovin kattavasti edustettuna kokonaiskuvaa ajatellen.

2.4 Testikokoelman rakentaminen ja tulosten arviointi

Tiedonhaun tarkoituksena on löytää jonkin tiedontarpeen pohjalta muodostettujen hakukyselyiden avulla relevantteja osumia eli löytyneiden dokumenttien sisältämän informaation tulisi olla mahdollisimman käyttökelpoista kyseessä olevaan tiedontarpeeseen nähden. Useimmiten *relevanssi* jaotellaan käyttäjä- ja aiherelevanssiin eli siihen, onko

hakutulos oikeanlainen käyttäjän vai hakuaiheen kannalta, sillä toisinaan niiden välillä voi olla varsin suurikin eroavaisuus muun muassa sanojen monimerkityksisyyden vuoksi. (Järvelin & Sormunen 2010, 164–166.) Tässä tutkielmassa tiedontarpeet liittyvät pelkästään kirjeiden päivämääriin, joten sinänsä monitulkintaisuudesta ei ole vaaraa eikä eri relevanssinäkökulmia ei ole tarpeen tällä kertaa erottaa toisistaan, sillä tietyn etsityn ajanjakson kirjeet ovat relevantteja niin käyttäjän kuin aiheenkin kannalta.

Tiedonhaussa tulosten evaluointi eli arviointi edellyttää tyypillisesti *testikokoelman* käyttöä. Testikokoelma rakentuu siten, että se sisältää jonkin tietokokonaisuuden eli havaintoaineiston, pienen määrän tiedontarpeita eli valmiita kyselyitä tai aiheita sekä relevanssikorpuksen eli ihmisen tarkistamat niin sanotut ”oikeat vastaukset” (englanniksi gold standard tai ground truth). Virallisempia ja perinteisempiä testikokoelmia on muodostettu TREC-konferensseissa (Text Retrieval Conference), joista ensimmäinen järjestettiin Yhdysvalloissa vuonna 1992. Sen jälkeen kokoelmia on kehitelty ja paranneltu vuosittain kulloistenkin esiin nousseiden erilaisten tarpeiden pohjalta. (Balog 2018, 88.)

Kaikkien testikokoelmien taustalla ovat niin sanotut Cranfield-kokoelmat, jotka kehiteltiin jo 1960-luvulla. Ne ovat pohjimmiltaan staattisia rakennelmia, joiden tarkoitus on standardoida hakutulosten vertailua, vaikka testikokoelmien tulosten suora soveltaminen oikeisiin reaalimaailman hakutuloksiin ei välttämättä aina onnistukaan. Kaikkien testikokoelmien suhteen on pohdittava sitä, millainen määrä dokumentteja ja hakuaiheita on riittävä tulosten luotettavuuden kannalta. Vaikka perinteisessä tiedonhaun tutkimuksessa Crandfield-testikokoelmien hyödyntäminen onkin kaikkein yleisintä, eivät ne kuitenkaan välttämättä ole liian abstrakteina hyödynnettävissä soveltavampiin hakutehtäviin, mitä varten toisinaan on tarpeen kehitellä uudenlaisia testikokoelmia. (Voorhees 2008, 1879–1881.) Esimerkiksi temporaalisen tiedonhaun ja sen tutkimuksen yleistyessä voi olla tarvetta rakentaa sopivampia testikokoelmia hakutulosten relevanssin arvioimista varten, sillä aikaan sekä muuhun temporaalisuuteen eli ajallisuuteen liittyviä seikkoja ei perinteisemmissä testikokoelmissa ole välttämättä juurikaan huomioitu (Joho et al. 2016, 677).

Koska tässä tutkimuksessa käytetty aineisto on vielä toistaiseksi varsin tutkimatonta erityisesti digitaalisessa muodossa, eikä aineisto sijaitse toistaiseksi vielä missään digitaalisessa arkistossa, on tutkimusta varten muodostettu eräänlainen pieni kokeellinen testikokoelma, joka sisältää määrällisesti noin seitsemäsosan koko digitoidun kirjekokoelman aineistosta. Koska kirjeaineisto on keskenään hyvin samankaltaista muun muassa

kirjoitusajankohtien suhteen, voinee testikokoelmasta tehtyjä havaintoja yleistää koko aineistoa koskeviksi, sillä päiväysten ilmaustavat ovat varsin samankaltaisia huolimatta eri ihmisten henkilökohtaisesta kirjoitustyylistä.

Hakutulosten yksinkertaisimmat arviointikriteerit liittyvät tarkkuuteen ja saantiin. Hakua tehdessä tulokset jakautuvat löydettyihin sekä hylättyihin ja edelleen relevantteihin sekä epärelevantteihin dokumentteihin. Näiden pohjalta muodostuu eräänlainen nelikenttä, sillä sekä löydetyt että hylätyt hakutulokset voivat sisältää niin relevantteja kuin epärelevanttejakin dokumentteja. Hakutuloksen *tarkkuus* kuvaa sitä, kuinka suuren osan relevantteja dokumentteja tulos sisältää. *Saanti* puolestaan kuvaa sitä, kuinka suuri osa tietokannan sisältämistä relevanteista dokumenteista löydettiin. Sekä tarkkuus että saanti esitetään yleensä joko desimaalilukuna välillä [0, 1] tai prosenttilukuna välillä 0–100 %. (Järvelin & Sormunen 2010, 166–173.) Useimmiten saantia on mahdotonta saada tietoonsa käytännön tiedonhaussa, mutta tämän tutkielman osalta myös saanti on hyvä arviointikriteeri, sillä relevanttien osumien eli kiinnostuksena olevien päiväysten lukumäärä on manuaalisen tarkastelun ansiosta tarkkaan tiedossa.

Hakutulosten arviointia varten on kehitelty hyvin monenlaisia erilaisia menetelmiä, kuten keskitarkkuus, keskitarkkuuden keskiarvo tai tarkkuus jossakin tietyssä pisteessä. Näiden hyödyntäminen itse koostetussa varsin suppeassa testikokoelmassa eivät ole tarkoituksenmukaisia, sillä tuloksia pystyy arvioimaan varsin hyvin myös pelkän tarkkuuden ja saannin avulla, sillä tarkat tulokset ovat tiedossa manuaalisen läpikäynnin ansiosta. Koska kirjeet kerätään tulostiedostoon id-numeron mukaisessa numeraalisessa järjestyksessä, ei relevantin osuman tarkempi analysointi hakutuloksessa sijainnin suhteen ole tarpeen, sillä hakutulokset eivät järjesty relevanssin, vaan id-numerojärjestyksen mukaan.

2.5 Lyhyesti luonnollisesta kielestä ja Pythonin valinnasta

Suurin osa historiallisista lähdeaineistoista on tekstiä, joka on siis kirjoitettu jollakin *luonnollisella kielellä*. Erityisesti suomen kaltaisen taipuvan ja morfologisesti monimutkaisen kielen suhteen tämä aiheuttaa automaattiseen tiedonhakuun ja tietokoneavusteisten menetelmien hyödyntämiseen varsin monenlaisia ongelmia. Suurin ongelma tulee hakusanojen eli kyselyn sekä dokumenttien sisällön väliseen vuorovaikutukseen, jos sanat eivät sellaisenaan kohtaa toisiaan, jolloin relevantteja tuloksia saattaa jäädä löytymättä. Toisaalta liian laajoilla hakusanoilla saattaa mukaan osua runsaasti epärelevanttejakin

osumia. Tällaisen niin sanotun historiallisen tiedonhaun suhteen voidaan hyödyntää monenlaisia kieltenväliseen tiedonhakuun kehiteltyjä menetelmiä, sillä varhaisemmassa ja nykykielessä on usein monenlaisia eroavaisuuksia. Tällaiset ongelmat eivät kuitenkaan liity sinänsä talvi- ja jatkosodan ajan kirjeisiin, sillä niissä kieli on hyvin lähellä nykypäivän suomea. Toisaalta murteelliset ilmaukset saattavat kuitenkin hankaloittaa sanojen tunnistumista, sillä kirjeiden kieli voi paikoitellen olla hyvin puhekielimäistä, ja kieliteknologiset menetelmät on kehitetty kuitenkin ensisijaisesti kirjakielelle. Puhekielisten ilmausten löytyminen hakusanojen avulla voi siis myös olla haastavaa. Käytännössä tämä ongelma ei kuitenkaan liity kirjeiden päiväyksiin, jotka koostuvat useimmiten numeroista, eikä murteellisuus ilmene juuri kirjaimin kirjoitetuissa päiväyksissäkään. Kieleen liittyvät optisen tunnistuksen tai tiedonhaun ongelmat voivat kuitenkin ilmetä monin tavoin myös kirjetekstissä.

Luonnollisen kielen tietoteknistä käsittelyä varten on kehitelty runsaasti erilaisia menetelmiä ja työkaluja, joiden avulla dokumentteja on helpompi käsitellä. Erilaisia ohjelmointikieliä on runsaasti, ja luonnollisten kielten tavoin ne muistuttavat monin tavoin toisiaan, vaikka niiden välillä on kaikkien kielten tavoin myös eroavaisuuksia. Ohjelmointikielistä Python on yksi parhaimmista luonnollisen kielen käsittelyyn, sillä siinä on sisäänrakennettuna paljon muun muassa merkkijonojen eli sanojen käsittelyä helpottavia työkaluja ja muita ominaisuuksia. Lisäksi sen avulla on helppo saada käyttöönsä myös alun perin Pennsylvanian yliopistossa vuonna 2001 tietokonelingvistiikan kurssin puitteissa kehitelty NLTK (Natural Language Toolkit), joka tarjoaa monenlaisia kielen käsittelyn työkaluja (Bird et al. 2009, xii–xiv.) Tästä syystä ja myös aiemman omakohtaisen kokemuksen pohjalta Python valikoitui tämän tutkimuksen merkittäväksi apuvälineeksi, jonka avulla tutkimusaineistoa on mahdollista käsitellä ja tarkastella monipuolisesti, minkä lisäksi sen avulla oli mahdollista rakentaa eräänlainen hakukone, jonka avulla hakuaiheiden mukaiset sopiviksi katsotut kirjeet saadaan poimittua tulosdokumenttiin.

3 AIEMPI TUTKIMUS

Sota-ajan kirjeitä ei ole aiemmin tutkittu ainakaan Suomessa informaatiotutkimuksen tai tarkemmin tiedonhaun näkökulmasta, sillä aiempi tutkimus on tapahtunut enemmän historia- ja kielitieteiden piirissä, jolloin tutkimus on pohjautunut lähinnä alkuperäisten paperimuotoisten arkistomateriaalien tarkasteluun ja lukemiseen. Koska myös kirjeitä on alettu viime vuosina digitoida, tarjoaa se kiinnostavia tutkimusongelmia ja mahdollisuuksia myös muille tieteenaloille. Tieteidenvälinen ja monitieteinen yhteistyö olisikin kaikkein paras tutkimuksellinen lähtökohta parhaan mahdollisen lopputuloksen saavuttamiseksi, sillä informaatio- ja tietojenkäsittelytieteillä on tarjota uudenlaisia menetelmiä historiallisten digitoitujen aineiston käsittelyyn. Jonkin verran tätä yhteistyötä onkin jo alettu viime vuosina hyödyntää.

Aiemman tutkimuksen osalta alaluvussa 3.1 tarkastellaan yleisemmällä tasolla sitä, millaisia mahdollisuuksia temporaalinen annotointi voisi tarjota tutkimukseen. Alaluvussa 3.2 esitellään suomenkieliseen historialliseen sanomalehtikokoelmaan liittyvää nimettyjä entiteettejä tarkastelevaa tutkimusta, ja alaluvussa 3.3 SA-kuvakokoelmaan eli sota-ajan valokuviin liittyvää tekstinlouhintaa. Alaluvussa 3.4 tarkastellaan lyhyesti sitä, missä määrin sota-ajan kirjeitä on aiemmin tutkittu muissa yhteyksissä. Tarkastelu on rajattu vain Suomessa tapahtuneeseen suomenkieliseen aineistoon kohdistuvaan tutkimukseen, sillä vaikka sotien aikana on ympäri maailman kirjoitettu kirjeitä, ovat talvi- ja jatkosodan aikaiset kirjeet kuitenkin hyvin vahvasti suomalaiseen historiaan ja identiteettiin liittyviä luomuksia, ja toisaalta samaa aineistoa on hyödynnetty myös joidenkin muiden alojen tutkimuksessa.

3.1 Temporaalinen annotointi

Temporaaliset ilmaukset ovat merkittävimpiä luonnollisen kielen semanttisia sisältöjä, ja niiden automaattinen poimiminen on tärkeää tekstin ymmärtämisen kannalta. Tämän pohjalta on viime vuosien aikana kehitetty muun muassa relevanssikorpus TimeBank Corpus, joka sisältää eksplisiittisesti annotoitua temporaalista informaatiota. Tällaisten korpusten perimmäisenä tarkoituksena on pystyä tunnistamaan tekstistä automaattisesti aikaan ja tapahtumiin liittyviä kohtia, jotka tarjoavat olennaista tietoa tekstistä. (Setzer et al. 2005, 243–244.) Temporaalinen annotointi tarkoittaa siis sitä, että tekstistä tunnistetaan ja

tulkitaan ilmauksia, jotka viittaavat aikaan, kestoon ja frekvenssiin eli taajuuteen, minkä lisäksi tarkoitus on tunnistaa tapahtumia sekä eri asioiden välisiä ajallisia suhteita. Nämä ilmaukset voivat olla joko todella tarkkoja tai sitten hyvinkin epämääräisiä. (Setzer et al. 2005, 249.) Kehiteltäessä automaattisia välineitä temporaaliseen annotaatioon, on manuaalisella annotointiprosessilla kuitenkin merkittävä rooli toimia ”gold standardina” eli vertailukohtana, joka sisältää mahdollisimman oikeat vastaukset ihmisen tulkitsemana, sillä automaattisesti ei välttämättä saada kiinni kaikkia tapauksia, jotka ihminen pystyy kuitenkin tunnistamaan (Setzer et al. 2005, 257). Menetelmien kehittely sekä erilaisten korpusten kattavuuden ja hyödynnettävyyden parantaminen vaativat kuitenkin vielä lisätyötä, sillä esimerkiksi annotointikäytännöt eivät ole vielä kovinkaan yhdenmukaiset (Setzer et al. 2005, 263).

TempEval on eräs menetelmistä, joiden tarkoituksena on evaluoida tekstistä temporaalisia yhteyksiä automaattisesti annotoivia systeemejä, ja se perustuu vuonna 2007 järjestettyyn SemEval-työpajaan, jossa temporaalisten yhteyksien merkitystä haluttiin kuitenkin korostaa aiempaa enemmän. TempEval hyödyntää TimeML-annotointikieltä sekä TimeBank-korpusta. Tarkoituksena oli identifioida tekstissä mainittuja tapahtumia ja sitoa ne aikaan, mikä parantaa luonnollisen kielen prosessoinnin menetelmiä ja siten edelleen muun muassa tiedonhakua. Huomiota kiinnitettiin kolmeen tärkeimpään temporaaliseen elementtiin: aikaan, tapahtumiin sekä temporaalisiin yhteyksiin, joista viimeisin nähtiin kaikkein tärkeimmäksi. Temporaalisia tekstissä mahdollisia ilmauksia ovat muun muassa kellonajat, päivämäärät, kestot ja frekvenssit, joiden pohjalta on mahdollista tehdä temporaalista annotointia, joka voi olla luonteeltaan joko tarkan absoluuttista tai suhteellisempaa. Luonnollinen kieli aiheuttaa annotoinnissa ongelmia sikäli, että osa temporaalisista ilmauksista voi olla hyvinkin epämääräisiä. Ihmisen on kuitenkin mahdollista tehdä temporaalisesta aikajanasta useimmiten parempia tulkintoja kuin koneiden, mutta toisaalta TempEval osoitti myös sen, että toisinaan automaattinen annotointi saattaakin tuottaa ihmistä parempia tuloksia suuremman kapasiteettinsa ansiosta. (Verhagen et al. 2009.)

Temporaalinen tiedonhaku on ollut viime vuosina nouseva tutkimusalue, sillä muun muassa internetin myötä datan määrä on kasvanut räjähdysmäisesti, ja sen informaatiosisältö on myös usein ajasta riippuvaista, jolloin esimerkiksi temporaalisten tunnistimien kehittäminen on tärkeää mahdollisimman relevanttien osumien löytämiseksi. Toistaiseksi menetelmien kehittely on kohdistunut pääasiassa englanninkielisiin uutisiin, mutta viime vuosina kehitystyötä on kohdistettu myös muihin kieliin. Erilaisten temporaalisten

tunnistimien tuloksia on mahdollista vertailla keskenään, sillä useimmat olemassa olevat tunnistimet hyödyntävät TIMEX2-merkkausta tai TimeML-annotaatiota, ja niiden avulla on mahdollista luokitella dokumenttien temporaalista dataa ja poimia edelleen relevanttia informaatiota. Erään tutkimuksen vertailussa oli neljä temporaalista tunnistinta: TempEx, GUTime, Annie sekä Heidel Time, joiden välillä ei todettu kuitenkaan suurtakaan eroa, vaan kukin tunnistin toimi varsin hyvin evaluointitulosten perusteella. (Yadav et al. 2015.)

Tiedonhaussa ja sen tutkimuksessa on yleisesti tunnustettu ajan ja ajallisuuden merkitys hakutulosten parantamisessa. Suurin osa temporaaliseen annotointiin ja erilaisten temporaalisten tunnistimien vertailuun liittyvästä tutkimuksesta on keskittynyt englanninkieliseen aineistoon. Huomio on kohdistettu lisäksi kaikenlaisiin ajallisiin ilmauksiin, joita tekstistä on mahdollista poimia ja tulkita, eikä vain tarkkoihin aikailmauksiin, kuten päivämääriin. Toisaalta olisi myös tärkeää ymmärtää ja löytää parhaimmat tekniikat niiden päiväysten löytämiseen, jotka puuttuvat tai ovat epäselviä. Historialliseen tai arkistoaineistoon liittyvää tutkimusta, joka kohdistuu ajallisten ilmausten etsimiseen, on kuitenkin tehty varsin vähän, ja suurin osa tästä tutkimuksesta on käyttänyt aineistonaan historioitsijoiden tekemiä manuaalisia annotointeja sen sijaan, että annotointi olisi tehty automaattisin menetelmin, mikä kuitenkin olisi tehokkuuden kannalta perusteltua. Jotta hakukyselyn ja tuloksen täsmäävyys olisi mahdollisimman hyvä, on erityisesti vuosiluku olennainen dokumentteihin liittyvä temporaalinen tieto, minkä vuoksi sen löytäminen ja annotointi olisi tärkeää. (Foley & Allan 2015.)

Temporaalisen annotoinnin hyödyntäminen kirjeaineistossa onnistuisi nähdäkseni paitsi yksittäisten kirjeiden sisällön osalta, myös laajemmin koko tietyn yksittäisen henkilön kirjeenvaihdon suhteen, sillä ajallisten ilmausten ja niiden välisten suhteiden löytyminen olisi tällaisesta kronologisesti etenevästä ja päiväkirjamaisesta vuoropuhelusta varsin yksiselitteistä. Temporaalisen annotoinnin keskiössä ovatkin kirjeiden päiväykset, sillä niiden perusteella koko aineisto on mahdollista järjestää kronologisesti ja sijoittaa historialliselle aikajanalle. Muut ajalliset tapahtumat ja ilmaukset lähinnä tukevat tätä ensisijaista tunnistamista. Erilaisia ajallisia ilmauksia tekstiin merkitseviä tunnistimia voisi tässä hyödyntää, ja yhtenä esimerkkinä tästä voisi olla suomen kielelle kehitelty FiNER.

3.2 Historiallinen suomenkielinen sanomalehtikokoelma

Digitoitua historiallista suomenkielistä aineistoa hyödyntävää nimettyihin entiteetteihin kohdistuvaa tutkimusta on tehty viime vuosina erityisesti sanomalehtikokoelmaan liittyen. Tämä historiallinen sanomalehtikokoelma sisältää yli 1,9 miljoonaa Kansalliskirjaston digitoimaa Suomessa julkaistua sanomalehtisivua vuosilta 1771–1910. Suurin osa aineistosta on suomenkielistä, ja Kimmo Kettusen ja muiden tutkimus (2016 sekä 2017) keskittyikin vain tähän suomenkieliseen materiaaliin. Tutkimuksessa vertailtiin muutamaa eri menetelmää, joiden avulla tekstistä pyrittiin löytämään nimettyjä entiteettejä. Vertailussa olivat muun muassa entiteettitunnistin FiNER sekä ARPA ja FST (Finnish Semantic Tagger), jotka eivät kuitenkaan ole varsinaisia entiteettitunnistimia, vaan toisenlaisia kieli- ja entiteettityökaluja. Kettusen ja muiden tutkimuksessa kiinnostuksen kohteena olivat erityisesti henkilöiden nimet ja paikat, joskin FiNER pystyy merkitsemään myös aikailmaisuja, kuten päivämääriä. Nimettyjen entiteettien tunnistusta vaikeuttavat tekstissä olevat OCR-virheet, joskin sanomalehtikokoelma sisältää oikein tunnistuneita sanoja noin 70–75 % kokonaismäärästä, mitä voidaan pitää aineiston luonne huomioiden kohtalaisen hyvänä tuloksena. Tutkimuksessa vertailtavien entiteettitunnistimien ongelmana oli se, että kaikkia niistä ei ole suunniteltu morfologisesti monimutkaiselle suomen kielelle, ja jos onkin, nykysuomi ja varhaisempi suomi eroavat toisistaan paikoitellen sen verran paljon, että nykysuomelle suunniteltu tunnistin ei välttämättä osaa käsitellä vanhempaa tekstiä kovin hyvin. (Kettunen et al. 2016b.) Tutkimuksessa käytettävästä aineistosta suurin osa ajoittui vuosiin 1890–1910, joten kieleltään lehtiartikkelit olivat lopulta lähempänä nykysuomea kuin kaikkein vanhin sanomalehtiaineisto, minkä vuoksi menetelmien käyttö ja vertailu kuitenkin oli mielekästä (Kettunen et al. 2016a).

Kettunen tarkasteli yhdessä muiden kanssa myös digitoidun sanomalehtikokoelman laatua lähinnä optisen tunnistuksen osalta, minkä pohjalta OCR-tunnistustulosta pyrittiin myöhemmin parantelemaan. Mitä laadukkaampaa teksti on, sitä helpompi siihen on muun muassa tehdä tietokoneavusteisia hakuja. Tiedonhaun koeasetelmissa jo 5–20 % virhemäärät heikentävät merkittävästi hakutuloksia, jolloin relevanttia aineistoa saattaa jäädä löytymättä, vaikka hakujärjestelmistä onkin toisaalta tehty kohtalaisen joustavia myös ”roskaisempaa” tekstiä ajatellen esimerkiksi sumeiden menetelmien avulla (Kettunen et al. 2016a, 5). Kansalliskirjaston suomenkielisen lehtiaineiston laadun arviointiprojekti aloitettiin vuonna 2014, jolloin sanat analysointiin ensin kahdella nykysuomen

morfologisella tunnistusohjelmalla Omorfilla ja FINTWOLilla. Tunnistettujen ja tunnistamattomien sanojen määrää arvioitiin, ja parhaimmissa digitoiduissa aineistoissa jopa noin 90 % sanoista oli oikeita ja huonoimmissakin noin 60 %. Nykysuomelle kehitettyjä tunnistusohjelmia voi kuitenkin käyttää myös erityisesti 1800-luvun jälkipuolta myöhemmän vaiheen tekstiin, sillä lopputulos on tällöin tarpeeksi hyvä. Laatua arvioitaessa perimmäinen tarkoitus on myös kehittää sitä parantavia menetelmiä, joita ovat lähinnä uusi optinen luku sekä ohjelmallinen jälkikorjaus, jotka kuitenkin vaativat lisää sekä ajallisia että työresursseja, eivätkä sen vuoksi ole aina mahdollisia. (Kettunen et al. 2016a, 7–11.) Nimettyjä entiteettejä automaattisin menetelmin etsittäessä hyvä OCR-laatu olisi kuitenkin ensiarvoisen tärkeä hyvän tunnistustuloksen saavuttamiseksi.

Kaiken kaikkiaan testatut entiteettitunnistimet ja muut kielityökalut saivat varsin hyviä tuloksia erityisesti henkilöiden nimien suhteen, eikä eri menetelmien välillä ollut kovin suuria eroja. Moniosaiset nimet kuitenkin aiheuttivat tunnistusongelmia verrattuna yksiosaisiin nimiin. Tässä historiallisessa sanomalehtiaineistossa tunnistustulokset eivät olleet kovinkaan hyviä muun muassa OCR-virheiden vuoksi, mikä kävi ilmi myös historiallisen ja uudemman sanomalehtiaineiston vertailussa. Nykyiset entiteettitunnistimet eivät kuitenkaan suoriutuneet täydellisesti uudemmastakaan sanomalehtiaineistosta, mikä osoittaa sen, että työkaluja tulisi kehittää entistä paremmin suomen kieleen sopiviksi. (Kettunen et al. 2016b; Kettunen & Löfberg 2017.) Nämä tutkimukset olivat ensimmäisiä historialliseen suomen kieleen kohdistuvia nimettyjen entiteettien tunnistukseen liittyviä tutkimuksia, joissa huomio kiinnitettiin ensisijaisesti henkilöiden nimiin ja paikkoihin, ja lisätutkimus ja menetelmien kehittely olisi jatkossa tarpeen.

Vaikka päiväys on sanomalehtiartikkelienkin kannalta merkittävä tieto ajalliseen kontekstiin sitomisen suhteen, ei Kettusen ja muiden tutkimuksessa huomioitu päiväysten tunnistautumista käytännössä millään tavalla, vaan heidän huomionsa oli lähes pelkästään ihmisten ja organisaatioiden sekä paikkojen nimissä, mitkä toki ovat useimmin sanomalehtikokoelmasta etsittäviä asioita. Testatuista tunnistimista ainakin FiNER kuitenkin tunnistaa myös päiväyksiä, joille oli Kettusen ja muiden tutkimuksessa laskettu myös muun muassa tarkkuus- ja saantiarvot, mutta ilmeisesti vertailu muiden tunnistimien kanssa ei ollut mahdollista, joten tutkimustuloksissa päiväysten tunnistumista ei juurikaan raportoitu. Henkilöiden nimillä on lehtiartikkeleissa yleistettävämpi merkitys kuin kirjeissä, joissa henkilöt kuuluvat pääsääntöisesti varsin pieneen lähipiiriin. Vaikka sanomalehtiartikkeli ja kirje ovatkin tekstityypiltään hyvin erilaisia, yhdistää niitä kuitenkin se,

että molemmista on mahdollista etsiä niin päiväyksiä kuin ihmisten ja paikkojen nimiäkin, joiden perusteella on mahdollista luoda laajempaa verkostoa eri entiteettien välille.

3.3 SA-kuvakokoelman louhintaa

Historiatieteissä tutkijat ovat perinteisesti hyödyntäneet ensisijaisesti lähdeaineiston manuaalista läpikäyntiä eli niin sanottua lähiluentaa, ja esimerkiksi suuret valokuvakokoelmat ovat suorastaan vaatineet tämänkaltaista tutkimusotetta, sillä kuva sisältää visuaalista tietoa, jota ei voi kaikilta osin muuttaa tekstimuotoon. Vaikka nykyiset kuvatietohaun menetelmät ovat jo varsin kehittyneitä, on niissä vielä runsaasti parantamisen varaa, sillä haun kohdistaminen pelkästään itse kuvaan on vielä varsin haasteellista. Valokuviin liittyvään metadataan eli kuviin liitettyihin lisätietoihin, kuten kuvatekstiin, on kuitenkin mahdollista päästä käsiksi perinteisempien tiedonhaun keinojen avulla. Haasteena on kuitenkin edelleen kuvien suuri lukumäärä, vaikka digitointi onkin mahdollistanut aiemmin hankalasti hyödynnettävien aineistojen paremman saatavuuden ja käytettävyyden (Elo & Kleemola 2016, 153). Digitaalisessa muodossa olevien aineistojen osalta perinteisten keinojen rinnalle tarvitaankin monenlaisia tietokoneavusteisia tutkimusmenetelmiä, joista yksi on niin sanottu louhinta, jonka avulla aineistosta on mahdollista tuottaa muun muassa visuaalista tutkimustietoa erilaisten verkostokaavioiden avulla, ja tämä menetelmä sopiikin varsin hyvin valokuvatutkimuksen tueksi (Elo & Kleemola 2016, 153–154).

Kimmo Elo ja Olli Kleemola (2016) tarkastelivat Suomen armeijan toisen maailmansodan aikaista digitoitua ja avoimena kuva-arkistona olemassa olevaa sotavalokuvien kokoelmaa eli SA-kuvakokoelmaa, joka sisältää suomalaisten sotakuvaajien eli TK-kuvaajien (tiedotuskomppaniakuvaajien) ottamia valokuvia. Kaiken kaikkiaan SA-kuva-arkisto sisältää yli 100 000 digitoitua valokuvaa. SA-kuva-arkistossa digitoidun kuvan yhteyteen on lisätty valokuvaan liittyviä lisätietoja, kuten kuvauspäivä, kuvaajan nimi sekä usein kuvaajan itsensä laatima kuvateksti, mutta näiden metatietojen hyödyntäminen suoraan ei ainakaan vielä tutkimushetkellä onnistunut, vaan ne täytyi poimia Suomen kirjastojen, arkistojen ja museoiden yhteisen Finna-hakuportaalin kautta. Tässä apuna käytettiin muun muassa Python-ohjelmointikielistä ohjelmaa. Tutkimuksen kiinnostuksena oli saada poimittua joukosta valokuvat, joiden kuvatekstissä oli termi ”saksa” tai jokin sen johdannainen, ja lopputuloksena muodostui 2650 valokuvaa sisältävä tutkimusaineisto. Aivan kattava tämä aineisto ei kuitenkaan ollut, sillä kaikkien valokuvien kuvatekstit

eivät olleet kovinkaan tarkat, minkä vuoksi osa halutuista valokuvista jäi todennäköisesti löytymättä. Toisaalta hakutermeillä joukkoon valikoitui myös sellaisia valokuvia, jotka eivät suorastaan liittyneet suomalaisten ja saksalaisten yhteyksiin sota-aikana, josta tutkimuksessa oltiin kiinnostuneita, vaan kuvassa saattoi olla esimerkiksi saksalainen ase. (Elo & Kleemola 2016.)

Kuvateksteihin oli mahdollista hyödyntää tekstinlouhintaa, jonka avulla aineistosta pyritään tietokoneavusteisesti löytämään ja erottelemaan kiinnostavia rakenteita ja systemaattisia piirteitä. Tutkimusaineistosta oli tämän jälkeen mahdollista luoda visuaalisia karttoja ja verkostoanalyysiä siitä, millaisia muita termejä haettuun sanaan liittyy, kuinka yleisiä ne ovat ja millä tavalla kaikki termit ovat suhteessa toisiinsa. Tämä kaikki on osaltaan niin sanottua visuaalista historiaa, joka on 1990-luvun loppupuolella muodostunut tutkimusalue, jonka puitteissa valokuvia voi tarkastella itsenäisinä tiedon ja merkitysten tuottajina eli tekstuaalisten lähteiden kanssa samanarvoisina historiallisina dokumentteina. (Elo & Kleemola 2016, 161–163.)

Tutkimustuloksia oli mahdollista havainnollistaa visuaalisin keinoin muun muassa sanapilven avulla, jossa termin yleisyyttä tutkimusaineistossa kuvataan erilaisen fonttikoon avulla, jolloin tulokset havainnollistuvat jopa vain yhdellä silmäyksellä. Lisäksi termien esiintyvyyttä erilaisissa konteksteissa analysoitiin yhteisesiintymien eli kollokaatioiden verkostolla, jossa tuloksia havainnollistetaan paitsi sanojen koon, myös niiden välisten viivojen avulla. Eri sotavuosien perusteella tulokseksi saatiin erilaisia termejä eri suhteessa toisiinsa sisältäviä visuaalisia kollokaatioverkostoja, joiden perusteella oli mahdollista luoda valokuvienkin avulla yleiskuva sodan kulloisestakin vaiheista. Tuloksia arvioitiin modulaarisuusarvon avulla, joka kuvaa yhteisörakenteen vahvuutta. Tämän menetelmän valossa tulokset saivat varsin korkeita lukemia, minkä pohjalta pääteltiin, että tietokoneavusteinen luokittelu tarjoaa perinteisemmälle sisällönanalyysille vaihtoehdon. (Elo & Kleemola 2016, 165–185.)

SA-kuvakokoelman tutkimus osoitti, että verkostoanalyysiin perustuva sisällönanalyysi on hyvä tutkimusmenetelmä erityisesti silloin, kun aineisto on rakenteeltaan tai kooltaan niin suuri, että sen hallitseminen manuaalisen luokittelun tai lähiluennan keinoin on haastavaa. Koska on oletettavaa, että tulevaisuudessa historiantutkimuksen kannalta kiinnostavien digitaalisessa muodossa olevien aineistojen lukumäärä jatkuvasti vain kasvaa, ei tarve aineistojen tehokkaassa hallinnassa avustaville menetelmille tai uudenlaisille

tutkimus- ja tietojenlouhintamenetelmille ainakaan vähene. Toisaalta erityisesti historiallisten aineistojen suhteen on olemassa niin sanottu puuttuvan datan ongelma eli aineisto saattaa olla puutteellista, pirstaloitunutta tai syystä tai toisesta vinoutunutta, mikä korostaa tutkijan tietämyksen merkitystä myös digitaalisia apuvälineitä hyödynnettäessä. (Elo & Kleemola 2016, 185–187.)

Pelkkien päivämäärien suhteen vastaavanlainen tietokoneavusteinen louhinta ja sen pohjalta muodostettu visuaalinen karttakuva olisi mahdollista muodostaa lähinnä sillä ajatuksella, milloin kirjeitä kirjoitettiin erityisen paljon, kun samaan aikaan näin jälkikäteen tiedetään sodan kulloinenkin vaihe. Toisaalta koska edes kaikki saman henkilön kirjeenvaihtoon liittyvät kirjeet eivät välttämättä ole syystä tai toisesta säilyneet, ei mitään kovin suuria päätelmiä tästäkään ole mahdollista tehdä. Muissa yhteyksissä, kuten kirjeissä mainittujen henkilöiden tai paikkojen suhteen edellä mainitun kaltainen tekstinlouhintaan perustuva kollokaatioverkosto voisi olla mielenkiintoinen sisällönanalyysin tutkimusmenetelmä. Tekstinlouhintaa voi kuitenkin hyödyntää myös perinteisemmän tiedonhaun keinoin, kun tutkimusaineistosta pyritään löytämään haluttua tietoa.

3.4 Sota-ajan kirjeet muiden alojen tutkimuksessa

Aiempaa tutkimusta Suomen talvi- ja jatkosodan ajan kirjekokoelmaan liittyen on tehty viime vuosina lähinnä historiantutkimuksen ja kielitieteiden parissa, ja käytössä ollut lähdeaineisto on koostunut alkuperäisistä paperimuotoisista kirjeistä digitoitujen versioiden sijasta. Koska tutkimuksen lähtökohdat ja tutkimusongelmat poikkeavat suuresti tämän tutkimuksen tiedonhakuun ja tietokoneavusteisiin menetelmiin keskittyvästä näkökulmasta, esitellään joitakin kirjeisiin kohdistettuja tutkimuksia yleisemmällä tasolla.

Kiinnostus sota-ajan kirjeisiin heräsi voimakkaana vuonna 1975, jolloin Tampereen yliopiston historian ja kansanperinteen laitokset sekä Tampereen Historiallinen Seura järjestivät laajamittaisen kirjeiden keruun, jonka ansiosta sota-aikaista yksityishenkilöiden kirjeenvaihtoa saatiin kerättyä runsaasti talteen arkistointia ja myöhempää tutkimusta varten. Vaikka kirjeiden merkitys onkin suurin historian- ja kansanperinteen tutkijoille, nähtiin jo alkuvaiheessa aineiston käyttömahdollisuudet myös monitieteisiä tutkimushankkeita ajatellen. Lyhyessä ajassa nykyiseen Kansanperinteen arkistoon saatiin koottua kymmeniätuhansia kirjeitä sisältävä kokoelma. (Suojanen 1984, 5; 55–56.) Heti tuoreeltaan kirjeet herättivät mielenkiintoa muun muassa historian-, kirjallisuuden- ja

perinteentutkijoiden keskuudessa, ja tuohon aikaan suuri osa kirjeiden kirjoittajista tai heidän läheisistään oli vielä elossa, mikä tarjosi tutkijoille monipuolisia mahdollisuuksia. Sensuurin vaikutus on luonnollisesti aina pidetty mielessä, mutta sota-aikana monet sellaisetkin ihmiset kirjoittivat toisilleen kirjeitä, jotka eivät välttämättä muuten olisi juuri sellaisia kirjoittaneet, mikä tekee aineistosta monipuolisen ja mielenkiintoisen, vaikka miljoonista sotavuosien aikana lähetetyistä kirjeistä ei arkistoon olekaan päätynyt kuin murto-osa. (Rasila 1984, 23–26.)

Moneksi vuosikymmeneksi tutkimuksellinen kiinnostus sota-ajan kirjeisiin kuitenkin hiipui, kunnes se taas 2000-luvun aikana heräsi uudelleen, kun sotahistorian tutkimus siirtyi kohti yksilöllisempää sotakokemusta sekä kulttuuri- ja sosiaalihistoriallisia kysymyksiä, mikä liittyy laajemminkin historiantutkimuksen trendien muutokseen. Kirjeet ovat herättäneet erityisesti kulttuurihistorioitsijoiden ja etnologien kiinnostuksen, ja tutkimuskysymykset ovat liittyneet muun muassa tapahtumahistoriaan, ihmisten sukulaisuussuhteisiin ja sosiaalisiin verkostoihin, mentaliteetteihin sekä identiteetin rakentumiseen. Erilaisten tieteellisten artikkelikokoelmien lisäksi Suomen sota-ajan kirjeitä on hyödynnetty aineistona myös ainakin kahdessa väitöskirjassa (Sonja Hagelstam ja Erkka Pehkonen, molemmat vuodelta 2014). Tavallisten ihmisten kirjeitä on alettu tutkia laajemmin myös kansainvälisesti, ja toisaalta Suomessakin myös muiden sotien kuin talvi- ja jatkosodan aikaiset kirjeet ovat herättäneet tutkimuksellista kiinnostusta. (Tikka et al. 2015, 7–8.)

Koska sota-aikana kirjeet toimivat merkittävänä yhteydenpitovälineenä, voidaan puhua jopa siitä, että koko Suomen kansa on kirjoittanut. Kirjeet eivät olleet vain taitavien ja kouluja käyneiden kommunikointiväline, vaan kuka tahansa saattoi saada äänensä kuuluville omista kokemuksistaan ja arjestaan kertoessaan, ja tämä näkyy myös arkistomateriaalissa. Kirjeet on myös kirjoitettu muutaman vuoden aikana ja hyvin samankaltaisessa elämäntilanteessa muihin kirjoittajiin verrattuna, sillä sota kosketti kaikkia tavalla tai toisella. Tämän vuoksi kirjekokoelmaa voikin pitää monin tavoin hyvin erityislaatuisena. (Taskinen 2015.)

Sota-ajan kirjeiden kielitieteellisempi kiinnostus kasvoi vuonna 2016 Tampereen yliopiston suomen kielen erikoiskurssin kautta, jonka seurauksena opiskelijat tekivät pienimuotoisia tutkimuksia kirjeaineiston avulla. Näkökulmat liittyivät ennen kaikkea erilaisiin kielellisiin piirteisiin, joiden perusteella kirjeistä pystyy tulkitsemaan muun muassa tunnetiloja, ajatuksia ja kokemuksia. Myös erilaiset suomen kielen murteet ovat

kirjeteksteissä näkyvissä, mikä on herättänyt tutkimuksellista kiinnostusta kielitieteiden parissa. (Mustanoja 2017a, 4–5.)

Monitieteisempää tutkimusotetta ilmentää vuonna 2017 alkanut ja tätä tutkielmaa kirjoitettaessa edelleen meneillään oleva Suomen Kulttuurirahaston rahoittama STASKO-hanke eli ”Suuret tietokanta-aineistot sodan kokemushistoriassa”, jonka tutkimusaineisto koostuu TK-kuvaajien ottamista valokuvista metatietoineen (SA-kuvakokoelma), sota-ajan kirjekokoelmasta, sota-aikana menehtyneiden tietokannasta sekä Kansallisarkiston sota-ajan kantakorttikokoelmasta. Näitä suuria aineistomääriä kootaan, tutkitaan ja yhdistellään kokonaiskuvan saamiseksi, ja hankkeeseen liittyy vahva pedagoginen ote. Hankkeen taustalla on historiallisten aineistojen digitoinnin aikaansaama muutos, kun perinteisten lähdeaineistojen rinnalle on tietotekniikan kehittymisen myötä noussut laajoja sähköisiä aineistoja, joita on mahdollista käsitellä aiempaa monipuolisemmin ja tehokkaammin. Tutkimustulosten esittelyssä on myös tarkoitus hyödyntää nykyaikaisia datavisualisointeja. (STASKO-hankkeen verkkosivut.) Koska tutkimus on edelleen kesken, ei sen tuloksia pääse toistaiseksi kuitenkaan vielä tarkastelemaan.

Kaikille edellä mainittujen alojen tutkimukselle on yhteistä se, että tutkimuksellinen kiinnostus on ennen kaikkea kirjeen sisällössä eli varsinaisessa kirjetekstissä. Tällöin alkuperäinen paperille usein käsin kirjoitettu aineisto tarjoaa hyvin henkilökohtaisen kosketuspinnan menneeseen aikaan. Kirjeen sisällön lukemisen suhteen ei ole niinkään väliä sillä, onko aineisto paperisessa vai sähköisessä muodossa, sillä siinä missä käsiala saattaa olla helppo- tai vaikeaselkoinen, myös optisesti tunnistettu digitaalinen teksti saattaa olla selkeää tai haastavampaa lukea riippuen tunnistustuloksen laadusta. Informaatiotutkimuksella ja edelleen tiedonhaun tutkimuksella on kuitenkin mahdollisuuksia kiinnittää huomiota myös kirjeiden ulkoisempiin tekijöihin, kuten digitaalisessa muodossa olevien aineistojen käsittelyyn, minkä perimmäisenä tarkoituksena on helpottaa halutun tiedon löytymistä tutkimusalasta riippumatta.

4 AINEISTON ESITTELY JA TUTKIMUSMENETELMÄT

Tässä luvussa tarkastellaan tarkemmin tutkimusaineistoa ja tutkimuksessa hyödynnettyjä menetelmiä. Alaluvussa 4.1 esitellään ensin tämän tutkimuksen kannalta tärkeät tutkimuskysymykset. Luvussa 4.2 tarkastellaan tutkimuksessa käytetyn aineiston taustoja hieman tarkemmin ja kerrotaan, kuinka testikokoelma koottiin tätä tutkimusta varten. Alaluvussa 4.3 kerrotaan tutkimusaineiston manuaalisesta annotoinnista eli siitä, miten aineisto on esikäsitelty tutkimusta ja hakutulosten evaluointia varten. Luvussa 4.4 tutustutaan tarkemmin kahteen automaattiseen tässä tutkimuksessa vertailtavaan tietotekniseen menetelmään, minkä lisäksi esitellään myös käytetyt hakuaiheet.

4.1 Tutkimuskysymykset

Tässä tutkimuksessa pyrittiin löytämään vastaukset seuraaviin kysymyksiin:

1. Tunnistaako FiNER kaikkien kirjeiden päiväyksen entiteetiksi? Miksi osa jää mahdollisesti tunnistumatta, ja merkitäänkö entiteetiksi myös muita ajallisia ilmauksia kuin kirjeen varsinainen päiväys?
2. Miten automaattiset menetelmät vertautuvat ihmisen tekemään tulkintaan päiväysten löytymisessä?
3. Miksi osa päiväyksistä jää mahdollisesti löytymättä automaattisin keinoin?

4.2 Sota-ajan kirjeet

Tässä tutkielmassa tarkasteltiin Suomen talvi- ja jatkosodan aikaisten kirjeiden digitoiduista ja tekstitiedostoiksi muutetuista versioista poimittua otantaa. Kirjekokoelman keruuta ja käyttöyhteyksiä tarkasteltiin lyhyesti jo edellä luvussa 3.4, mutta alaluvussa 4.2.1 tarkastellaan vielä hieman tarkemmin alkuperäisen sota-ajan kirjekokoelman syntyä, ja alaluvussa 4.2.2 puolestaan digitoidun kirjekokoelman muodostumista. Luvussa 4.2.3 esitellään tarkemmin tätä tutkimusta varten koottu otanta digitoiduista kirjeistä.

4.2.1 Kansanperinteen arkiston sota-ajan kirjekokoelma

Tampereen yliopiston Kansanperinteen arkistossa on Sota-ajan kirjekokoelmaksi (SAK) nimetty kokoelma, joka sisältää yli 40 000 yksittäistä kirjettä, postikorttia tai muuta

viestiä, jotka on kirjoitettu pääasiassa talvi- ja jatkosodan sekä välirauhan aikana eli vuosina 1939–1945. Jonkin verran kokoelma sisältää myös muina vuosina kirjoitettuja kirjeitä, mutta niitä on varsin vähän. Kirjeet ovat yksityishenkilöiden välistä kirjeenvaihtoa rintamalta toiselle sekä kotiin, ja niiden kerääminen arkistoon aloitettiin yleisen kutsuilmoituksen avulla vuonna 1975. Tällöin Tampereen yliopiston historian ja kansanperinteen laitokset sekä Tampereen Historiallinen Seura järjestivät siis laajamittaisen kirjeiden keruun, jonka ansiosta sota-aikaista yksityishenkilöiden kirjeenvaihtoa saatiin lyhyessä ajassa kerättyä runsaasti talteen arkistointia ja myöhempää tutkimusta varten. Keruun pääasiallisena innoittajana oli halu pelastaa jälkipolvia varten ”historiantutkimukselle ja muullekin kulttuuritoiminnalle korvaamattomia tietolähteitä” siitä, ”mitä tavallinen ihminen noina aikoina tunsi ja koki”. (Suojanen 1984, 55–56.)

Koska sota-ajasta oli kulunut kirjeiden keruuta aloitettaessakin jo muutama vuosikymmen, oli huoli siitä, että kirjeitä häviää tai hävitetään lisää, kun ajallinen etäisyys sotaan edelleenkin kasvaa (Rasila 1984, 23–24). Tämän vuoksi kirjeitä haluttiin saada talteen arkistoon mahdollisimman paljon, ja laajamittainen keruu katsottiin tarpeelliseksi. Ensimmäisen kymmenen vuoden aikana kirjeitä vastaanotettiin Kansanperinteen arkistoon parikymmentätuhatta, ja koska niitä vastaanotetaan edelleenkin, on sota-ajan kirjekokoelmaan karttunut sittemmin jo reilusti yli 40 000 yksittäistä kirjettä, korttia tai muuta viestilappusta. Kirjeet on arkistoitu niiden lahjoittajan kotipaikan perusteella, ja kokoelmanumerointi on muodostunut kirjeiden saapumisjärjestyksen mukaan eli mitä pienempi SAK-kokoelmanumero tietyn henkilön kirjeenvaihdolla on, sitä aikaisemmin se on Kansanperinteen arkistoon vastaanotettu. (Mustanoja 2017b, 9–10; 18.)

Sota-ajan kirjekokoelma sisältää pääasiassa käsinkirjoitettuja kirjeitä, mutta 1980-luvulla näistä kirjeistä reilu 7000 kappaletta on kirjoitettu konekirjoituslaitteella puhtaaksi. Nämä kirjeiden puhtaaksikirjoitetut versiot sijaitsevat Kansanperinteen arkistossa printteinä.

4.2.2 STASKO-hankkeen digitoimat kirjeet

”Suuret tietokanta-aineistot sodan kokemushistoriassa” (STASKO) on monitieteinen tämän tutkielman kirjoitushetkellä edelleen käynnissä oleva Tampereen yliopiston tutkimushanke, jossa tarkastellaan Suomen talvi-, jatko- ja Lapin sodan kokemushistoriaa suurten aineistojen eli niin sanotun big datan valossa (STASKO-hankkeen verkkosivut). Yhden hankkeen suurimmista aineistoista muodostaa Kansanperinteen arkiston sota-ajan kirjekokoelma. STASKO-hankkeen puitteissa on digitoitu 7764 yksittäistä

puhtaaksikirjoitettua kirjettä, jotka kuuluvat 121 osakokoelmaan, eli digitoitu kokoelma käsittää 121 eri henkilön kirjeenvaihdon. Digitointi on tapahtunut siten, että konekirjoitetut printit on ensin skannattu pdf-muotoon, minkä jälkeen ne on ajettu OCR:n eli optisen tunnistuksen läpi. Lopuksi OCR-käsitellyt pdf-tiedostot on muutettu myös tekstitiedostoiksi. Tämän tutkimuksen aineisto koostuu näiden digitoitujen kirjeiden tekstitiedostoista, joista on poimittu tutkimusta varten pieni testikokoelma. Alkuperäisiä paperisia kirjeitä tämän tutkimuksen käytössä ei siis ole ollut.

Toistaiseksi STASKO-hankkeen digitoimat kirjeet eivät sijaitse missään arkistossa, vaan ne ovat vielä ainoastaan tutkimuskäytössä.

4.2.3 Tutkimusta varten luotu testikokoelma

Tämän tutkimuksen aineistona käytettiin 992 yksittäistä kirjettä sisältävää otantaa talvi- ja jatkosodan aikaisten kirjeiden digitoiduista versioista. Aineisto koottiin Tampereen yliopiston silloisessa viestintätieteiden tiedekunnassa suoritetulla työharjoittelujaksolla tutkimusapulaisen tehtävässä syksyllä 2018 liittyen Suomen Akatemian tutkimushankkeeseen ”Muuttuvien työtehtävien tukeminen digitaalisissa ympäristöissä sumeiden merkkijonomenetelmien avulla”. Tässä tutkimushankkeessa aineistona hyödynnettiin STASKO-hankkeen digitoimaa 7764 kirjettä sisältävää kokoelmaa, josta noin seitsemäsosa valittiin tämän tutkielman tutkimusaineistoksi. Koko digitoitu aineisto on liian suuri, jotta se olisi tämän tutkimuksen puitteissa mahdollista käydä manuaalisesti läpi, ja lukumäärällisesti valittu otos on oletettavasti riittävä valittujen tutkimusmenetelmien testaamista ja tulosten analysointia varten. Tässä tutkimuksessa on käytetty ainoastaan digitoituja tekstitiedostoksi muutettuja kirjeitä eli skannattuja digitoituja versioita tai alkuperäisiä paperisia kirjeitä ei ole edes nähty.

Tämän tutkimuksen käytössä olleet kirjeet kuuluvat viiteen osakokoelmaan eli ne sisältävät viiden eri henkilön kirjeenvaihtoa. Koska koko digitoitu kirjeaineisto kattaa yli sata osakokoelmaa, joiden sisältämien yksittäisten kirjeiden lukumäärä vaihtelee muutamasta kappaleesta useaan sataan, tutkimusaineistoon pyrittiin valitsemaan sellaiset osakokoelmat, joiden kirjemäärä on kohtalainen. Valintaa tehdessä ulkopuolelle jätettiin ne osakokoelmat, joissa kirjeitä on alle sata, ja toisaalta ulkopuolelle jäivät manuaalisen annotoinnin työläyden vuoksi myös hyvin suuret osakokoelmat yhtä kokoelmaa lukuun ottamatta.

Kirjekokoelmat nimettiin tutkimuskäyttöä varten anonyymisti muotoon ”kokoelma 1”, ”kokoelma 2”, ”kokoelma 3”, ”kokoelma 4” ja ”kokoelma 5”. Kirjeiden varsinaiseen tekstisisältöön ei tässä tutkielmassa keskitytty päiväyksiä lukuun ottamatta. Kaksi tutkimusaineistoon ensin valittua kirjekokoelmaa (kokoelma 1 ja kokoelma 2) olivat ne, joita käsiteltiin eniten myös harjoittelujakson aikana, minkä vuoksi kokoelma 2 valikoitui mukaan laajuudestaan huolimatta. Niiden lisäksi tutkimusaineistoon valittiin kolme muuta kirjekokoelmaa sen perusteella, että tarvittava yksittäisten kirjeiden yhteislukumäärä (noin tuhat) täyttyisi.

Tutkimusaineistoon valikoituivat seuraavat viisi osakokoelmaa, ja niiden sisältämä yksittäisten eroteltujen kirjeiden lukumäärä on seuraava: kokoelma 1 (SAK/41) 143 kirjettä, kokoelma 2 (SAK/54) 415 kirjettä, kokoelma 3 (SAK/27) 160 kirjettä, kokoelma 4 (SAK/64) 154 kirjettä sekä kokoelma 5 (SAK/89) 120 kirjettä. Yhteensä yksittäisiä kirjeitä on siis tutkimusaineistoa varten muodostetussa testikokoelmassa 992 kappaletta. Kirjeiden lukumäärät perustuvat omaan laskennalliseen tulkintaan kirjeiden välisistä rajoista. Taulukossa 1 havainnollistetaan tarkemmin, montako kirjettä vuosittain jaoteltuna kukin osakokoelma sisältää.

Taulukko 1. Kirjeiden lukumäärät eri vuosina osakokoelmittain (kpl)

vuosi	**1939**	**1940**	**1941**	**1942**	**1943**	**1944**	**muu**	**ei tiedossa**	**yht.**
kokoelma 1	21	34	62	0	0	21	0	5	143
kokoelma 2	0	29	69	199	107	6	1	4	415
kokoelma 3	0	0	0	0	66	88	1	5	160
kokoelma 4	33	93	22	0	0	0	1	5	154
kokoelma 5	3	38	46	24	2	0	1	6	120
yht.	57	194	199	223	175	115	4	25	992

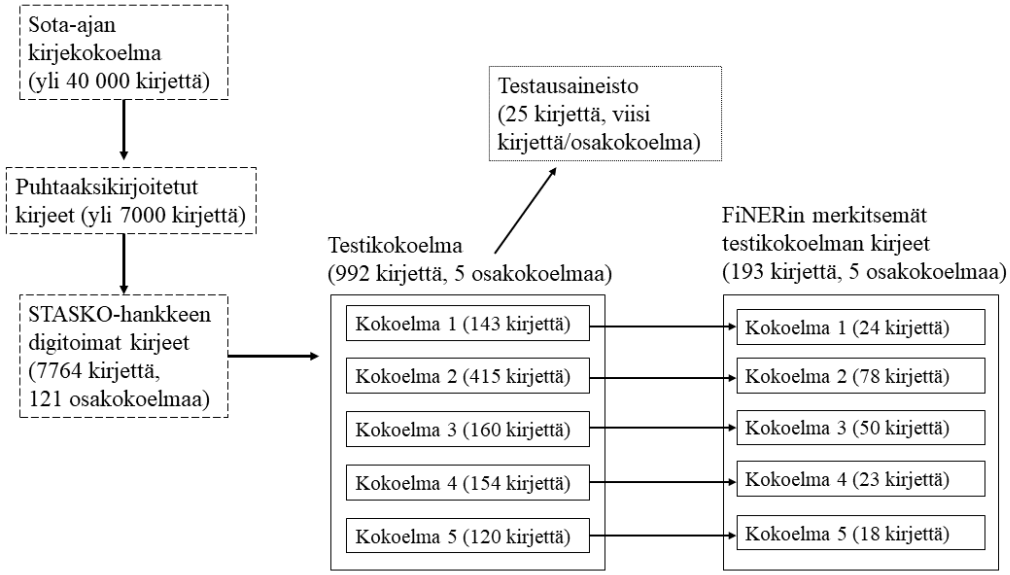
Kuten taulukosta 1 havaitaan, mikään kirjekokoelma ei sisällä kirjeitä kaikilta sota-ajan vuosilta. Vuosiluku ”muu” käsittää saatekirjeen 1970-luvulta, jolloin sodanaikaisia

kirjeitä lähetettiin arkistoitavaksi Kansanperinteen arkistoon. Nämä saatekirjeet ovat myös osa kokoelmaa, mutta niitä ei tässä tutkimuksessa käytännössä huomioida, sillä ajallinen kiinnostus on sotavuosissa. Jokaisessa osakokoelmassa on muutama kirje, joissa vuosilukua tai koko päiväystä ei ollut mainittu lainkaan, eikä sitä näin ollen voi tietää. Yhteensä näitä kirjeitä on 25 kappaletta (2,5 % testikokoelman kirjeistä), kuten taulukosta 1 ilmenee.

Kaikki kirjeet ovat suomenkielisiä, ja teksti sisältää vaihtelevasti myös OCR-virheitä eli väärin tunnistuneita sanoja tai merkkejä. Kirjeet sijaitsevat viidessä erillisessä tekstitiedostossa osakokoelmittain. Koska alkuperäiset paperiset kirjeetkin sijaitsevat arkistossa yhdessä osakokoelmittain, ei niiden erottelua yksittäistiedostoiksi digitaalisessa muodossakaan katsottu tutkimuksessa tästä syystä perustelluksi. STASKO-hanke on eritellyt kirjeet myös osakokoelmittain yksittäistiedostoiksi, jotka on nimetty mahdollisuuksien mukaan kirjeen päiväyksen perusteella, mutta koska en ole itse tätä tulkintaa tehnyt ja havaitsin jossain määrin päiväyksissä myös virheellisyyksiä tai puutteita, päätin koota tätä tutkimusta varten päiväyksistä oman tekstin lukemiseen perustuvan tulkintani. Varsinaisen tekstisisällön suhteen kirjeiden välisten rajojen erottelussa nojauduttiin kuitenkin pääasiassa hankkeen tulkintaan, sillä heidän käytössään on ollut paitsi alkuperäiset paperiset kirjeet, myös alkuperäiset digitoidut versiotkin.

Tutkimusta varten luotiin siis testikokoelma, joka sisältää viiden henkilön kirjeenvaihtoa. Kolme osakokoelmaa koostuu pääasiassa aviopuolisoiden välisestä kirjeenvaihdosta, joskin myös muiden sukulaisten tai tuttavien kirjeitä on mukana. Kaksi osakokoelmaa sisältää pääasiassa vanhempien ja heidän aikuisten lastensa välistä kirjeenvaihtoa. Neljässä osakokoelmassa on useamman henkilön lähettämiä kirjeitä, mutta yksi kokoelma koostuu vain yhden henkilön lähettämistä kirjeistä. Koko tutkimusaineisto sisältää useamman eri henkilön kirjoittamia kirjeitä ja täten päiväysten merkintätavoissakin on jonkin verran yksilöllistä vaihtelua, mutta siitä huolimatta voitaneen tarkastelun pohjalta tehdä yleistettäviä johtopäätöksiä päiväysten löytyvyydestä koskien koko digitoitua kirjeaineistoa, sillä päiväysten erilaisia merkintätapoja on lopultakin varsin rajallinen määrä.

Kaaviosta 1 näkee tarkemmin testikokoelman muodostumisen vaihe vaiheelta alkuperäisestä paperisesta ja sittemmin digitoidusta kirjeaineistosta.



Kaavio 1. Aineiston muodostuminen ja kirjeiden lukumäärät eri vaiheissa

Kaaviossa 1 kuvatut katkoviivalaatikossa olevat kirjeaineistot ovat niitä, joita tässä tutkimuksessa ei ole sellaisenaan käytetty, vaan tutkimusta varten luotiin edellä kuvatun kaltainen testikokoelma. Testikokoelman pohjalta muodostettu pieni testausaineisto liittyi automaattisten menetelmien testaukseen ja kehittelyyn, ja siitä kerrotaan tarkemmin luvussa 4.4. Entiteettitunnistin FiNERin käsittelyn jälkeen testikokoelmasta muodostui päiväysentiteettitunnisteen saaneiden kirjeiden pohjalta pienempi kirjemäärä. FiNERin ominaispiirteistä kerrotaan tarkemmin luvussa 4.4.2 sekä tuloksista luvussa 5.1, jossa kaaviossa mainitut luvut selitetään tarkemmin. Jokaisesta aineiston ja testikokoelman osavaiheesta on ilmoitettu kirjeiden tarkka lukumäärä.

4.3 Aineiston manuaalinen annotointi

Jotta aineistoa olisi mahdollista tarkastella selkeästi ja käsitellä myös automaattisin tietoteknisin menetelmin, kirjetiedostot oli tarpeen esikäsitellä annotoimalla niille rakennetta ja jonkin verran metatietoja. Kukin viisi osakokoelmaa käytiin silmäillen ja päiväystietojen osalta tarkemmin lueskellen läpi.

Havaittiin, että kirjeteksti sisälsi jonkin verran kulmasulkeita eli merkkejä ”<” ja ”>”, jotka ovat tulleet tekstiin OCR-käsittelyn myötä niin sanottuina OCR-virheinä. Koska

annotoinnin aikana tekstille oli tarkoitus luoda metatietoja kulmasulkeita sisältävien tagien avulla, oli OCR-virheet tarpeen korjata tekstistä ensin pois, jotta automaattinen käsittely tulisi ylipäätään mahdolliseksi. Tekstistä muokattiin pois kolmenlaisia kulmasuljetapauksia: 1) ”l<”-merkit korvattiin isolla K-kirjaimella, sillä tekstiä luettaessa kyse oli tavallisesti virheellisesti tunnistuneesta K-kirjaimesta. 2) Pelkkä ”<”-merkki korvattiin tilanteesta riippuen pienellä k-kirjaimella tai poistettiin kokonaan. 3) Kaikki ”>”-merkit poistettiin tekstistä kokonaan, sillä niillä ei vaikuttanut olevan mitään selkeää sisällöllistä merkitystä. Kirjeiden varsinaiseen tekstisisältöön ei ole koskettu muutoin eli esimerkiksi kaikki muunlaiset OCR-virheet on jätetty tekstiin.

Osakokoelma kerrallaan yksittäiset kirjeet eroteltiin toisistaan rakenteen luomiseksi ja kirjeiden tarkastelua selkeyttämään. Kunkin kirjeen alkuun merkittiin omalle rivilleen ”<letter>”-tagi, ja vastaavasti kirjeen loppuun merkittiin sulkeva ”</letter>”-tagi omalle rivilleen. Kirjeiden väliset rajat oli varsin helppo tunnistaa, sillä uusi kirje alkoi yleensä sanalla ”Kirje” tai ”Postikortti”, joka on lisätty ilmeisesti puhtaaksikirjoitus- ja arkistointivaiheessa. Toisinaan kirjeen alussa saattoi olla myös muita sanoja tai merkkejä, jolloin oli tarpeen tukeutua STASKO-hankkeen tulkintaan, mutta pääsääntöisesti kirjeiden erottelu oli varsin yksiselitteistä.

Jokaiselle kirjeelle luotiin keinotekoinen, yksilöivä id-numerointi alkaen numerosta 1, ja numerointi kulkee tiedoston alusta loppua kohden järjestyksessä kirje kerrallaan. Id-numerointi merkittiin omalle rivilleen kirjeen alkuun alkutagin ”<id>” sekä lopputagin ”</id>” avulla. Jokaisessa osakokoelmassa numerointi alkaa alusta, sillä kokoelmia käsiteltiin omina kokonaisuuksinaan. Yksilöivän id-numeroinnin tarkoitus oli helpottaa tietojen hakemista ja hakutulosten listaamista, sillä sen avulla kirjeen sijainti osakokoelmassa olisi helpompi myöhemmin paikallistaa, ja myös tulosten arviointi olisi selkeämpää.

Tämän tutkimuksen kannalta merkittävin kirjeeseen liittyvä tieto on kirjeen kirjoitushetken päiväys, joka tämän vuoksi haluttiin tulkita ja poimia tekstistä erilleen metatiedoksi ISO-standardia mukailevaan muotoon vvvvkkpp, jossa ”v” tarkoittaa vuotta, ”k” kuukautta ja ”p” päivää (ks. esim. Nagao 2003, 63). Päiväys merkittiin id-numeron jälkeen omalle rivilleen alku- ja lopputagin väliin. Esimerkiksi ”<date>19400101</date>” tarkoittaa tammikuun ensimmäistä päivää vuonna 1940. Päiväyksen olisi voinut annotoida myös suoraan varsinaiseen kirjetekstiin tagien avulla, mutta koska pyrkimys oli pitää kirjeen varsinainen sisältö mahdollisimman autenttisena, ei tämänkaltaista annotaatiota

haluttu käyttää. Päiväyksen poimiminen metatiedoksi mahdollisti myös epämääräisempien ilmausten tulkinnan tarkaksi päivämääräksi paremmin ja useammissa tapauksissa kuin tekstinsisäisellä annotoinnilla olisi ollut mahdollista.

Pääasiassa päiväykset on kirjeissä merkitty perinteiseen suomalaiseen muotoon, jossa ensin mainitaan päivä, sitten kuukausi ja lopuksi vuosi joko kokonaan tai sen kaksi viimeistä lukua. Numeroiden välisenä erottimena käytettiin tavallisesti pistettä, pilkkua tai kauttaviivaa, ja vuosiluvun kahden viimeisen luvun edessä saattoi olla lyhyt viiva. Joissakin kirjeissä kuukausi saatettiin kirjoittaa kirjaimin joko ennen tai jälkeen päivän. Toisinaan päiväys on saatettu ilmaista tarkkaa päivämäärää kertomatta käyttämällä jonkin juhlapyhän tai nimipäivän ilmausta, kuten ”jouluaatto”, ”Marianpäivä” tai ”äitienpäivä”, jonka perusteella tarkka päivämäärä on kuitenkin tulkittavissa ainakin siinä tapauksessa, että myös vuosiluku on mainittu. Kiinteiden juhlapäivien suhteen tulkinta ei tarvinnut aina edes vuosilukua, mutta liukuvien juhlien suhteen tarkan päiväyksen tulkinta hankaloitui ilman sitä. Annotoidessa päiväys pyrittiin tulkitsemaan tekstistä mahdollisuuksien mukaan tarkaksi metatiedoksi, ja esimerkiksi OCR-virheet pyrittiin myös tulkitsemaan. Päiväyksissä OCR-virheet osuivat pääsääntöisesti kuukauden tai vuosiluvun kohdalle, jolloin ne oli varsin helppo päätellä, sillä virheelliset merkit muistuttivat usein tarkoitettua numeroa, ja toisaalta kuukauden suhteen merkintä saattoi olla myös kirjaimista koostuva roomalainen numero. Mikäli päiväystä ei ollut tarkkaan ilmaistu tai sitä ei virheellisyyksistä huolimatta pystytty tulkitsemaan tarpeeksi selvästi, merkittiin päiväysmetatietoon x-kirjain puuttuvan tai epävarman numeron kohdalle.

Jokaisen kirjeen metatietoihin lisättiin päiväyksen jälkeen kommenttirivi mahdollisten epäselvien tai tulkintaa vaatineiden päiväysten selventämistä varten. Mahdollinen kommentti kirjoitettiin suomeksi alkutagin ”<comment>” ja lopputagin ”</comment>” väliin. Mikäli mitään kommentoitavaa ei ollut, kommenttikenttään kirjoitettiin kuitenkin ”NULL”, joka liittyy tietokantamaailmassa tyhjään kenttään. Jokaiselle kirjeelle annotoitiin siis sama määrä metatietotageja, jolloin tietojen myöhempi käsittely tai lisääminen on helpompaa.

Varsinainen kirjeen tekstisisältö eroteltiin metatiedoista omien tagiensa sisälle, jotta se ei sotkeutuisi muuhun jälkikäteen lisättyyn materiaaliin, vaan pysyisi koskemattomana ja mahdollisimman autenttisena. Alkutagi ”<text>” merkittiin omalle rivilleen, sen jälkeen uudelta riviltä alkoi kirjeteksti ja loppuun omalle rivilleen lisättiin vielä lopputagi

”</text>”. Tekstisisällön lopettavan tagin perään omalle rivilleen merkittiin siis vielä koko kirjeen lopettava ”</letter>”-tagi, jonka jälkeen seuraavalta riviltä alkoi uusi kirje.

Edellä kuvatulla tavalla jokainen viisi osakokoelmaa käytiin alusta loppuun kirje kerrallaan läpi metatietoja lisäten, rakennetta luoden ja päiväyksiä tulkiten. Varsinaisesta kirjeen sisällöstä päiväystä ja paikkakuntaa lukuun ottamatta karsittu esimerkkikatkelma kokoelman 1 ensimmäisestä kirjeestä lisättyine metatietoineen havainnollistaa tehtyä annotointia:

```
<letter>
<id>1</id>
<date>19391106</date>
<comment>NULL</comment>
<text>
Kirje:
...
Kankaanrää  6.11.1939.
...
</text>
</letter>
```

Tässä tutkielmassa itse tekemäni manuaalinen annotointi toimii automaattisten tutkimusmenetelmien tarkastelussa ja vertailussa niin sanottuna gold standardina, sillä kirjeet on tällä tavoin käyty yksitellen tarkasti silmäillen läpi, minkä pohjalta päiväykset on ollut mahdollista löytää ja tulkita mahdollisuuksien mukaan. Manuaalisen annotoinin perusteella on mahdollista tietää, mitkä kirjeet kustakin osakokoelmasta pitäisivät löytyä kulloisessakin tiedonhakutilanteessa. Toisaalta tämän pohjalta on myös mahdollista nähdä, kuinka suuri osa päiväyksistä on hyvin selkeitä, ja moniko niistä tulkinnankin jälkeen jää vielä epäselväksi. Manuaalisesta vertailupohjasta on mahdollista saada selville sekin, miksi jotkin kirjeet eivät todennäköisesti tule löytymään millään menetelmällä eli mistä syistä päiväystä ei voi tietää tai pystyä poimimaan ja siten löytää.

Manuaalisen annotoinnin lopuksi kaikkien kirjekokoelmatiedostojen teksti muutettiin UTF-8-merkistöön, jotta ennen kaikkea ääkkösiin liittyviä merkistöongelmia ei tietoteknisten menetelmien suhteen jatkossa tulisi.

4.4 Automaattiset tutkimusmenetelmät

Digitoitujen kirjeiden tarkastelu automaattisin menetelmin liittyy hyvin pitkälti tekstinlouhintaan, joka puolestaan on pohjimmiltaan tiedonhakua. Toisaalta päiväys on yksi nimetyistä entiteeteistä, joiden tunnistus tekstistä automaattisin entiteettitunnistimin on

mahdollista. Koska sota-ajan kirjeet ovat ennen kaikkea luonnollista kieltä, tulee tietoteknisten menetelmien suhteen pyrkiä hyödyntämään olemassa olevia luonnollisen kielen työkaluja mahdollisimman paljon, ja ohjelmointikielistä Python tarjoaa tähän parhaat mahdollisuudet. Automaattisten menetelmien testausta ja ohjelman kehittelyä varten tutkimusaineistosta poimittiin pieni 25 kirjettä sisältävä testausaineisto, joka sisältää viisi yksittäistä kirjettä jokaisesta viidestä osakokoelmasta. Testausaineisto suunniteltiin ja koottiin siten, että vain osan kirjeistä pitäisi tulla tuloslistaukseen eri hakutilanteissa. Tietoteknisten menetelmien ajaminen suoraan koko tutkimusaineistolle ei tuntunut mielekkäältä, joten siksi pienen testausaineiston koostaminen tuntui testaus- ja suunnitteluvaiheessa parhaimmalta lähestymistavalta. Ohjelmien testaus ja käyttö sekä testaus- ja tutkimusaineiston käsittely tapahtuivat Tampereen yliopiston shell.sis.uta.fi-palvelimella, jonne pääsee kirjautumaan peruspalvelutunnusten avulla.

Alaluvussa 4.4.1 esitellään tässä tutkimuksessa tarkastellut testiluontoiset tiedonhakukyselyt. Luvussa 4.4.2 tarkastellaan suomalaista entiteettitunnistinta nimeltään FiNER, jonka käyttöönotto Kielipankin tarjoaman Finnish Tagtools -työkalupaketin kautta on mahdollista. Alaluvussa 4.4.3 puolestaan käydään pääpiirteissään läpi itse kehitelty Python-ohjelmointikielinen ohjelma, jonka tarkoitus on pystyä poimimaan kaikkien kirjeiden joukosta ne kirjeet, joita kulloinkin halutaan tarkastella.

4.4.1 Testattavat hakukyselyt

Valittu sota-ajan kirjeaineisto toimi siis testikokoelmana, jota tarkasteltiin muutaman ennalta määritellyn testikyselyn avulla. Samat kolme hakua suoritettiin sekä manuaalisesti annotoiduille viidelle kirjeet sisältävälle tekstitiedostolle että viidelle FiNERin avulla käsitellylle vastaavalle tiedostolle. Koska hakujen suorittamista varten ei ollut olemassa mitään valmista sovellettavissa olevaa hakukonetta, suoritettiin haut itse kehitellyn Python-ohjelmointikielisen ohjelman hieman muokattujen versioiden avulla. Sekä niin sanotusti puhdas että FiNERin kautta kulkenut aineisto hyödynsivät siis samankaltaisia pienin muokkauksin toisistaan eroavia ohjelmia, joita on kuvailtu tarkemmin luvussa 4.4.3, ja kunkin ohjelman version koodisisältö löytyy kokonaisuudessaan liitteistä 1–6.

Ensimmäisellä haulla pyrittiin löytämään kaikki kirjeet, jotka on kirjoitettu vuonna 1940. Tämä vuosi valikoitui haun kohteeksi siksi, että viidestä osakokoelmasta neljä sisältää kirjeitä tältä vuodelta. Myös vuodelta 1941 olisi löytynyt kirjeitä neljästä kokoelmasta,

mutta koska 41 on erään osakokoelman alkuperäinen SAK-kokoelmanumero, tämä rajautui pois tarkastelusta todennäköisen hakutuloksiin tulevan liiallisen hälyn vuoksi.

Toista hakua tarkennettiin pyrkimällä löytämään vuoden 1940 maaliskuussa kirjoitetut kirjeet. Talvisota loppui tuolloin, joten oletetusti kirjeitä tuolloin vielä kirjoitettiin, sillä rintamalta ei päässyt heti kotiin.

Kolmannen haun avulla pyrittiin löytämään kirjeitä tarkalla päiväyksellä. Kiinnostuksen kohteeksi valikoituivat jouluna kirjoitetut kirjeet vuodesta riippumatta, sillä oli oletettavaa, että mikäli jouluksi ei rintamalta päässyt kotiin, kirjoitettiin kotiväelle kuitenkin kirje, sillä joulu on perinteisesti perhejuhla. Jouluksi laskettiin sekä jouluaatto että joulupäivä eli 24.12. sekä 25.12., minkä lisäksi etsintää suoritettiin termin ”joulu” avulla siten, että ”joulukuu” kuitenkin rajautuisi pois tuloksista, sillä myös sanalliset päiväykset, kuten ”jouluaattona” tai ”jouluna” haluttiin mukaan tuloksiin.

4.4.2 FiNER

FiNER eli Finnish Named Entity Recognizer on sääntöihin perustuva (rule-based) nimettyjen entiteettien tunnistin. Sen tarjoaa muun muassa tutkijoiden ja opiskelijoiden käyttöön FIN-CLARIN-konsortioon kuuluva Kielipankki. Suomalaisten yliopistojen, CSC:n ja Kotimaisten kielten keskuksen muodostama FIN-CLARIN on osa eurooppalaista CLARIN ERIC -yhteisöä, jonka tarkoitus on luoda digitaalisille ihmis- ja sosiaalitieteille yhteistä tutkimusinfrastruktuuria. Kielipankki puolestaan on eräs tämän konsortion palvelukokonaisuus, joka tarjoaa vapaaseen käyttöön monenlaisia suomenkielisiä kieliaineistoja ja -työkaluja. Kielipankin kautta on mahdollista saada käyttöönsä muun muassa Finnish Tagtools -kielityökalupaketti (FinTag), jonka yksi osa on nimettyjen entiteettien tunnistin FiNER. Pakettiin kuuluu myös monia morfologisia työkaluja, jotka eivät kuitenkaan tämän tutkimuksen kannalta ole olennaisia. (Ks. FIN-CLARIN; Kielipankki, Finnish Tagtools.) Koska tutkimuksen kiinnostus on vain päiväyksissä ja siinä, tunnistaako FiNER ne nimetyiksi entiteeteiksi vai ei, muita kielityökaluja ei tässä hyödynnetä, sillä esimerkiksi sanojen perusmuotoistamisesta ei päiväysten osalta ole juurikaan hyötyä.

FiNER perustuu FinnPos-nimiseen sanaluokkatunnistimeen (POS, Part of Speech) ja sääntöihin perustuvaan nimettyjen entiteettien tunnistimeen, morfologiseen suomen kielen pakettiin Omorfiin sekä FinnTreeBank-korpukseen. Näiden avulla FiNERin käsittelemä teksti pilkotaan yksittäisiin sanoihin omille riveilleen, ja tunnistuneille entiteeteille

lisätään sopiva entiteettitunniste eli tagi. FiNER annotoi kaiken kaikkiaan 15 erilaista entiteettiä, joita ovat muun muassa ihmisten nimet, ammattinimikkeet tai tittelit, yritykset, päiväykset ja muut ajalliset ilmaukset. Kaikki annotoitavat entiteetit kuuluvat viiteen laajempaan kategoriaan, joita ovat paikka, mittayksikkö, aika, henkilö ja organisaatio. (Silfverberg 2015.) Tässä tutkielmassa huomiota kiinnitettiin vain päiväyksiin, joten tutkimusaineistoon merkityt mahdolliset muut entiteettitunnisteet eli tagit jätettiin tässä huomiotta. FiNERin päiväykselle antama päiväystagi on muotoa ”<TimexTmeDat/>”, ”<TimexTmeDat>” tai ”</TimexTmeDat>”, joista ensimmäinen liitetään yksiosaiseen entiteettiin, ja kaksi jälkimmäistä toimivat moniosaisen entiteetin alku- ja lopputagina (ks. esim. Kettunen et al. 2016b, 8). Yksiosainen päiväysentiteetti on siis pelkkä vuosiluku, kuukauden nimi tai kokonainen numeraalinen päiväys (esimerkiksi 1.3.1940) ja moniosainen päiväysentiteetti on esimerkiksi ”vuonna 1940”.

Näiden entiteettitagien avulla automaattinen tiedonhaku pyrittiin kohdistamaan niihin aineiston kirjeisiin, jotka sisälsivät etsittävän vuoden/kuukauden/päivän. Haku kohdistettiin siis vain niihin tutkimusaineiston kirjeisiin, jotka saivat jonkin kolmesta päiväystunnisteesta tekstiinsä eli kaikki kirjeet, jotka eivät tunnistetta syystä tai toisesta saaneet, jäivät haun ulkopuolelle. FiNER on varsin yleisluontoinen menetelmä eikä sitä ole suunniteltu ensisijaisesti päiväysten poimimiseen, vaan ylipäätään nimettyjen entiteettien, kuten henkilöiden ja paikkojen nimien tunnistamiseen, mutta tässä yhteydessä sen avulla voitiin tarkastella sitä, tunnistettiinko kaikki kirjeiden päiväykset entiteetiksi vai jäikö osa tunnistumatta ja mistä syystä näin tapahtui. Samalla oli mahdollista selvittää, millaisiin muihin yhteyksiin FiNER merkitsi päiväysentiteettitunnisteen.

Aineiston ajaminen FiNER-tunnistuksen läpi tapahtui shell.sis.uta.fi-palvelimen komentorivillä seuraavanlaisella komennolla:

```
xxxxxxx@shell:~/tagtools/finnish-tagtools-1.3.2$  finnish-nertag
< ../../kokoelma_1.txt > ../../finer_kokoelma_1.txt
```

Tutkimusaineisto sijaitsi päähakemistossa, johon nähden FiNER sijaitsi alempana hakemistohierarkiassa kahden kansion päässä, minkä vuoksi ”../”-merkkiä oli tarpeen käyttää kaksi kertaa. FiNER otti käsittelyyn aineiston yksi osakokoelmatiedosto kerrallaan (esimerkissä ”kokoelma_1.txt”) ja ajoi entiteettitunnistustuloksen eri tavalla nimettyyn tiedostoon (esimerkissä ”finer_kokoelma_1.txt”). Tämä työvaihe tehtiin jokaiselle viidelle osakokoelmalle erikseen. Havaittiin, että FiNER oli varsin hidas menetelmä, sillä koko

tutkimusaineiston läpikäymiseen meni aikaa melkein kaksi tuntia, ja laajin yksittäinen kokoelma 2 oli käsittelyssä siitä ajasta melkein puolet. Komennon alussa oleva käyttäjän peruspalvelutunnus on korvattu tässä tekstissä x-merkein.

FiNERin osalta tässä tutkimuksessa tarkasteltiin paitsi päiväysentiteettien tunnistumista ja merkkausta ylipäätään, myös sitä, miten entiteettitagin saaneista kirjeistä pystyttiin löytämään hakuaiheiden kannalta relevantit osumat. Päivämäärätietojen etsimisessä ”hakukoneena” toimi itse kehitellyn Python-ohjelman kuhunkin eri hakuaiheeseen sopivaksi muokattu versio (ks. liitteet 4–5 sekä luku 4.4.3), jossa huomio kohdistettiin vain niihin kirjeisiin, joille FiNER on merkannut jonkin kolmesta päiväysentiteetistä ”<TimexTmeDat/>”, ”<TimexTmeDat>” tai ”</TimexTmeDat>”, joita käsiteltiin toisiinsa nähden yhdenvertaisina rajaamalla tarkasteluun kirjeet ”TimexTmeDat”-termin esiintyvyyden perusteella.

FiNERin osalta pääasiallisin huomio oli siis siinä, tunnistettiinko kirjeiden varsinainen päiväys entiteetiksi, mistä syystä näin ei kenties tapahtunut ja mitä muita ajallisia ilmauksia merkittiin päiväysentiteetin avulla muualle tekstiin. Tutkimuksessa tarkasteltiin myös sitä, oliko FiNERin merkitsemän kirjeaineiston koko samanlainen vai erilainen alkuperäiseen verrattuna, ja mitä vaikutuksia tällä oli mahdollisesti hakutuloksiin.

4.4.3 Itse kehitelty hakuohjelma

Toisena automaattisena vertailtavana menetelmänä oli itse kehitelty Python-ohjelmointikielinen ohjelma, joka avulla kuhunkin kolmeen testikyselyyn sopivia tuloksia pyrittiin löytämään tutkimusaineistosta. Ohjelma rakentui siten, että se toimi ikään kuin hakukoneena eli sen avulla tiedot pyrittiin löytämään aineistosta yksi tiedosto kerrallaan. Selkeyden vuoksi jokaista hakuaihetta eli kyselyä varten saman ohjelman rungosta muokattiin haettavan asian perusteella kolme erilaista versiota, joista ohjelma ”find_year.py” (liite 1) etsi tutkimusaineistosta vain vuonna 1940 kirjoitetut kirjeet, ohjelma ”find_month.py” (liite 2) etsi puolestaan vuoden 1940 maaliskuussa kirjoitetut kirjeet, ja ohjelma ”find_date.py” (liite 3) etsi vuodesta riippumatta kaikki kirjeet, jotka oli kirjoitettu jouluna. Muuttujien arvoja eli hakuaiheiden tietoja vaihtamalla myös muita päiväyksiä olisi ollut mahdollista etsiä, mutta tutkimuksen testihakuun valikoituivat edellä kuvatut ajanjaksot luvussa 4.4.1 perustelluista syistä.

Ohjelman perustoiminta muotoutui siten, että ohjelma valitsi annetussa hakemistossa (path) olevista kaikista mahdollisista tiedostoista sopivat tiedostotyypit eli tässä tapauksessa tekstitiedostot (pääte ”.txt”), joiden nimi alkoi sanalla ”kokoelma_”. Tällä tavoin käsittelyyn valikoituivat vain kaikki viisi tutkimusaineistona olevaa osakokoelmaa, jotka eivät ole menneet FiNER-tunnistuksen läpi. Kyseiset tiedostot sisältävät itse tehdyn manuaalisen annotoinnin, mutta tietoja etsittäessä lisättyä metadataa ei huomioitu, vaan päiväysten etsiminen kohdistettiin varsinaiseen kirjetekstiin, joka oli eroteltu tagien ”<text>” ja ”</text>” avulla, sillä kiinnostus oli nimenomaan siinä, miten Pythonin avulla oli mahdollista louhia mahdollisimman alkuperäistä tekstiä. Myös Elo & Kleemola (2016) hyödynsivät Python-ohjelmointikieltä omassa tutkimuksessaan, ja toisaalta myös aiemman oman kokemukseni perusteella tämä ohjelmointikieli soveltuu varsin hyvin luonnollisella kielellä kirjoitettuihin aineistoihin, eikä suomen kielikään tuota sille ylitsepääsemättömiä ongelmia. Jotta tarkastelu ei olisi kohdistunut aivan koko pitkään kirjetestiin, vaan tekstin alkupuolelle, jossa päiväyskin pääsääntöisesti sijaitsee, rajaus kohdistettiin yksittäisen kirjeen alusta laskien 200 merkin kokoiseen teksti-ikkunaan. Testauksessa pienempi teksti-ikkuna, joka oli vain 100 merkin kokoinen, ei saanut kaikkia päiväyksiä kiinni, joten tästä syystä päädyttiin tarkastelemaan hieman laajempaa aluetta, vaikka osumien tarttuminen muualta kuin päiväyksestä saattaisikin tällöin lisääntyä ja aiheuttaa tuloksiin hälyä.

Tutkimusaineistoa käsiteltiin kussakin eri ohjelman versiossa samalla tavalla eli tiedostojen valinnan jälkeen alkuperäisen lähdetiedoston teksti luettiin ja siitä pyrittiin etsimään haluttu päiväystieto, joka oli määritelty aluksi muuttujan (year1, year2, month1, month2, date1, date2) arvoksi. Sekä vuoden, kuukauden että päivämäärän suhteen etsintä pyrittiin suorittamaan sekä pidemmällä että lyhyemmällä ilmiasulla mahdollisimman kattavien tulosten saavuttamiseksi. Vuodesta haettiin siis sekä ”1940” että ”40”, kuukaudesta ”3” ja ”maalisk”. Vaikka lyhyt vuosiluku oletettavasti tulisi aiheuttamaan virheellisiä osumia, päätettiin sekin ottaa mukaan hakuun, sillä kun tutkimusaineistoa käytiin testimielessä läpi etsimällä pelkästään auki kirjoitettua vuotta 1940, tuli osumia vain neljä kappaletta yhdestä kokoelmasta (kokoelma 4), sillä vuosiluvun lyhentäminen päiväystä kirjoitettaessa on kautta aikain ollut varsin tavallista. Tarkasta päivästä vaihtoehtoisina osumina kelpuutettiin 24.12 ja 25.12 siten, että välimerkkinä kelpuutettiin piste (.), pilkku (,), kauttaviiva (/) sekä välilyönti ( ). Termin ”joulu” suhteen hyväksyttiin sekä pieni että iso alkukirjain ja sana sai jatkua millä tahansa muulla kirjaimella kuin pienellä k-kirjaimella,

jotta mukaan ei tulisi mahdollisia ”joulukuu”-ilmauksia. Vuoden ja tarkan päiväyksen suhteen ohjelma suodatti aineistosta halutut kirjeet vain kertaalleen, mutta myös kuukautta etsittäessä kirjeet valikoitiin ensin vuoden mukaan, minkä jälkeen ohjelma suodatti tästä joukosta osumat määritellyn kuukauden mukaan. Oli oletettavaa, että lyhyt vuosiluku, numerolla ilmaistu kuukausi ja ilmaus ”joulu” saattaisivat aiheuttaa tuloksiin myös epärelevantteja osumia, mutta nämä haluttiin ottaa hakuun mukaan sen vuoksi, että haluttujen ajanjaksojen kirjeitä löytyisi mahdollisimman kattavasti. (Ks. liitteet 1–3.)

Kun koko tiedostosta oli saatu valikoitua sopivat yksittäiset kirjeet, näiden tekstisisältö kirjoitettiin osakokoelma kerrallaan uuteen tiedostoon, jonka nimi oli kopioitu alkuperäisetä tiedostosta hieman muokaten lisäämällä nimen alkuun hakuaiheesta riippuen ”year_”, ”month_” tai ”date_”. Pythonin avulla tekstin käsittely Unicode-muodossa on suoraviivaisempaa, minkä vuoksi merkistö oli tarpeen dekoodata käsittelyvaiheessa tilapäisesti UTF-8:sta Unicodeksi. Lopputulosta varten teksti enkoodattiin Unicodesta takaisin UTF-8:aan.

Edellä kuvattu toimi toisena automaattisena tutkimusmenetelmänä, jonka avulla tutkimusaineistosta pyrittiin löytämään haluttujen ajanjaksojen kirjeitä. Koska FiNER on entiteettitunnistin eikä hakukone, oli itse kehitellyistä ohjelmista tarpeen muokata FiNERin merkkaamaa aineistoa varten omat versionsa, jossa ennen vuoden/kuukauden/päiväyksen etsimistä haku kohdistettiin niihin kokoelmien kirjeisiin, joiden tekstiin oli johonkin kohtaan tullut päiväysentiteettitunniste ”TimexTmeDat” (ks. liitteet 3–6). Toiminnallisesti ohjelmat olivat kuitenkin muutoin keskenään varsin identtiset.

Kaikki erilaiset ohjelmien versiot löytyvät kokonaisuudessaan liitteistä 1–6, ja ohjelmien eri funktioita ja niiden osavaiheita on kommentoitu koodissa selvyyden vuoksi. Sekä ohjelma että kommentit on kirjoitettu englanniksi, sillä suomen kieli ääkkösineen olisi aiheuttanut muutoin merkistöongelmia ohjelmaa suoritettaessa.

5 TULOKSET

Tässä luvussa tarkastellaan tutkimuksessa saatuja tuloksia, ja vastataan siis aiemmin esiteltyihin tutkimuskysymyksiin. Selkeyden ja havainnollisuuden vuoksi tulokset on koottu taulukoihin, joista kerrotaan tarkemmin tekstissä. Huomiota kiinnitetään siihen, mitä on löytynyt ja miksi, ja toisaalta mitä ei ole löytynyt ja miksi. Kunkin haun onnistumista arvioidaan tarkkuuden ja saannin avulla niin osakokoelmittain kuin yleisesti hakuaihetta ajatellen. Tarkkuus kertoo sen, kuinka suuri osa hakutuloksista on oikeasti relevantteja, ja saanti puolestaan sen, kuinka suuri osa kaikista hakuaiheen kannalta relevanteista osumista on löytynyt. Tulos merkitään desimaalilukuna välillä [0, 1], ja mitä lähempänä lukua 1 ollaan, sitä parempi on haun tulos. Mikäli jompaakumpaa lukua ei ole pystytty laskemaan relevanttien osumien puuttumisen vuoksi, on taulukkoon merkitty tähän kohtaan x-kirjain. Muussa tapauksessa tulos on esitetty pääsääntöisesti kahden desimaalin tarkkuudella.

Alaluvussa 5.1 tarkastellaan ensin sitä, millaisin tuloksin FiNER on ylipäätään tunnistanut tutkimusaineistosta päiväyksiä ja missä kohtaa tekstiä merkityt tunnisteet sijaitsevat. Seuraavassa kolmessa alaluvussa tarkastellaan testihakujen tuloksia hakuaihe kerrallaan vertailemalla FiNERin päiväysentiteettimerkittyjen ja koko tutkimusaineiston kirjeiden löytymistä itse kehitellyn Python-ohjelmointikielisen hakuohjelman avulla. Luvussa 5.2 tarkastellaan, millaisia tuloksia on löytynyt haettaessa kirjeitä tietyltä vuodelta, luvussa 5.3 tarkastellaan tuloksia, kun kirjeitä on etsitty tietyn vuoden tietyltä kuukaudelta, ja luvussa 5.4 tarkastellaan jouluna kirjoitettujen kirjeiden löytyvyyttä. Lopuksi alaluvussa 5.5 kootaan lyhyt yhteenveto tutkimustuloksista.

5.1 Päiväysentiteettien löytyminen FiNERillä

Koska tässä tutkimuksessa kiinnostus on muun muassa nimetyissä entiteeteissä ja niiden hyödyntämisessä tiedonhaun apuna, tarkastellaan aluksi sitä, kuinka suomen kielelle kehitelty entiteettitunnistin FiNER suoriutuu päiväysten tunnistamisesta sota-ajan kirjeaineistossa. Tutkimuksessa tarkastellaan, tunnistaako FiNER kaikista kirjeistä päiväyksen nimetyksi entiteetiksi, ja mitä muita mahdollisia päivämääriin liittyviä kohtia FiNER merkkaa päiväysentiteettitunnisteella. Vaikka tässä tutkimuksessa kaikkia kolmenlaista päiväysentiteettitagia käsitellään sinänsä samanarvoisina, tarkastellaan taulukossa 2

aluksi hieman tarkemmin sitä, paljonko FiNER määrällisesti on kutakin eri tunnistetta kirjeaineistoon merkannut. Tunniste ”<TimexTmeDat/>” viittaa siis yksiosaiseen päiväykseen, jollainen on esimerkiksi pelkkä vuosiluku ”1940” tai kokonainen numeraalisessa muodossa oleva päiväys, kuten ”1.3.1940”. Moniosaisessa päiväyksessä, kuten ”vuonna 1940”, toimii alkutunnisteena ”<TimexTmeDat>” ja lopputunnisteena ”</TimexTmeDat>”. Vaikka alku- ja lopputunnisteita on aina sama lukumäärä, kaikki kolme erilaista entiteettitunnistetyyppiä on selvyyden vuoksi laskettu erikseen osakokoelmittain, kuten taulukosta 2 ilmenee.

Taulukko 2. Eri päiväysentiteettitunnisteiden esiintyvyys kirjeaineistossa (kpl)

tunniste-tyyppi	**<TimexTmeDat/>**	**<TimexTmeDat>**	**</TimexTmeDat>**	**yht.**
kokoelma 1	30	1	1	32
kokoelma 2	96	4	4	104
kokoelma 3	60	2	2	64
kokoelma 4	28	3	3	34
kokoelma 5	21	1	1	23
yht.	235	11	11	257

Kuten taulukosta 2 havaitaan, eniten FiNER on tunnistanut yksiosaisia päiväyksiä, joita on kaikista merkityistä tunnisteista yli 90 %. Koko tutkimusaineistosta FiNER on löytänyt vain yksitoista moniosaista päiväystä, mikä on erittäin vähän. Kaiken kaikkiaan eri päiväystunnisteita on koko aineistoon merkitty 257 kappaletta, mikä sekin on erittäin vähän, kun koko tutkimusaineistossa on yksittäisiä kirjeitä kuitenkin 992 kappaletta. Näin ollen vain vajaassa kolmanneksessa koko aineiston kirjeistä on FiNER tunnistanut entiteetiksi jonkin päiväykseen liittyvän kohdan. Taulukossa 3 tarkastellaan tarkemmin, miten tämä määrä jakaantuu eri osakokoelmiin. Lisäksi huomiota kiinnitetään siihen, kuinka

moni tunnisteista sijaitsee kirjeen varsinaisen päiväyksen kohdalla ja minkä verran tunnisteita on merkitty muualle kirjetekstiin.

Taulukko 3. Entiteettitunnisteen ”TimexTmeDat” esiintyvyys kirjeaineistossa (kpl)

tunnisteen sijainti	päiväyksessä	muualla tekstissä	kaikkialla yhteensä	eri id-numeroita	kirjeitä
kokoelma 1	4	28	32	24	143
kokoelma 2	4	100	104	78	416
kokoelma 3	28	36	64	50	160
kokoelma 4	3	31	34	23	154
kokoelma 5	1	22	23	18	120
yht.	40	217	257	193	992

Kuten taulukosta 3 nähdään, testikokoelman 992 kirjeestä vain 193 yksittäistä kirjettä eli noin 19 % kaikista kirjeistä sisältää päiväysentiteettitunnisteen johonkin tekstin kohtaan liittyen. Suurin osa eli noin 84 % päiväysentiteeteistä on merkitty muualle kirjetekstiin kuin sen varsinaiseen päiväykseen. Kokoelmassa 3 tulos on parhaiten tasapainossa, sillä entiteettitunnisteita on melkein saman verran sekä päiväyksessä että muualla tekstissä. Vertailemalla tunnisteiden yhteislukumäärää ja eri id-numeroiden lukumäärää osakokoelmittain havaitaan, että tunnisteita on merkitty toisinaan useampi samaan yksittäiseen kirjeeseen, sillä eri id-numeroiden lukumäärä on kaikkien viiden osakokoelman osalta pienempi kuin montako tunnistetta FiNER on ylipäätään aineistoon merkinnyt. Varsinaisen päiväyksen kohdalla tunnisteita on pääsääntöisesti erittäin vähän, vain noin 16 % kaikista kirjeistä, eli FiNERin hyödyntäminen kirjeen päiväyksen tunnistamisen apuna on varsin huonoa, sillä se ei tunnista kirjeen varsinaista päiväystä entiteetiksi kuin harvoin.

Muualla tekstissä on paljon merkittyjä päiväysentiteettejä, ja tämä johtuu siitä, että kirjeen tekstissä mainitaan usein kuukausia tai vuosilukuja ja toisinaan myös tarkempia

päiväyksiä, jotka FiNER tunnistaa ja merkitsee. FiNER ei tunnista ainakaan tässä aineistossa päiväysentiteeteiksi lainkaan juhlapyhiä, kuten joulua, helatorstaita tai juhannusta, eikä toisaalta myöskään nimipäiviä. Vuosiluvun tunnistaakseen FiNER tarvitsee siihen kaikki neljä numeroa, eli vuosilukuna esimerkiksi 1940 tunnistetaan entiteetiksi, mutta 40 ei, vaikka se liittyisi muuhun päiväykseen. Tarkka päivämäärä tunnistuu ainoastaan, jos se on kirjoitettu tavallisin numeroin ja erotettu toisista numeroista pistein. Millään muulla välimerkillä eroteltuna FiNER ei tunnista päiväystä entiteetiksi. Tarkasta päiväyksestä saa puuttua vuosiluku, mutta se kuitenkin tunnistetaan silti entiteetiksi, mutta tunnistuvalle päiväykselle on tiukasti rajattu muoto. Esimerkiksi 24.12. tunnistetaan päiväykseksi, mutta 24.12 tai 24/12 ei. Jos vuosilukukin on mukana, 24.12.1940 tunnistettaisiin päiväykseksi, mutta 24.12.40 ei, vaikka muuten päiväys olisi muodoltaan hyväksyttävissä rajoissa. Pelkkä kuukausi tunnistetaan päiväysentiteetiksi, jos se on kirjoitettu kirjaimin, ja numeraalisena osana päiväystä se tunnistetaan edellä kuvatuissa tilanteissa.

FiNERissä on myös entiteettitunniste muotoa ”TimexTmeHrm” kellonajoille, ja FiNERin merkkaustuloksia läpi käydessä huomattiin, että kokoelmissa 1, 2 ja 3 oli kirjeen varsinaiseen päiväykseen tullut tämä tunniste eli päiväys olikin tunnistunut kellonajaksi. Kokoelmassa 1 näitä tapauksia oli 9 kpl, kokoelmassa 2 puolestaan 17 kpl ja kokoelmassa 3 vain 1 kpl. Kokoelmassa 4 ja 5 ei kellonaikaentiteettitunnisteita ollut päiväyksen kohdalla lainkaan. Päiväyksen tunnistuminen kellonajaksi on sikäli ymmärrettävää, että mikäli vuosiluku on päivän ja kuukauden tiedosta irrallaan tai puuttuu kokonaan ja muu osa on muotoa 9.11, voi tämän tulkita varsin helposti myös kellonajaksi.

Kirjeiden päiväysten tunnistuminen entiteetiksi on tutkimusaineiston osalta varsin heikkoa, sillä suurimmassa osassa kirjeitä vuosiluku on merkitty lyhyessä muodossaan käyttämällä vain vuosiluvun kahta viimeistä numeroa, ja FiNER tunnistaa vuosiluvun vain, jos se on kirjoitettu kokonaan. Kokoelmassa 3 kirjeiden kirjoittajat ovat käyttäneet muuhun aineistoon verrattuna poikkeuksellisen paljon kokonaan auki kirjoitettua vuosilukua yhdistettynä FiNERin tunnistamaan päivämäärämuotoon, jollainen on esimerkiksi 24.12.1940, minkä vuoksi tässä osakokoelmassa kirjeiden varsinaisia päiväyksiä on tunnistunut kaikkein eniten. Kaikkia päiväyksiä ei tästäkään kokoelmasta ole silti tunnistettu. On varsin ymmärrettävää, että päiväysentiteettejä on merkitty myös muualle kirjetekstiin, sillä kirjeissä kerrotaan hyvin paljon sekä menneistä että tulevista tapahtumista, jolloin ajallisia ilmauksia tulee muuallekin kuin päiväykseen. Laajemmassa temporaalisessa

tarkastelussa FiNERin merkinnöistä on siis hyötyä, mutta tarkkojen kirjeen alkupuolella mainittujen päiväysten suhteen tulos jää varsin heikoksi.

Tutkimusaineistossa päiväys on tavallisimmin merkitty numeraalisesti siten, että päivän ja kuukauden välimerkkinä on käytetty pistettä tai kauttaviivaa, ja vuosiluku on kirjoitettu vain kahden numeron avulla, kuten aiemmin mainittiin. Jos FiNER tunnistaisi päiväyksiä nykyistä laajemmin erilaisilla variaatioilla, olisi päiväysentiteettien tunnistuminen ollut parempaa, jolloin tuloskin olisi ollut huomattavasti parempi. Koska testihakujen aineistoksi hyväksytään FiNERin osalta vain päiväysentiteetin sisältävät kirjeet, on kirjeiden lukumäärä huomattavasti todellisuutta pienempi, millä on väistämättä vaikutuksia myös hakutuloksiin. Toisaalta aineistoon on hyväksytty mukaan kaikki kirjeet, joissa on jonkinlainen ”TimexTmeDat”-entiteettitunniste kirjetekstissä eli tunnisteen ei ole tarvinnut sijaita pelkästään varsinaisessa päiväyksessä. FiNERin osalta aineistoon mahtuivat tämän vuoksi mukaan kaikki 193 tunnisteen saanutta kirjettä sen sijaan, että hyväksyttyjä kirjeitä olisi ollut koko aineistosta vain 40.

Paitsi että FiNER tunnistaa kirjeiden päiväykset todella huonosti niiden kirjoitusasusta ja ohjelmaan tehdyistä rajoituksista johtuvista syistä, päiväysentiteettitunniste on liitetty myös sellaisiin kohtiin, joissa ei mitään päiväykseen liittyvää tietoa kuitenkaan ole. Vuosilukujen suhteen kaikissa merkatuissa maininnoissa ei ole kyse kirjeen päiväykseen liittyvästä ajankohdasta, vaan tekstissä on mainittu esimerkiksi ihmisten syntymävuosia. Jonkin verran tunnisteen ovat saaneet myös muut vuosiluvun kaltaiset nelinumeroiset luvut, jotka eivät kuitenkaan tekstiä lukiessa ole tulkittavissa lainkaan vuosiluvuiksi. Todellisuudessa FiNERin merkkaamien päiväysentiteettitunnisteiden lukumäärä todellisissa päiväyksiin liittyvissä tilanteissa jää siis 257 löytynyttä tunnistetta huonommaksi, mutta jatkokäsittelyä ja tiedonhakua ajatellen tällainen erottelu olisi ollut hankala toteuttaa, joten kaikki ”TimexTmeDat”-tunnisteen saaneet kirjeet hyväksyttiin aineistoon.

5.2 Tulokset pelkkää vuosilukua haettaessa

Ensimmäisenä sekä koko tutkimusaineistosta että FiNERin merkkaamasta aineistosta haluttiin poimia kaikki vuonna 1940 kirjoitetut kirjeet, ja kummassakin tapauksessa hyödynnettiin itse kehiteltyä Python-ohjelmointikielistä ”hakukonetta”. Ohjelmat ovat muuten täysin identtiset, mutta FiNERin osalta tarkasteltavaksi hyväksyttiin vain päiväysentiteettitunnisteen saaneet kirjeet (ks. liitteet 1 ja 4). Hakusanoiksi eli muuttujien ”year1”

ja ”year2” arvoiksi asetettiin sekä ”1940” että ”40”, jotta haku kohdistuisi myös lyhyessä muodossa kirjoitettuun vuosilukuun. Haun tulokset ilmenevät taulukosta 4.

Taulukko 4. Löydettyjen vuonna 1940 kirjoitettujen kirjeiden tulosten vertailu (kpl)

menetelmä	**FiNER**	**find_year.py**	**manuaalinen läpikäynti**
kokoelma 1	6	34	34
kokoelma 2	5	28	29
kokoelma 3	0	1	0
kokoelma 4	7	92	93
kokoelma 5	4	40	38
yht.	22	195	194

Taulukkoon 4 on koottu haussa löytyneet osumat osakokoelmittain kahden eri menetelmän avulla, ja vertailukohtana on itse tehty manuaalinen läpikäynti, josta ilmenee, paljonko vuonna 1940 kirjoitettuja kirjeitä tutkimusaineistossa todellisuudessa on. Kaiken kaikkiaan koko aineistossa on vuonna 1940 kirjoitettuja kirjeitä siis 194 kappaletta. Oma ohjelma find_year.py löytää hakutulokseen osumia kaiken kaikkiaan 195, mutta FiNER vain 22, mikä selittyy sillä, että tässä läpikäytävä aineisto on huomattavasti suppeampi kuin koko tutkimusaineisto. Taulukkoon 4 on kerätty kaikki eri menetelmien avulla löytyneet kirjeet ilman, että niiden relevanssiin on kiinnitetty huomiota manuaalista läpikäyntiä lukuun ottamatta. Osumien relevanssia tarkastellaan tarkemmin taulukossa 5, jossa hakutulokset on arvioitu tarkkuuden ja saannin osalta osakokoelmittain sekä koko hakuaiheen kannalta kummankin automaattisen menetelmän osalta.

Taulukko 5. Ensimmäisen hakuaiheen tulosten arviointi tarkkuuden ja saannin osalta

menetelmä	**find_year.py**		**FiNER**	
evaluointi	**tarkkuus**	**saanti**	**tarkkuus**	**saanti**
kokoelma 1	1	1	1	0,18
kokoelma 2	0,89	0,86	0,8	0,14
kokoelma 3	0	x	x	x
kokoelma 4	0,98	0,97	1	0,08
kokoelma 5	0,88	1	1	0,11
koko haku	0,959	0,964	0,95	0,11

Kuten taulukosta 5 havaitaan, oma ohjelma suoriutuu kautta aineiston paremmin kuin FiNER relevanttien kirjeiden löytymisen osalta, mistä oman ohjelman korkeat saantiluvut kertovat. Toisaalta tarkkuus on molemmilla menetelmillä lähestulkoon yhtä hyvä eli hakutuloksiin osuneet kirjeet ovat pääsääntöisesti relevantteja eli nimenomaan vuonna 1940 kirjoitettuja. Kokoelmassa 3 ei ole lainkaan vuonna 1940 kirjoitettuja kirjeitä, joten sen osalta tuloksia ei voi laskennallisesti arvioida samalla tavoin kuin muiden kokoelmien suhteen, joskin find_year.py-ohjelma löysi haulla yhden tuloksen, joka ei kuitenkaan ollut relevantti. Oma ohjelma löytää koko tutkimusaineistosta haettavan vuoden kannalta relevantit kirjeet varsin hyvin toisin kuin FiNER, jonka tulos saannin osalta jää siis heikommaksi siksi, että päiväysentiteettitunnisteen saaneiden kirjeiden osuus oli varsin vähäinen eli vain vajaa kolmasosa kaikista tutkimusaineiston kirjeistä.

Eri osakokoelmien välillä on jonkin verran eroa siinä, paljonko epärelevantteja kirjeitä osui mukaan hakutulokseen, ja toisaalta jotkin relevantit kirjeet jäivät myös löytymättä. Oman ohjelman suhteen epärelevantteja tuloksia osui mukaan hyvin vähän tai ei lainkaan, ja vain kokoelman 2 ja 4 osalta jäi muutama relevantti kirje löytymättä. Löytymättä jääneiden relevanttien kirjeiden osalta kyse on ensinnäkin siitä, että koska haku kohdistettiin kirjeen alusta 200 merkkiä sisältävään teksti-ikkunaan, saattaa päiväys jäädä löytymättä siksi, että se sijaitseekin poikkeuksellisesti myöhemmin kirjeessä. Toisaalta yhdessä

löytymättä jääneessä vuosiluvussa on OCR-virhe tai vuosiluvun numeroiden välissä on ylimääräisiä välilyöntejä, jolloin niitä ei voikaan löytyä. Vuosilukujen suhteen tutkimusaineisto on kuitenkin varsin vähän OCR-virheitä sisältävää eli vuosiluvut ovat optisesti tunnistuneet varsin hyvin – vuoden 1940 osalta vain yhdessä vuosiluvussa on OCR-virhe. Epärelevanttien tulosten osalta mukaan tullut kirje sisältää tavallisesti arkistointia varten aikanaan luodun, kokoelman sisällä yksilöivän SAK-tunnistenumeron, jossa on ollut luku 40 jossain muodossa (40, 140, 240, 340) ja joka sijaitsee varsin alkupuolella kirjetekstiä. Toisaalta haku on tarttunut yhdessä tapauksessa myös postinumeroon sekä kirjeen kirjoittajan virheellisesti merkitsemään vuosilukuun, jolloin päiväys ei myöskään ole oman tulkinnan perusteella tämän haun osalta relevantti.

Pelkällä lyhyen vuosiluvun haulla myös pidemmät vuosiluvun kirjoitusmuodot olisivat toki löytyneet, mutta koska ajatuksena oli etsiä nimenomaan vuonna 1940 kirjoitetut kirjeet, mukaan haluttiin ottaa myös tämä pidempi muoto. Koska manuaalisen läpikäynnin pohjalta päiväysten vuosilukujen havaittiin kuitenkin olevan pääsääntöisesti lyhyessä muodossa, oli hakuun tarpeen sisällyttää myös tämä, vaikka se oletettavasti ja tutkimuksessa saatujen tulostekin valossa aiheuttaa jonkin verran hälyä eli epärelevantteja osumia. Näitä hakutuloksissa on kuitenkin varsin vähän. Mikäli jonkin osakokoelman alkuperäinen SAK-kokoelmanumero olisi ollut 40, olisi hakutuloksiin tullut huomattavasti enemmän epärelevantteja osumia, ja näin olisi ollut, jos olisi haluttu etsiä esimerkiksi vuoden 1941 kirjeet, mistä syystä tämä vuosi rajattiin tällä kertaa pois.

Vaikka ohjelmallisesti haku eli kirjeiden poiminta aineistosta suoritetaan molemmilla menetelmillä samaan tapaan, on erityisesti määrällisesti suurta eroa siinä, käytetäänkö haun lisätukena nimettyjen entiteettien tunnistamista vai ei. Koska FiNERin tulokset päiväysten tunnistamisessa tässä kirjeaineistossa ovat varsin alhaiset, ei kunnollista vertailua määrien suhteen pysty tekemään, mutta tuloksia arvioitaessa tarkkuuden suhteen näiden kahden menetelmän välillä ei ole suurtakaan eroa. Saannissa ero eri menetelmien välillä on suuri, mutta se liittyy ennen kaikkea kummankin eri hakumenetelmän käytettävissä olevan aineiston kokoon.

5.3 Tulokset vuoden ja kuukauden avulla haettaessa

Toisessa testihaussa hakua haluttiin tarkentaa rajaamalla mukaan tietyn vuoden ja edelleen tietyn kuukauden kirjeet. Vuoden 1940 osalta kuukaudeksi valittiin maaliskuu, koska

talvisota loppui silloin ja kirjeitä oletettavasti vielä lähetettiin, koska rintamalta ei heti päässyt kotiin. Tälläkin kertaa hakuohjelmasta on kaksi melkein samanlaista versiota: toiselle annetaan koko tutkimusaineisto ja toiselle FiNERin päiväysentiteettitunnisteen saaneet kirjeet (ks. liitteet 2 ja 5). Ohjelma poimii ensin hakutermin ”1940” tai ”40” saaneet kirjeet (muuttujien ”year1” ja ”year2” arvot) ja etsii näistä sen jälkeen numeron ”3” (muuttuja ”month1”) tai ”maalisk”-alkuiset sanat (muuttuja ”month2”) tekstin alkupuolella (200 merkin teksti-ikkuna) sisältävät kirjeet, jotta ei olisi väliä, onko kuukausi kirjoitettu kirjeessä numeroin vai kirjaimin. Löytyneet kirjeet ilmenevät osakokoelmittain eri menetelmien osalta taulukosta 6.

Taulukko 6. Löydettyjen vuoden 1940 maaliskuussa kirjoitettujen kirjeiden tulosten vertailu (kpl)

menetelmä	**FiNER**	**find_month.py**	**manuaalinen läpikäynti**
kokoelma 1	4	20	7
kokoelma 2	3	10	0
kokoelma 3	0	1	0
kokoelma 4	3	30	15
kokoelma 5	2	24	14
yht.	12	85	36

Kaiken kaikkiaan vuonna 1940 kirjoitetusta 194 kirjeestä siis vain 36 on manuaalisen tarkastelun perusteella kirjoitettu maaliskuussa, kuten taulukosta 6 ilmenee. Siinä missä oma ohjelma find_month.py löytää tuloksia yli tuplaten vastaavan määrän, on FiNERin löytämä lukumäärä vähäisempi ja sinänsä lähempänä todellista määrää. Kumpikin menetelmä löytää haulla kirjeitä myös kokoelmista 2 ja 3, vaikka näissä ei todellisuudessa ole lainkaan maaliskuussa 1940 kirjoitettuja kirjeitä. Erityisesti oman ohjelman suuri hakutulosten määrä viittaa suureen epärelevanttien osumien määrään, ja tätä voidaan arvioida taulukosta 7, johon on merkitty osakokoelmittain sekä koko haun osalta tulosten tarkkuus ja saanti.

Taulukko 7. Toisen hakuaiheen tulosten arviointi tarkkuuden ja saannin osalta

menetelmä	**find_month.py**		**FiNER**	
evaluointi	**tarkkuus**	**saanti**	**tarkkuus**	**saanti**
kokoelma 1	0,35	1	0,5	0,29
kokoelma 2	0	x	0	x
kokoelma 3	0	x	x	x
kokoelma 4	0,47	0,93	0	0
kokoelma 5	0,58	1	0,5	0,07
koko haku	0,41	0,97	0,25	0,08

Kun hakua tarkennetaan, tulokset heikkenevät molemmilla menetelmillä erityisesti tarkkuuden suhteen, ja FiNERin osalta myös saanti heikkenee entisestään. Saanti pysyy erittäin hyvänä oman ohjelman osalta, mikä kertoo siitä, että find_month.py löytää ongelmitta lähes kaikki relevantit tulokset, vaikka mukaan tulee tällä kertaa myös runsaasti epärelevantteja osumia, mikä heikentää haun tarkkuutta. Epärelevanttien kirjeiden mukaan tulo liittyy erityisesti numerolla ”3” hakemiseen, jolloin haku poimii mukaan kaikki nekin kirjeet, joissa kyseinen numero esiintyy kirjeen alkupuolella muualla kuin kuukauden kohdalla. Numero kolme saattaa esiintyä päiväyksessä myös päivän tai vuosiluvun kohdalla, minkä lisäksi se saattaa sijaita myös muualla tekstin joukossa. Suurin osa epärelevantiksi katsotuista find_month.py-ohjelman hakutulokseen poimimista kirjeistä sisältävätkin numeron kolme joko muualla päiväyksessä tai tekstissä, minkä lisäksi kyseinen numero esiintyy myös usein alkuperäisessä aikanaan arkistointia varten luodussa yksilöivässä SAK-tunnistenumerossa. Alun perin virheellisesti mukaan on tullut muutama yksittäinen tapaus myös jo ensimmäisessä haun rajauksessa eli vuosiluvussa, mikäli numero 40 esiintyy SAK-tunnistenumerossa.

Oman ohjelman hakutuloksena kaikki muut relevantit eli maaliskuussa 1940 kirjoitetut kirjeet tulevat mukaan tulokseen, mutta kokoelmasta 4 jää yksi kirje löytymättä. Tässä kirjeessä päiväys on vuosiluvun 1940 lisäksi ”Marianpäivä”, joka on juhlapyhänä

kalenterin avulla tulkittu sijoittumaan ajallisesti maaliskuulle (25.3.), mutta jota ei kirjetekstissä ole tarkemmin määritelty maaliskuulle, minkä vuoksi myöskään haku ei pääse siihen kiinni.

FiNERin kautta päiväysentiteettitunnisteen saaneiden kirjeiden osalta tarkennetulla haulla tuloksia löytyy määrällisesti vähän, eikä niiden tarkkuuskaan ole kovinkaan suuri. Epärelevantteja osumia tulee mukaan hakutuloksiin edellä mainituista syistä myös tällä menetelmällä, eikä suurin osa relevanteista kirjeistä osu lainkaan mukaan tuloksiin tarkasteltavan aineiston suppeuden vuoksi. Saanti onkin tässä haussa erittäin huono, vain 0,08. Entiteettien tunnistumisesta ei siis tässäkään haussa ole apua, sillä FiNER on tunnistanut päiväyksiä aineistosta niin vähän. Toisaalta kun FiNERin hakuohjelma ylipäätään löytää vuotta 1940 haettaessa 22 kirjettä ja niistä 12 se kerää hakutuloksiin tarkennettuna maaliskuuta etsittäessä, on tulos sinänsä kohtalainen, vaikka sekä saanti että erityisesti tarkkuus aiemmasta heikkenevätkin.

Molemmilla vertailtavilla hakumenetelmillä havaitaan, että hälyn eli hakutuloksessa olevien epärelevanttien osumien määrä lisääntyy varsin paljon hakua tarkennettaessa, mikä liittyy ennen kaikkea hakusanana olevaan numeroon kolme. Numeron esiintymää tekstissä olisi voinut kenties rajoittaa tarkemmin, mutta sopivien rajausten määrittely on haastavaa, jotta saanti ei heikkenisi liikaa. Koko aineiston osalta saanti on kuitenkin tässäkin tapauksessa hyvä, joten sen suhteen hakutulosta voi pitää kokonaisuudessaan kohtalaisen hyvänä ainakin find_month.py-ohjelman suhteen.

5.4 Tulokset tarkkaa päiväystä haettaessa

Kolmannen hakuaiheen tarkoituksena oli pyrkiä löytämään kirjeitä tarkan päivämäärän perusteella. Koska joulu on hyvin voimakkaasti perhejuhla, ja se on ollut oletettavasti myös sota-aikana paljon mielessä erityisesti vuodenvaihteen molemmin puolin, valittiin joulu sopivaksi tarkaksi ajankohdaksi. Tarkaksi päivämääräksi hyväksyttiin kaksi vaihtoehtoa: jouluaatto eli 24.12. sekä joulupäivä eli 25.12., joihin molempiin voidaan viitata, jos päiväys on kirjallisesti ilmaistuna ”jouluna”. Hakuohjelmaan hyväksyttiin siis kumpikin näistä päivämääristä vaihtoehtoisina, minkä lisäksi myös sanallisen ilmauksen kirjeet haluttiin mukaan tuloksiin. Jotta haku tarttuisi termiin ”joulu”, mutta ei ”joulukuu”, sopivasta termistä oli tarpeen valikoida tämä pois (ks. liitteet 3 ja 6). Koska ylipäätään

jouluna kirjoitetut kirjeet kiinnostivat, ei vuotta tässä haussa haluttu tarkentaa. Tällä haulla löytyneet tulokset ilmenevät osakokoelmittain ja menetelmittäin taulukosta 8.

Taulukko 8. Löydettyjen jouluna kirjoitettujen kirjeiden tulosten vertailu (kpl)

menetelmä	**FiNER**	**find_date.py**	**manuaalinen läpikäynti**
kokoelma 1	2	13	3
kokoelma 2	6	26	0
kokoelma 3	0	12	1
kokoelma 4	4	31	1
kokoelma 5	3	15	2
yht.	15	97	7

Kuten taulukosta 8 ilmenee, manuaalisen läpikäynnin perusteella jouluna kirjoitettuja kirjeitä katsotaan koko tutkimusaineistossa olevan vain seitsemän. Kokoelmassa 1 on määrällisesti eniten jouluna kirjoitettuja kirjeitä (3 kpl), mutta niitä löytyy vähintään yksi kaikista muista osakokoelmista lukuun ottamatta kokoelmaa 2, jossa jouluna kirjoitettua kirjeitä ei ole lainkaan. Kumpikin automaattinen haku löytää osumia kuitenkin enemmän, oma ohjelma find_date.py jopa yli kymmenkertaisen määrän, mikä liittyy siihen, että rajattu teksti-ikkuna kohdistaa haun 200 ensimmäiseen kirjetekstin merkkiin, jolloin kaikki kirjeen alussa mainitut jouluun liittyvät sanat saavat ne tarttumaan mukaan hakutuloksiin. Sanallinen hakutermi on kuitenkin tarpeen olla mukana, jotta myös pelkällä kirjallisella päiväyksellä, kuten ”jouluna” tai ”jouluaattona” löytyneet kirjeet löytyisivät myös. Taulukosta 9 nähdään, mitkä hakutulosten tarkkuus ja saanti ovat ylipäätään hakutuloksen kannalta sekä eri osakokoelmissa.

Taulukko 9. Kolmannen hakuaiheen tulosten arviointi tarkkuuden ja saannin osalta

menetelmä	**find_date.py**		**FiNER**	
evaluointi	**tarkkuus**	**saanti**	**tarkkuus**	**saanti**
kokoelma 1	0,23	1	0,5	0,33
kokoelma 2	0	x	0	x
kokoelma 3	0,08	1	0	0
kokoelma 4	0,03	1	0,25	1
kokoelma 5	0,13	1	0,33	0,5
koko haku	0,07	1	0,2	0,43

Kuten taulukosta 9 nähdään, aiempiin hakuihin verrattuna kummankin menetelmän tarkkuus heikkenee entisestään, sillä hälyä eli epärelevantteja osumia on hakutuloksessa mukana niin runsaasti. Toisaalta saanti paranee kummallakin menetelmällä siten, että oma ohjelma löytää kaikki relevantit kirjeet, ja FiNERin merkkaamien kirjeiden osaltakin saanti paranee kaikkien kolmen eri hakuaiheen parhaimmaksi, vaikka pysyy toiseen tarkasteltavana olevaan menetelmään verrattuna edelleen heikompana. Tarkkuus on erittäin heikko, mikä liittyy hakutermin ”joulu” ongelmallisuuteen, sillä se tarttuu kaikkiin tekstin alkupuolella oleviin mainintoihin joulusta. Muutamaa yksittäistä poikkeusta lukuun ottamatta kaikki epärelevantit kirjeet on kirjoitettu lokakuun ja helmikuun välisenä aikana, jolloin maininnat joulusta kirjetekstissä ovat varsin luontevia, sillä joulu on joko ennalta tai jälkikäteen ollut monilla mielessä ja siitä on kirjoitettu paljon myös muulloin kuin jouluaattona tai joulupäivänä. Kaikkien epärelevanttien osumien osalta kyse onkin siitä, että joulu on mainittu muualla kirjetekstin alkupuolella.

Siinä missä oma ohjelma löytää runsaan hälyn lisäksi myös kaikki relevantit kirjeet aineistosta, ei FiNER yllä aivan samanlaiseen tulokseen, sillä sen kautta hakutuloksiin osuu vain kolme seitsemästä relevantista kirjeestä. Syy relevanttien kirjeiden puuttumiseen on

se, ettei niihin ole merkitty päiväysentiteettitunnistetta, jolloin ne eivät ole päässet FiNERin aineistoon mukaan. Tälläkin menetelmällä hakutuloksiin osuu paljon myös epärelevantteja osumia, mutta koska niiden määrä on suhteessa huomattavasti vähäisempi kuin oman ohjelman tuloksissa, on saanti tällä kertaa huomattavasti parempi.

5.5 Yhteenveto tuloksista

Yhteenvetona todettakoon, että FiNER ei ole kovin hyvä tunnistamaan päiväyksiä entiteetiksi varsinkaan tutkimuksen kohteena olevasta kirjeaineistosta, jossa erityisesti vuosiluku on useimmiten kirjoitettu vain kahden viimeisen luvun avulla, sillä FiNER ei tällöin tunnista lukua lainkaan vuodeksi. Myös erilaisten muiden päiväysten merkintätapojen vuoksi kirjeiden varsinaisen päiväyksen tunnistuminen on hyvin huonoa, eikä entiteettitunnisteita ole merkattu muutenkaan kuin vajaalle kolmasosalle tutkimusaineistosta. Lisäksi suurin osa tunnisteista sijaitsee muualla tekstissä kuin päiväyksen kohdalla. FiNERin hyödyntäminen päiväysten tunnistamisessa ei toistaiseksi siis paranna tuloksia, vaan pikemminkin huonontaa niitä, mikä havaitaan myös kaikkien kolmen testihaun tulosten tarkastelussa ja arvioinnissa.

Hakutulosten tarkastelussa itse kehitelty ohjelma suoriutuu pääsääntöisesti varsin hyvin ja löytää relevantit kirjeet aineistosta muutamia poikkeuksia lukuun ottamatta kattavasti, mikä ilmenee vähintäänkin kohtalaisen korkeina saantilukuina. Myös tarkkuus pysyy hyvänä etsittäessä kirjeitä pelkän vuosiluvun avulla, mutta se heikkenee kahdessa muussa testihaussa, sillä hakusanoihin ja haun kohdistukseen liittyen mukaan tulee myös runsaasti epärelevantteja osumia.

FiNERin entiteettimerkkauksen pohjalta tarkasteltava aineisto on huomattavan suppea alkuperäiseen testikokoelmaan verrattuna, mikä heikentää erityisesti saantia huomattavasti, kun kaikille relevanteille kirjeille ei ole pystytty lisäämään päiväysentiteettitunnistetta. Tarkkuus on myös varsin huono vastaavista syistä kuin oman ohjelmankin suhteen, mutta hyviin tuloksiin päästään tälläkin menetelmällä pelkkää vuosilukua haettaessa. Itse kehitelty tutkimusaineistoon paremmin räätälöity ohjelma suoriutuu siis kautta linjan paremmin, sillä FiNERin entiteettitunnistuksessa on päiväysten osalta tällä hetkellä suuria rajoitteita ja erinäisistä syistä johtuvia tunnistusongelmia ainakin tämän tutkimuksen aineistona olevissa sota-ajan kirjeissä.

6 JOHTOPÄÄTÖKSET JA DISKUSSIO

Tässä luvussa kootaan yhteen tutkimuksessa saadut tulokset ja huomiot sekä tehdään ehdotuksia jatkoa ajatellen niin tutkimusaiheiden kuin menetelmien kehittelyn osalta. Alaluvussa 6.1 tehdään tarkempia johtopäätöksiä tämän tutkimuksen tuloksista, ja luvussa 6.2 suunnataan katse tulevaisuuteen.

6.1 Johtopäätöksiä tutkimustuloksista

Kirjeiden etsiminen päivämäärän perusteella on yksi merkittävimpiä keinoja, kun ollaan kiinnostuneita esimerkiksi tietyn sotavaiheen kirjeistä. Relevanttien kirjeiden etsiminen automaattisin menetelmin on perusteltua etenkin, kun aineisto on digitaalisessa muodossa ja sitä on määrällisesti paljon. Päiväykseen tarvitsee tällöin päästä jotenkin kiinni, ja yksi varteenotettava keino tässä on päiväyksen määrittely nimetyksi entiteetiksi, jonka jokin automaattinen tunnistin voi aineistoon merkitä. Olemassa olevista valmiista menetelmistä FiNER ei suoriudu kirjeaineistosta kuitenkaan kovinkaan hyvin, sillä kirjeiden päiväykset eivät pääsääntöisesti ole FiNERin parhaiten tunnistamassa säännöllisessä muodossa, jossa päivä, kuukausi ja vuosi on kirjoitettu numeroin ja erotettu toisistaan pisteillä, ja vuosiluku on kirjoitettu kokonaisuudessaan auki. Kirjeissä on käytetty pääasiassa vuosiluvun lyhennettyä muotoa kahden viimeisen luvun osalta, ja välimerkki saattaa olla jokin muukin kuin piste, minkä vuoksi entiteettien tunnistus erityisesti juuri kirjeen päiväyksen suhteen on hyvin heikkoa, eikä niitä tunnistunut koko aineistosta kovinkaan paljon. Suurin osa päiväysentiteettitunnisteista sijaitsi muualla kuin kirjeen varsinaisen päiväyksen kohdalla, eivätkä kaikki niistäkään liittyneet suoranaisesti ajalliseen ilmaukseen, kuten vuosilukuun tai kuukauden nimeen, vaan myös muita vuosiluvun kaltaisia kohtia oli merkattu. Toisaalta FiNER ei tunnistanut juhlapyhiä tai nimipäiviä lainkaan entiteeteiksi, mikä heikensi päiväysten tunnistumista entisestään. Tämä kaikki vaikutti osaltaan myös varsinaisten tutkimuksen puitteissa testattujen hakuaiheiden tuloksiin, sillä FiNERin aineisto jäi vain murto-osaan koko tutkimusaineistosta, jolloin suuri osa relevanteista kirjeistä rajautui pois, eikä niitä siten olisikaan voinut löytyä.

Itse kehitelty hakuohjelma toimi sinänsä varsin tehokkaasti, että kaikki relevantit kirjeet löytyivät aineistosta muutamaa poikkeusta lukuun ottamatta. Vaikka haku eli käytännössä tässä tapauksessa tekstinlouhinta oli rajattu kohdistumaan vain 200 ensimmäiseen

merkkiin kirjetekstin alusta laskien, oli tämäkin teksti-ikkuna useimmissa tapauksissa liian laaja, ja epärelevantteja osumia tuli mukaan kohtalaisen paljon haettaessa epämääräisemmillä hakutermeillä, joita olivat numerot 40 ja 3 sekä joulu. Toisaalta osa päiväyksistä jäi silti tämän rajauksen ulkopuolelle, kun ne sijaitsivatkin poikkeuksellisesti paljon myöhempänä tekstissä. Toisenlainen hakusanojen muotoilu tai teksti-ikkunan rajaus olisi todennäköisesti parantanut tulosten tarkkuutta, mutta toisaalta vaarana olisi ollut, että saanti olisi heikentynyt huomattavasti.

Parhaimmat hakutulokset molemmilla menetelmillä tulivat etsittäessä kirjeitä pelkän vuosiluvun avulla, jolloin sekä tarkkuus että saanti olivat varsin hyvät, pääsääntöisesti suorastaan erinomaiset. Kun hakua alettiin rajata tarkemmaksi kuukauden avulla, tulokset heikkenivät hakutermin ”3” vuoksi erityisesti tarkkuuden suhteen, mutta myös saanti huononi eli kaikki relevantit kirjeet eivät enää löytyneet. Tarkalla päivämäärällä haettaessa hälyn eli epärelevanttien osumien määrä lisääntyi huomattavasti tekstissä runsaasti esiintyvän hakusanan ”joulu” vuoksi, joskin pääsääntöisesti myös relevantit kirjeet löytyivät. Saanti oli siis kohtalaisen hyvä, mutta tarkkuus heikkeni molemmilla menetelmillä hyvin alhaiseksi.

Käytetyillä hakuaiheilla tulokset olivat sinänsä kohtalaiset, mutta jos vuosiluku, kuukausi tai tarkka päiväys olisi muutettu toisenlaisiksi, olisivat tulokset saattaneet olla paljon huonommat. Vuodella 1941 haettaessa joukkoon olisi tullut luultavasti kaikki kokoelman 1 kirjeet sen alkuperäisen SAK-kokoelmanumeron vuoksi, jolloin hälyä olisi tullut huomattavasti myös vuosipoimintaan, ja toisaalta koska joiltakin vuosilta kirjeitä oli tutkimusaineistossa niin vähän, ei hakua näiden vuosien perusteella haluttu suorittaa, jotta osumia tulisi tuloksiin enemmän. Maaliskuun osalta kuukauden ilmaisussa oli varsin vähän esimerkiksi OCR-virheitä, kun taas manuaalista läpikäyntiä tehdessä huomio kiinnittyi siihen, että joissain osakokoelmissa erityisesti helmi- ja marraskuun kohdalla oli runsaasti OCR-virheiksi tulkittavia muunnoksia kuukauden ilmauksessa, sillä ne oli saatettu merkitä kirjaimin ikään kuin roomalaisella numerolla. Tällaisten kuukausien löytyminen tarkastelluilla menetelmillä olisi ollut käytännössä mahdotonta, ellei OCR-virhettä olisi huomioitu hakutermissä jollain tavalla, ja ainakaan FiNER ei olisi tällaista päiväystä tunnistanut. Joulu tuntui parhaimmalta tarkalta päiväykseltä paitsi aiemmin mainituista syistä, myös siksi, että sillä on juhlana kiinteä vuodesta riippumaton päivämäärä toisin kuin vaikkapa juhannuksella tai äitienpäivällä, joiden päivämäärä vaihtelee vuosittain. Tarkastelun kohteeksi valitut ajanjaksot päätettiin siis ensisijaisesti sen perusteella, että osumia

aineistosta tulisi mahdollisimman paljon, ja näiden avulla hakuohjelman toteuttaminen oli yksinkertaisempaa, joskin testaus- ja kehittelyvaiheessa kokeiltiin myös muita hakutermejä ohjelmallisten virheiden minimoimiseksi.

Kahden menetelmän vertailussa aineiston ominaispiirteet huomioiden kehitelty oma hakuohjelma suoriutui kautta linjan eri paremmin kuin FiNERin päiväystunnistein merkattu aineisto, sillä haku kohdistui huomattavasti suurempaan kirjemäärään. Kumpikaan menetelmistä ei kuitenkaan suoriutunut täydellisesti, kun tuloksia verrattiin siihen, mitä manuaalisen läpikäynnin perusteella olisi pitänyt löytyä. Luonnollinen kieli, päiväysten yksilölliset merkintätavat, entiteettien tunnistuminen sekä haun kohdistaminen vaikuttavat siis siihen, millaisia tuloksia aineistosta on mahdollista saada. Kaiken kaikkiaan saadut tulokset ovat tarkkuuden ja saannin arvioinnin valossa kohtalaisen hyvät, mutta paikoitellen parantamisen varaa on myös varsin runsaasti.

Vaikka tätä tutkimusta varten koottiinkin kaikista digitoiduista sota-ajan kirjeistä pienempi testikokoelma, voi tutkimustulokset kuitenkin yleistää varsin hyvin koskemaan koko aineistoa, sillä erilaiset päiväyksen merkintätavat kirjeissä ovat loppujen lopuksi varsin rajalliset, vaikka yksilöllistä vaihtelua luonnollisesti on jonkin verran. Päiväyksen kirjoittaminen kirjeeseen pelkästään numeraalisesti pistein eroteltuna ja vuosiluvun kirjoittaminen kokonaisuudessaan (esimerkiksi 1940) parantaisivat päiväyksen löytymistä, sillä tällöin epämääräisempiä hakutermejä (kuten ”40”) ei tarvitsisi käyttää ja epärelevanttien osumien mukaan tulo vähenisi. Lisäksi tällä tavoin myös FiNER osaisi tunnistaa päiväyksen paremmin entiteetiksi, mikä helpottaisi ajanjaksoltaan kiinnostavien kirjeiden löytymistä, kun haun voisi kohdistaa myös entiteettitunnisteeseen. Nimettyjen entiteettien tai muun tekstinsisäisen annotoinnin hyöty on siinä, että tällä tavoin päiväys olisi mahdollista saada paremmin kiinni, mutta toistaiseksi nykyiset menetelmät eivät tämän tutkimuksen valossa toimi vielä parhaalla mahdollisella tavalla. Vaikka päiväys kaikissa kirjeissä olisikin merkitty mahdollisimman yhdenmukaisesti ja selkeästi, eivät kaikki päiväykset ja kirjeet silti todennäköisesti löytyisi automattisin menetelmin johtuen muun muassa OCR-virheistä. Tutkimuksen aikana havaittiin, että varsinkin kuukausien ja vuosilukujenkin kohdalla erilaisia OCR-tunnistusvirheitä oli jonkin verran. Toisaalta osa päiväyksistä oli kirjoitettu juhlapyhän tai nimipäivän perusteella kirjaimin eli tarkka numeraalinen päivämäärä jäi tällöin puuttumaan, ja ainakaan nykyisellään FiNER ei tunnistanut näitä päiväysentiteeteiksi.

Itse tehtyyn manuaaliseen annotointiin kerättiin kaikista kirjeistä päiväys yhdenmukaiseen muotoon, ja tulkintaa tarkan päiväyksen selvittämiseksi pyrittiin tekemään mahdollisuuksien mukaan. Aivan kaikkiin kirjeisiin ei tarkkaa päiväystä voinut silti merkitä, sillä vuosiluku tai koko päiväys saattoi puuttua kokonaan, eikä kaikkien epäselvempien tilanteiden kohdalla ollut mahdollista tehdä varmaa tulkintaa pelkän digitaalisessa muodossa olevan aineiston pohjalta. Manuaalisesti läpikäyty ja tulkittu päiväys toimi tässä tutkimuksessa lähinnä tulosten arvioinnin vertailupohjana, sillä kyseinen tieto jäi automaattisten menetelmien ulkopuolelle tarkoituksella, jotta oli mahdollista selvittää, miten päiväys löytyi varsinaisesta kirjeen tekstisisällöstä. Metadataksi kerätty yhdenmukaisessa muodossa oleva päiväys voisi kuitenkin toimia yleisesti ottaen tekstinlouhinnan ja tiedonhaun apuvälineenä, mikäli tulkinnan päiväyksen oikeellisuudesta olisi tehnyt useampi kuin yksi henkilö, jolloin sitä voitaisiin pitää luotettavampana. Sota-ajan kirjeisiin olisikin digitaalisessa arkistossa hyvä liittää varmennettu päiväystieto mahdollisuuksien mukaan, jotta kirjeitä olisi mahdollista löytää tarvittaessa suoraan sen avulla.

6.2 Ajatuksia tulevaisuutta ajatellen

Digitoitujen historiallisten aineistojen tiedonhaun menetelmiä kehiteltäessä olisi tärkeää jatkuvasti ottaa huomioon se, millaista aineisto on luonteeltaan. Tutkimusaineistona olleet sota-ajan kirjeet poikkeavat virallisemmista kirjeistä siinä mielessä, että päiväysten suhteen niissä on luultavasti käytetty useammanlaisia, enemmän henkilökohtaiseen mieltymykseen ja tottumukseen perustuvia merkintätapoja, kun taas virallisemmissa kirjeissä ja muissa dokumenteissa käytännöt ovat todennäköisesti olleet yhdenmukaisemmat. Koska variaatiota sota-ajan kirjeaineistossa on enemmän, vaikuttaa se myös siihen, miten päiväys on tunnistettavissa ja löydettävissä. Tämän tutkimuksen puitteissa havaittiin myös OCR-laadun vaikutus tuloksiin, vaikka päiväysten kohdalla OCR-virheitä olikin huomattavasti vähemmän kuin muualla kirjetekstissä. Historiallisten aineistojen digitoinnissa onkin tärkeää pyrkiä mahdollisimman laadukkaaseen OCR-tunnistustulokseen, jotta aineisto olisi käytettävää, minkä myös Kettunen ja muut (2016 ja 2017) huomasivat omissa tutkimuksissaan. Ihmisen tekemä jälkikorjaus takaisi todennäköisesti parhaimman lopputuloksen etenkin, jos apuna käytettäisiin alkuperäisiä paperisia aineistoja, mutta tämä ei aineistomäärien ja inhimillisten resurssien puitteissa ole kovinkaan varteenotettava vaihtoehto, jolloin automaattisia menetelmiä tulisi pyrkiä parantelemaan muilla keinoin.

Digitaalisiin aineistoihin on mahdollista soveltaa monenlaisia tietoteknisiä menetelmiä, jotka helpottavat esimerkiksi tutkijoiden työtä aineistojen käsittelyssä ja tietojen etsimisessä. Tekstinlouhinnalla, nimettyjen entiteettien hyödyntämisellä ja annotoinnilla voidaan digitoitujen aineistojen käytettävyyttä parantaa monin tavoin, mutta toistaiseksi näiden menetelmien avulla ei vielä päästä parhaaseen mahdolliseen lopputulokseen, sillä menetelmät vaativat vielä kehittelyä etenkin suomen kielen käsittelyn osalta. Nimettyjen entiteettien tunnistuksella ja erityisesti tekstinsisäisellä muulla annotoinnilla voisi aineistoon merkitä huomionarvoisia kohtia, kuten päiväyksiä tai henkilöiden sekä paikkojen nimiä. Päiväysten automaattinen annotointi joko nimettyjen entiteettien tunnistuksen avulla tai jollain muulla menetelmällä mahdollistaisi paitsi päiväysten löytymisen itsessään, myös temporaalisen annotoinnin, jonka avulla kirjeistäkin olisi mahdollista koota erilaisia aikajanoja. Kirjeissä mainitaan luultavasti eniten henkilökohtaiseen elämään liittyvistä menneistä ja tulevista tapahtumista, mutta sensuurista huolimatta niistä olisi luultavasti mahdollista eritellä myös sotaan liittyviä tapahtumia, jotka voisi sitten liittää osaksi laajempaa sotavaiheiden aikajanaa. Kirjeiden päiväys kelpaisi tähän tarkoitukseen luonnollisesti myös, mutta kirjeiden sisäinen temporaalinen annotointi saattaisi tarjota myös monia muita tutkimuksellisia mahdollisuuksia, joihin ei tämän tutkielman puitteissa nyt keskitytty.

Tekstinlouhinta osana tiedonhakua on varsin yleinen menetelmä, ja sota-ajan kirjeiden osalta olisi ollut myös mahdollista laatia samankaltaisia visuaalisia karttoja ja verkostoja kuin Elo & Kleemola (2016) omassa tutkimuksessaan tekivät. Samana ajankohtana eri henkilöiden kirjoittamat kirjeet kertovat kukin omasta näkökulmastaan koettuja tapahtumia, jolloin tietyn sodan hetken perusteella saisi tuotua esille lukuisia henkilökohtaisia kokemuksia, joiden vertailu voisi olla tutkimuksellisesti kiinnostavaa. Sota-ajan kirjeet tarjoavat parhaimmat tutkimusmahdollisuudet muiden alojen kuin informaatiotutkimuksen parissa, sillä esimerkiksi historia- ja kielitieteiden suhteen kirjeiden sisällön tarkempi analyysi on tarkoituksenmukaisempaa. Informaatiotutkimuksen ja tietojenkäsittelytieteiden avulla muille aloille pystyttäisiin kuitenkin tarjoamaan erilaisia menetelmiä ja apuvälineitä, joiden avulla erityisesti digitaalisten aineistojen hyödyntäminen helpottuu ja nopeutuu.

Vaikka monenlaisia tietoteknisiä menetelmiä onkin jo olemassa ja niitä on kehitelty myös suomenkielistä aineistoa silmällä pitäen, eivät ne kuitenkaan kykene vielä tarjoamaan aukottomia käyttömahdollisuuksia historialliselle digitaalisessa muodossa olevalle

aineistolle, sillä aineisto itsessään saattaa olla optisen tunnistuksen vuoksi virheellistä. Kirjeiden yksilöllinen kirjoitustyyli vaikuttaa myös tilanteeseen, sillä sota-ajan kirjekokoelma sisältää epävirallisempaa materiaalia, vaikkakin kirjoitustapojen erot hämärtyvät eniten päiväysten kohdalta, sillä niiden merkintätavat ovat kuitenkin olleet kautta aikain kohtalaisen vakiintuneet. Tietoteknisiä menetelmiä tulisi kehittää aina ensisijaisesti kohdeaineisto ja käyttäjät huomioiden, sillä muulla tavoin parhaaseen mahdolliseen lopputulokseen ei voi päästä. Monitieteisyyttä ja tieteidenvälisyyttä hyödynnetään jo nyt hyvin erilaisissa tutkimushankkeissa sekä laajemmin digitaalisten ihmistieteiden parissa, mutta yhteistyötä eri alojen välillä tulisi pyrkiä jatkuvasti vain parantamaan ja lisäämään. Digitoinnin ansiosta historialliset aineistot ovat aiempaa paremmin saavutettavissa ja jotta kaikkien olisi niitä mahdollisuus myös tarkastella ja hyödyntää parhaaksi katsomallaan tavalla, tulee sitä varten kehitellä ja tarjota aina vain parempia menetelmiä, jotta arkistojen ainutlaatuinen aineisto ei hukkuisi digitaalisuuden syövereihin.

LÄHTEET

Aineistolähteet:

Kansanperinteen arkisto. Sota-ajan kirjekokoelma (SAK). STASKO-projektin digitoimat kirjeet. SAK/41, SAK/54, SAK/27, SAK/64, SAK/89.

Tutkimuskirjallisuus:

Balog, Krisztian 2018. *Entity-Oriented Search*. The Information Retrieval Series, Volume 39. Springer Open. https://doi.org/10.1007/978-3-319-93935-3

Bird, Steven; Klein, Ewan & Loper, Edward 2009. *Natural Language Processing with Python*. Beijing, Cambridge, Farnham, Köln, Sebastopol, Taipei, Tokyo: O'Reilly.

Davies, John; Duke, Alistair & Kiryakov, Atanas 2009. "Semantic Search". Teoksessa Ayşe Göker & John Davies (toim.) *Information Retrieval. Searching in the 21st Century*. Chichester: Wiley. S. 179–210.

Elo, Kimmo (toim.) 2016. *Digitaalinen humanismi ja historiatieteet*. Historia Mirabilis 12. Turku: Turun Historiallinen Yhdistys.

Elo, Kimmo 2016. "Digitaalisen historiantutkimuksen kenttää louhimassa". Teoksessa Kimmo Elo (toim.) *Digitaalinen humanismi ja historiatieteet*. Historia Mirabilis 12. Turku: Turun Historiallinen Yhdistys. S. 11–35.

Elo, Kimmo & Kleemola, Olli 2016. "SA-kuva-arkistoa louhimassa. Digitaaliset tutkimusmenetelmät valokuvatutkimuksen tukena". Teoksessa Kimmo Elo (toim.) *Digitaalinen humanismi ja historiatieteet*. Historia Mirabilis 12. Turku: Turun Historiallinen Yhdistys. S. 151–190.

FIN-CLARIN. Suomalaiset kielivarat yhteiskäyttöön. https://kitwiki.csc.fi/twiki/bin/view/FinCLARIN/KielipankkiEtusivu [viitattu 8.3.2019]

Finnish Tagtools. https://korp.csc.fi/download/finnish-tagtools/v1.3/ [käytetty 7.3.2019]

Foley, John & Allan, James 2015. "Retrieving Time from Scanned Books". European Conference on Information Retrieval (ECIR) 2015, LNCS 0922. S. 221–232.

Foulonneau, Muriel & Riley, Jenn 2008. *Metadata for Digital Resources. Implementation, Systems Design and Interoperability*. Oxford: Chandos Publishing.

Joho, Hideo; Jatowt, Adam; Blanco, Roi; Yu, Haitao & Yamamoto, Shuhei 2016. "Building Test Collections for Evaluating Temporal IR". SIGIR '16, July 17–21, 2016, Pisa, Italy. S. 677–680. doi: http://dx.doi.org/10.1145/2911451.2914673

Järvelin, Kalervo & Sormunen, Eero 2010. "Tiedon tallennus ja haku". Teoksessa Sami Serola (toim.) *Ote informaatiosta: johdatus informaatiotutkimukseen ja interaktiiviseen median*. Helsinki: BTJ Kustannus. S. 155–210.

Kettunen, Kimmo & Löfberg, Laura 2017. "Tagging Named Entities in 19th Century and Modern Finnish Newspaper Material with a Finnish Semantic Tagger". *Proceedings of the 21st Nordic Conference of Computational Linguistics*. Gothenburg, Sweden, 23–24 May 2017. Linköping University Electronic Press. S. 29–36.

Kettunen, K. & Ruokolainen, T. 2017. "Names, Right or Wrong: Named Entities in an OCRed Historical Finnish Newspaper Collection". *Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage*. Göttingen, Germany June 01–02 2017. S. 181–186. doi:10.1145/3078081.3078084

Kettunen, Kimmo; Pääkkönen, Tuula & Koistinen Mika 2016a. "Kansalliskirjaston digitoitu historiallinen lehtiaineisto 1771–1910: sanatason laatu, kokoelmien käyttö ja laadun parantaminen". *Informaatiotutkimus* 35 (3), 2016. S. 3–14.

Kettunen, Kimmo; Mäkelä, Eetu; Ruokolainen, Teemu; Kuokkala, Juha & Löfberg, Laura 2016b. "Old Content and Modern Tools – Searching Named Entities in a Finnish OCRed Historical Newspaper Collection 1771–1910." https://arxiv.org/ftp/arxiv/papers/1611/1611.02839.pdf [viitattu 8.3.2019]

Kielipankki. https://www.kielipankki.fi/ [viitattu 8.3.2019]

Mani, Inderjeet; Pustejovsky, J. & Gaizauskas, Robert 2005. *The Language of Time. A Reader*. New York: Oxford University Press.

Mustanoja, Liisa (toim.) 2017a. *Arjen sirpaleita ja suuria tunteita. Kirjeet sodan sanoittajina ja ihmissuhteiden ylläpitäjinä 1939–1944*. Tampere: Tampereen yliopisto.

Mustanoja, Liisa 2017b. “Johdanto: Kirjeitä sodasta ja rakkaudesta”. Teoksessa Liisa Mustanoja (toim.) *Arjen sirpaleita ja suuria tunteita. Kirjeet sodan sanoittajina ja ihmissuhteiden ylläpitäjinä 1939–1944*. Tampere: Tampereen yliopisto. S. 8–21.

Nadeau, David & Sekine, Satoshi 2007. ”A survey of named entity recognition and classification”. *Linguisticae Investigationes*. Volume 30, Issue 1, January 2007. Special Issue. Named Entities: Recognition, classification and use. S. 3–26.

Nagao, Katashi 2003. *Digital Content Annotation and Transcoding*. Boston, MA: Artech House.

Rasila, Viljo 1984. ”Miksi kirjeitä kerättiin?” Teoksessa M. K. Suojanen (toim.) *Sota-ajan kirjeet*. Tampereen yliopiston kansaperinteen laitoksen moniste 6. Tampere: Tampereen yliopisto. S. 23–26.

Setzer, Andrea; Gaizauskas, Robert & Hepple, Mark 2005. ”The Role of Inference in the Temporal Annotation and Analysis of Text”. *Language Resources and Evaluation*. May 2005, Volume 39, Issue 2–3. S. 243–265.

Silfverberg, Miikka 2015. ”Reverse Engineering a Rule-Based Finnish Named Entity Recognizer”. Named Entity Recognition in Digital Humanities Workshop, Helsinki, kesäkuu 2015. https://kitwiki.csc.fi/twiki/pub/FinCLARIN/KielipankkiEventNERWorkshop2015/Silfverberg_presentation.pdf [viitattu 8.3.2019]

STASKO-hankkeen verkkosivut. *Sota, suuret tietokannat ja datavisualisointi*. https://research.uta.fi/stasko/ [viitattu 7.3.2019]

Suojanen, M. K. (toim.) 1984. *Sota-ajan kirjeet*. Tampereen yliopiston kansaperinteen laitoksen moniste 6. Tampere: Tampereen yliopisto.

Tikka, Marko; Taskinen, Ilari & Nevala-Nurmi, Seija-Leena (toim.) 2015. *Kirjeitä sodasta. Kirjoittamisen tavat ja merkitykset kriisiaikoina*. Tampere: Postimuseo, Tampereen Historiallinen Seura.

Verhagen, Marc; Gaizauskas Robert; Schilder, Frank; Hepple, Mark, Moszkowicz, Jessica & Pustejovsky, James 2009. “The TempEval Challenge: Identifying Temporal Relations in Text”. *Language Resources and Evaluation*. June 2009, Volume 43, Issue 2. S. 161–179.

Voorhees, Ellen M. 2008. ”On test collections for adaptive information retrieval”. *Information Proceedings and Management* 44 (2008). S. 1879–1885.

Yadav, Anjali; Kumar-Sharma, Dilip & Pradhan, Rahul 2015. ”Implicit Queries based Temporal Information Retrieval using Temporal Taggers”. 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) doi: 10.1109/ICRITO.2015.7359271 [viitattu 8.3.2019]

LIITE 1: Python-ohjelma “find_year.py”

```
# This file: "find_year.py"
# Usage: python find_year.py
# Comment mark: #. Comments and the program are written in English, because
# Finnish alphabets might cause character encoding problems.
#
# Created by: Selja Vanamo (selja.vanamo@tuni.fi) at March 2019
#
# The purpose of this program is to find letters from certain exact year
# and extract them to a new file.

import nltk # Natural Language Toolkit
import os, sys # os for operating system dependent interface, sys for system
               # spesific parameters
import os.path # for checking existing paths
import re # for regular expressions

# This is the path to the directory, x-marks indicate your basic user account
# number.
path = "/home/stud/sis/xxxxxxx"

# Change the numbers or strings of the search terms according to your search
# interest.
year1 = "1940"
year2 = "40"

# This function returns all the filenames from the given directory as a list.
def read_all_filenames(path):
    all_files = [] # empty list
    dirs = os.listdir(path) # enter the path of the file, return the
                            # dictionary name of path name
    for file in dirs: # make a for loop to extract file names one by one
        ufile = file.decode('utf-8') # decode to Unicode
        all_files.append(ufile) # add a single element to the end of the list
    return all_files

# This function returns a selected subset of filenames as a list.
def select_filenames(all_files):
    selected_files = [] # empty list
    for filename in all_files: # make a for loop to extract file names one by
                               # one
        # Check that filename starts with certain kind of string and file type
        # is ".txt".
        if filename.startswith("kokoelma_") and filename.endswith(".txt"):
            selected_files.append(filename) # add a single element to the end
# of the list
    return sorted(selected_files) # return alphabetically sorted list

# This function opens and reads file's contents and returns it as Python's
# Unicode string.
def get_file_contents(filename):
    orig_file = open(filename, "r") # open the original file for reading
    filetext = orig_file.read() # read the contents of the file
    ufiletext = filetext.decode('utf-8') # decode to Unicode
    orig_file.close() # close the file
    return ufiletext

# This function creates a copy from the original filename.
def modify_filename(filename):
    new_name = "year_" + filename # create a new filename from the original
                                  # name
    modified_filename = new_name.decode('utf-8') # decode to Unicode
    return modified_filename

# First the function checks that the given filename doesn't exist
# already (if so, an error message is given and program exits).
```

```
# Then the function writes a copy of the file content to the newly
# named file.
def write_to_file(modified_filename, result_letters):
    if os.path.exists(modified_filename): # check if the filename already
                                          # exists to avoid overwriting
        print "File already exists. Please use another name."
        sys.exit("Exiting.")
    else:
        new_filetext = result_letters.encode('utf-8') # encode to UTF-8
        new_file = open(modified_filename, "w") # open the new file for
                                                # writing
        new_file.write(new_filetext) # write to file
        new_file.close() # close the file

# This function searches the letter collection and filters relevant letters
# based on search term.
def search_and_filter(letter_text, search_term1, search_term2):
    results = "" # empty string
    letters = letter_text.split("<letter>") # split letters apart from each
                                            # other
    for letter in letters: # check letters one by one in one collection at a
                           # time
        tag1 = letter.find("<text>") # check only the actual letter text
                                     # starting with this tag...
        tag2 = letter.find("</text>", tag1) # ... and ending with this tag
        text = letter[tag1+len("<text>"):tag2]
        # check if any of the given years are found in the letter text
        if ((text.find(search_term1, 0, 200) > 0) or (text.find(search_term2,
0, 200) > 0)):
            results = results + "\n" + "<letter>" + letter # collect results
                                                          # of the relevant letters
    return results
#
# Begin main
#
all_files = read_all_filenames(path) # in Unicode
selected_files = select_filenames(all_files) # in Unicode
for filename in selected_files: # in Unicode
    utext = get_file_contents(filename) # in Unicode
    year_letters = search_and_filter(utext, year1, year2) # in Unicode
    copy_name = modify_filename(filename) # in Unicode
    write_to_file(copy_name, year_letters) # filenames written in UTF-8
```

LIITE 2: Python-ohjelma “find_month.py”

```
# This file: "find_month.py"
# Usage: python find_month.py
# Comment mark: #. Comments and the program are written in English, because
# Finnish alphabets might cause character encoding problems.
#
# Created by: Selja Vanamo (selja.vanamo@tuni.fi) at March 2019
#
# The purpose of this program is to find letters from certain exact year and
# month and extract them to a new file.

import nltk # Natural Language Toolkit
import os, sys # os for operating system dependent interface, sys for system
               # spesific parameters
import os.path # for checking existing paths
import re # for regular expressions

# This is the path to the directory, x-marks indicate your basic user account
# number.
path = "/home/stud/sis/xxxxxxx"

# Change the numbers or strings of the search terms according to your search
# interest.
year1 = "1940"
year2 = "40"
month1 = "3"
month2 = "maalisk"

# This function returns all the filenames from the given directory as a list.
def read_all_filenames(path):
    all_files = [] # empty list
    dirs = os.listdir(path) # enter the path of the file, return the
                            # dictionary name of path name
    for file in dirs: # make a for loop to extract file names one by one
        ufile = file.decode('utf-8') # decode to Unicode
        all_files.append(ufile) # add a single element to the end of the list
    return all_files

# This function returns a selected subset of filenames as a list.
def select_filenames(all_files):
    selected_files = [] # empty list
    for filename in all_files: # make a for loop to extract file names one by
                               # one
        # Check that filename starts with certain kind of string and file type
        # is ".txt".
        if filename.startswith("kokoelma_") and filename.endswith(".txt"):
            selected_files.append(filename) # add a single element to the end
                                            # of the list
    return sorted(selected_files) # return alphabetically sorted list

# This function opens and reads file's contents and returns it as Python's
# Unicode string.
def get_file_contents(filename):
    orig_file = open(filename, "r") # open the original file for reading
    filetext = orig_file.read() # read the contents of the file
    ufiletext = filetext.decode('utf-8') # decode to Unicode
    orig_file.close() # close the file
    return ufiletext

# This function creates a copy from the original filename.
def modify_filename(filename):
    new_name = "monthyear_" + filename # create a new filename from the
                                       # original name
    modified_filename = new_name.decode('utf-8') # decode to Unicode
```

```
    return modified_filename

# First the function checks that the given filename doesn't exist already
# (if so, an error message is given and program exits).
# Then the function writes a copy of the file content to the newly named file.
def write_to_file(modified_filename, result_letters):
    if os.path.exists(modified_filename): # check if the filename already
                                          # exists to avoid overwriting
        print "File already exists. Please use another name."
        sys.exit("Exiting.")
    else:
        new_filetext = result_letters.encode('utf-8') # encode to UTF-8
        new_file = open(modified_filename, "w") # open the new file for
                                                # writing
        new_file.write(new_filetext) # write to file
        new_file.close() # close the file

# This function searches the letter collection and filters relevant letters
# based on search term.
def search_and_filter(letter_text, search_term1, search_term2):
    results = "" # empty string
    letters = letter_text.split("<letter>") # split letters apart from each
                                            # other
    for letter in letters: # check letters one by one in one collection at a
                           # time
        tag1 = letter.find("<text>") # check only the actual letter text
                                     # starting with this tag...
        tag2 = letter.find("</text>", tag1) # ... and ending with this tag
        text = letter[tag1+len("<text>"):tag2]
        # check if any of the given years/months are found in the letter text
        if ((text.find(search_term1, 0, 200) > 0) or (text.find(search_term2,
0, 200) > 0)):
            results = results + "\n" + "<letter>" + letter # collect results
                                                          # of the relevant letters
    return results
#
# Begin main
#
all_files = read_all_filenames(path) # in Unicode
selected_files = select_filenames(all_files) # in Unicode
for filename in selected_files: # in Unicode
    utext = get_file_contents(filename) # in Unicode
    year_letters = search_and_filter(utext, year1, year2) # in Unicode
    month_letters = search_and_filter(year_letters, month1, month2) # in
                                                                    # Unicode
    copy_name = modify_filename(filename) # in Unicode
    write_to_file(copy_name, month_letters) # filenames written in UTF-8
```

LIITE 3: Python-ohjelma “find_date.py”

```
# This file: "find_date.py"
# Usage: python find_date.py
# Comment mark: #. Comments and the program are written in English, because
# Finnish alphabets might cause character encoding problems.
#
# Created by: Selja Vanamo (selja.vanamo@tuni.fi) at March 2019
#
# The purpose of this program is to find letters from certain exact date
# and extract them to a new file.

import nltk # Natural Language Toolkit
import os, sys # os for operating system dependent interface, sys for system
               # spesific parameters
import os.path # for checking existing paths
import re # for regular expressions

# This is the path to the directory, x-marks indicate your basic user account
# number.
path = "/home/stud/sis/xxxxxxx"

# Change the numbers or strings of the search terms according to your search
# interest.
date1 = "2[45][.,/ ]12" # accept any character within []
date2 = "[jJ]oulu[^k]" # accept word starting with "joulu" with the letter "j"
                       # or "J" but not for example "joulukuu"

# This function returns all the filenames from the given directory as a list.
def read_all_filenames(path):
    all_files = [] # empty list
    dirs = os.listdir(path) # enter the path of the file, return the
                            # dictionary name of path name
    for file in dirs: # make a for loop to extract file names one by one
        ufile = file.decode('utf-8') # decode to Unicode
        all_files.append(ufile) # add a single element to the end of the list
    return all_files

# This function returns a selected subset of filenames as a list.
def select_filenames(all_files):
    selected_files = [] # empty list
    for filename in all_files: # make a for loop to extract file names one by
                               # one
        # Check that filename starts with certain kind of string and file type
        # is ".txt".
        if filename.startswith("kokoelma_") and filename.endswith(".txt"):
            selected_files.append(filename) # add a single element to the end
                                            # of the list
    return sorted(selected_files) # return alphabetically sorted list

# This function opens and reads file's contents and returns it as Python's
# Unicode string.
def get_file_contents(filename):
    orig_file = open(filename, "r") # open the original file for reading
    filetext = orig_file.read() # read the contents of the file
    ufiletext = filetext.decode('utf-8') # decode to Unicode
    orig_file.close() # close the file
    return ufiletext

# This function creates a copy from the original filename.
def modify_filename(filename):
    new_name = "date_" + filename # create a new filename from the original
                                  # name
    modified_filename = new_name.decode('utf-8') # decode to Unicode
    return modified_filename
```

```
# First the function checks that the given filename doesn't exist already
# (if so, an error message is given and program exits).
# Then the function writes a copy of the file content to the newly named file.
def write_to_file(modified_filename, result_letters):
    if os.path.exists(modified_filename): # check if the filename already
                                          # exists to avoid overwriting
        print "File already exists. Please use another name."
        sys.exit("Exiting.")
    else:
        new_filetext = result_letters.encode('utf-8') # encode to UTF-8
        new_file = open(modified_filename, "w") # open the new file for
                                                # writing
        new_file.write(new_filetext) # write to file
        new_file.close() # close the file

# This function searches the letter collection and filters relevant letters
# based on search term.
def search_and_filter(letter_text, search_term1, search_term2):
    results = "" # empty string
    letters = letter_text.split("<letter>") # split letters apart from each
                                            # other
    for letter in letters: # check letters one by one in one collection at a
                           # time
        tag1 = letter.find("<text>") # check only the actual letter text
                                     # starting with this tag...
        tag2 = letter.find("</text>", tag1) # ... and ending with this tag
        text = letter[tag1+len("<text>"):tag2]
        # check if any of the given dates are found in the letter text
        if ((re.search(date1, text)) or (re.search(date2, text))):
            results = results + "\n" + "<letter>" + letter # collect results
                                                  # of the relevant letters
    return results
#
# Begin main
#
all_files = read_all_filenames(path) # in Unicode
selected_files = select_filenames(all_files) # in Unicode
for filename in selected_files: # in Unicode
    utext = get_file_contents(filename) # in Unicode
    date_letters = search_and_filter(utext, date1, date2)
    copy_name = modify_filename(filename) # in Unicode
    write_to_file(copy_name, date_letters) # filenames written in UTF-8
```

LIITE 4: Python-ohjelma “finer_find_year.py”

```
# This file: "finer_find_year.py"
# Usage: python finer_find_year.py
# Comment mark: #. Comments and the program are written in English, because
# Finnish alphabets might cause character encoding problems.
#
# Created by: Selja Vanamo (selja.vanamo@tuni.fi) at March 2019
#
# The purpose of this program is to find letters from certain exact year
# and extract them to a new file. First the files have gone thorough FiNER's
# named entity recognition. For the search only date entity tagged letters
# (some form of "<TimexTmeDat>") have been excepted.

import nltk # Natural Language Toolkit
import os, sys # os for operating system dependent interface, sys for system
               # spesific parameters
import os.path # for checking existing paths
import re # for regular expressions

# This is the path to the directory, x-marks indicate your basic user account
# number.
path = "/home/stud/sis/xxxxxxx"

# Change the numbers or strings of the search terms according to your search
# interest.
year1 = "1940"
year2 = "40"

# This function returns all the filenames from the given directory as a list.
def read_all_filenames(path):
    all_files = [] # empty list
    dirs = os.listdir(path) # enter the path of the file, return the
                            # dictionary name of path name
    for file in dirs: # make a for loop to extract file names one by one
        ufile = file.decode('utf-8') # decode to Unicode
        all_files.append(ufile) # add a single element to the end of the list
    return all_files

# This function returns a selected subset of filenames as a list.
def select_filenames(all_files):
    selected_files = [] # empty list
    for filename in all_files: # make a for loop to extract file names one by
                               # one
        # Check that filename starts with certain kind of string and file type
        # is ".txt".
        if filename.startswith("finer_kokoelma_") and file-
name.endswith(".txt"):
            selected_files.append(filename) # add a single element to the end
                                            # of the list
    return sorted(selected_files) # return alphabetically sorted list

# This function opens and reads file's contents and returns it as Python's
# Unicode string.
def get_file_contents(filename):
    orig_file = open(filename, "r") # open the original file for reading
    filetext = orig_file.read() # read the contents of the file
    ufiletext = filetext.decode('utf-8') # decode to Unicode
    orig_file.close() # close the file
    return ufiletext

# This function creates a copy from the original filename.
def modify_filename(filename):
    new_name = "year_" + filename # create a new filename from the original
                                  # name
    modified_filename = new_name.decode('utf-8') # decode to Unicode
```

```
    return modified_filename

# First the function checks that the given filename doesn't exist already
# (if so, an error message is given and program exits).
# Then the function writes a copy of the file content to the newly named file.
def write_to_file(modified_filename, result_letters):
    if os.path.exists(modified_filename): # check if the filename already
                                          # exists to avoid overwriting
        print "File already exists. Please use another name."
        sys.exit("Exiting.")
    else:
        new_filetext = result_letters.encode('utf-8') # encode to UTF-8
        new_file = open(modified_filename, "w") # open the new file for
                                                # writing
        new_file.write(new_filetext) # write to file
        new_file.close() # close the file

# This function searches the letter collection and filters relevant letters
# based on search term.
def search_and_filter(letter_text, search_term1, search_term2):
    results = "" # empty string
    letters = letter_text.split("<letter>") # split letters apart from each
                                            # other
    for letter in letters: # check letters one by one in one collection at a
                           # time
        tag1 = letter.find("<text>") # check only the actual letter text
                                     # starting with this tag...
        tag2 = letter.find("</text>", tag1) # ... and ending with this tag
        text = letter[tag1+len("<text>"):tag2]
        # accept only the letters with some form of tag "<TimexTmeDat>"
        if "TimexTmeDat" in text:
            # check if any of the given years are found in the letter text
            if ((text.find(search_term1, 0, 200) > 0) or
(text.find(search_term2, 0, 200) > 0)):
                results = results + "\n" + "<letter>" + letter # collect
                                                # results of the relevant letters
    return results
#
# Begin main
#
all_files = read_all_filenames(path) # in Unicode
selected_files = select_filenames(all_files) # in Unicode
for filename in selected_files: # in Unicode
    utext = get_file_contents(filename) # in Unicode
    year_letters = search_and_filter(utext, year1, year2) # in Unicode
    copy_name = modify_filename(filename) # in Unicode
    write_to_file(copy_name, year_letters) # filenames written in UTF-8
```

LIITE 5: Python-ohjelma “finer_find_month.py”

```
# This file: "finer_find_month.py"
# Usage: python finer_find_month.py
# Comment mark: #. Comments and the program are written in English, because
# Finnish alphabets might cause character encoding problems.
#
# Created by: Selja Vanamo (selja.vanamo@tuni.fi) at March 2019
#
# The purpose of this program is to find letters from certain exact year and
# month and extract them to a new file. First the files have gone thorough
# FiNER's named entity recognition. For the search only date entity tagged
# letters (some form of "<TimexTmeDat>") have been excepted.

import nltk # Natural Language Toolkit
import os, sys # os for operating system dependent interface, sys for system
               # spesific parameters
import os.paths # for checking existing paths
import re # for regular expressions

# This is the path to the directory, x-marks indicate your basic user account
# number.
path = "/home/stud/sis/xxxxxxx"

# Change the numbers or strings of the search terms according to your search
# interest.
year1 = "1940"
year2 = "40"
month1 = "3"
month2 = "maalisk"

# This function returns all the filenames from the given directory as a list.
def read_all_filenames(path):
    all_files = [] # empty list
    dirs = os.listdir(path) # enter the path of the file, return the
                            # dictionary name of path name
    for file in dirs: # make a for loop to extract file names one by one
        ufile = file.decode('utf-8') # decode to Unicode
        all_files.append(ufile) # add a single element to the end of the list
    return all_files

# This function returns a selected subset of filenames as a list.
def select_filenames(all_files):
    selected_files = [] # empty list
    for filename in all_files: # make a for loop to extract file names one by
                               # one
        # Check that filename starts with certain kind of string and file type
        # is ".txt".
        if filename.startswith("finer_kokoelma_") and file-
name.endswith(".txt"):
            selected_files.append(filename) # add a single element to the end
                                            # of the list
    return sorted(selected_files) # return alphabetically sorted list

# This function opens and reads file's contents and returns it as Python's
# Unicode string.
def get_file_contents(filename):
    orig_file = open(filename, "r") # open the original file for reading
    filetext = orig_file.read() # read the contents of the file
    ufiletext = filetext.decode('utf-8') # decode to Unicode
    orig_file.close() # close the file
    return ufiletext

# This function creates a copy from the original filename.
def modify_filename(filename):
    new_name = "monthyear_" + filename # create a new filename from the
                                       # original name
```

```
    modified_filename = new_name.decode('utf-8') # decode to Unicode
    return modified_filename

# First the function checks that the given filename doesn't exist already
# (if so, an error message is given and program exits).
# Then the function writes a copy of the file content to the newly named file.
def write_to_file(modified_filename, result_letters):
    if os.path.exists(modified_filename): # check if the filename already
                                          # exists to avoid overwriting
        print "File already exists. Please use another name."
        sys.exit("Exiting.")
    else:
        new_filetext = result_letters.encode('utf-8') # encode to UTF-8
        new_file = open(modified_filename, "w") # open the new file for
                                                # writing
        new_file.write(new_filetext) # write to file
        new_file.close() # close the file

# This function searches the letter collection and filters relevant letters
# based on search term.
def search_and_filter(letter_text, search_term1, search_term2):
    results = "" # empty string
    letters = letter_text.split("<letter>") # split letters apart from each
                                            # other
    for letter in letters: # check letters one by one in one collection at a
                           # time
        tag1 = letter.find("<text>") # check only the actual letter text
                                     # starting with this tag...
        tag2 = letter.find("</text>", tag1) # ... and ending with this tag
        text = letter[tag1+len("<text>"):tag2]
        # accept only the letters with some form of tag "<TimexTmeDat>"
        if "TimexTmeDat" in text:
            # check if any of the given years/months are found in the letter
            # text
            if ((text.find(search_term1, 0, 200) > 0) or
(text.find(search_term2, 0, 200) > 0)):
                results = results + "\n" + "<letter>" + letter # collect
                                            # results of the relevant letters
    return results
#
# Begin main
#
all_files = read_all_filenames(path) # in Unicode
selected_files = select_filenames(all_files) # in Unicode
for filename in selected_files: # in Unicode
    utext = get_file_contents(filename) # in Unicode
    year_letters = search_and_filter(utext, year1, year2) # in Unicode
    month_letters = search_and_filter(year_letters, month1, month2) # in
                                                                # Unicode
    copy_name = modify_filename(filename) # in Unicode
    write_to_file(copy_name, month_letters) # filenames written in UTF-8
```

LIITE 6: Python-ohjelma “finer_find_date.py”

```
# This file: "finer_find_date.py"
# Usage: python finer_find_date.py
# Comment mark: #. Comments and the program are written in English, because
# Finnish alphabets might cause character encoding problems.
#
# Created by: Selja Vanamo (selja.vanamo@tuni.fi) at March 2019
#
# The purpose of this program is to find letters from certain exact date
# and extract them to a new file. First the files have gone thorough FiNER's
# named entity recognition. For the search only date entity tagged letters
# (some form of "<TimexTmeDat>") have been excepted.

import nltk # Natural Language Toolkit
import os, sys # os for operating system dependent interface, sys for system
               # spesific parameters
import os.path # for checking existing paths
import re # for regular expressions

# This is the path to the directory, x-marks indicate your basic user account
# number.
path = "/home/stud/sis/xxxxxxx"

# Change the numbers or strings of the search terms according to your search
# interest.
date1 = "2[45][.,/ ]12" # accept any character within []
date2 = "[jJ]oulu[^k]" # accept word starting with "joulu" with the letter "j"
                       # or "J" but not for example "joulukuu"

# This function returns all the filenames from the given directory as a list.
def read_all_filenames(path):
    all_files = [] # empty list
    dirs = os.listdir(path) # enter the path of the file, return the
                            # dictionary name of path name
    for file in dirs: # make a for loop to extract file names one by one
        ufile = file.decode('utf-8') # decode to Unicode
        all_files.append(ufile) # add a single element to the end of the list
    return all_files

# This function returns a selected subset of filenames as a list.
def select_filenames(all_files):
    selected_files = [] # empty list
    for filename in all_files: # make a for loop to extract file names one by
                               # one
        # Check that filename starts with certain kind of string and file type
        # is ".txt".
        if filename.startswith("finer_kokoelma_") and file-
name.endswith(".txt"):
            selected_files.append(filename) # add a single element to the end
                                            # of the list
    return sorted(selected_files) # return alphabetically sorted list

# This function opens and reads file's contents and returns it as Python's
# Unicode string.
def get_file_contents(filename):
    orig_file = open(filename, "r") # open the original file for reading
    filetext = orig_file.read() # read the contents of the file
    ufiletext = filetext.decode('utf-8') # decode to Unicode
    orig_file.close() # close the file
    return ufiletext

# This function creates a copy from the original filename.
def modify_filename(filename):
    new_name = "date_" + filename # create a new filename from the original
                                  # name
```

```
    modified_filename = new_name.decode('utf-8') # decode to Unicode
    return modified_filename

# First the function checks that the given filename doesn't exist already
# (if so, an error message is given and program exits).
# Then the function writes a copy of the file content to the newly named file.
def write_to_file(modified_filename, result_letters):
    if os.path.exists(modified_filename): # check if the filename already
                                          # exists to avoid overwriting
        print "File already exists. Please use another name."
        sys.exit("Exiting.")
    else:
        new_filetext = result_letters.encode('utf-8') # encode to UTF-8
        new_file = open(modified_filename, "w") # open the new file for
                                                # writing
        new_file.write(new_filetext) # write to file
        new_file.close() # close the file

# This function searches the letter collection and filters relevant letters
# based on search term.
def search_and_filter(letter_text, search_term1, search_term2):
    results = "" # empty string
    letters = letter_text.split("<letter>") # split letters apart from each
                                            # other
    for letter in letters: # check letters one by one in one collection at a
                           # time
        tag1 = letter.find("<text>") # check only the actual letter text
                                     # starting with this tag...
        tag2 = letter.find("</text>", tag1) # ... and ending with this tag
        text = letter[tag1+len("<text>"):tag2]
        # accept only the letters with some form of tag "<TimexTmeDat>"
        if "TimexTmeDat" in text:
            # check if any of the given dates are found in the letter text
            if ((re.search(date1, text)) or (re.search(date2, text))):
                results = results + "\n" + "<letter>" + letter # collect
                                                # results of the relevant letters
    return results
#
# Begin main
#
all_files = read_all_filenames(path) # in Unicode
selected_files = select_filenames(all_files) # in Unicode
for filename in selected_files: # in Unicode
    utext = get_file_contents(filename) # in Unicode
    date_letters = search_and_filter(utext, date1, date2)
    copy_name = modify_filename(filename) # in Unicode
    write_to_file(copy_name, date_letters) # filenames written in UTF-8
```