Tekstifragmenttien välisen semanttisen samanlaisuuden tunnistaminen

Lili Aunimo

Helsinki 11. tammikuuta 2002

Pro gradu -tutkielma

HELSINGIN YLIOPISTO
Yleisen kielitieteen laitos
Kieliteknologian oppiaine

Tekstifragmenttien välisen semanttisen samanlaisuuden tunnistaminen

Lili Aunimo

Pro gradu -tutkielma
Yleisen kielitieteen laitos
Helsingin yliopisto
11. tammikuuta 2002, 68 sivua + 13 liitesivua

Tekstifragmenttien välisen semanttisen samanlaisuuden tunnistaminen on yleinen tehtävä luonnollista kieltä käsittelevissä järjestelmissä kuten tiedonhakujärjestelmät, tekstin luokittelu- ja ryvästysjärjestelmät, tiivistelmän generointijärjestelmät sekä kysymysvastausjärjestelmät. Koska tehtävä on yleinen ja koska sitä on tutkittu jo melko paljon, on sen suorittamiseen kehitetty useita erilaisia menetelmiä.

Tutkielmassa käsitellään tekstin leksikaaliseen koheesioon, leksikaalisiin ketjuihin ja vektorimalliin perustuvia menetelmiä fragmenttien välisen samanlaisuuden laskemiseksi. Tutkielman kokeellisessa osassa tutkittiin erilaisia tapoja laskea fragmenttien välistä samanlaisuutta vektorimallin avulla. Samanlaisuutta tunnistava komponentti toteutettiin osana kysymysvastausjärjestelmää.

Tutkimuksessa päädyttiin siihen, että tekstifragmenttien välistä samanlaisuutta voidaan tunnistaa tyydyttävästi melko yksinkertaisilla menetelmillä, mutta että hyvien tai erinomaisten tunnistustulosten saavuttaminen on vaikeaa.

Aiheluokat (Computing Reviews 1998): H.3.1, H.3.3, H.3.4, I.2.7

Avainsanat: luonnollisen kielen käsittely, semantiikka, tiedonhaku, kysymysvastausjärjestelmät

Sisältö

1 Johdanto

Tekstifragmenttien välisen semanttisen samanlaisuuden havaitsemiseksi on olemassa monia menetelmiä. Se johtuu osittain jo siitä, että fragmenttien välistä semanttista samanlaisuutta on monenasteista: vahvimmillaan semanttinen samanlaisuus on sitä, että tekstifragmentit tarkoittavat täysin samaa vaikka niissä on käytetty osittain tai täysin eri sanoja. Heikoimmillaan tekstifragmenttien samanlaisuus jää siihen, että niissä käsitellään osittain samaa aihealuetta. Täysin identtisten tekstifragmenttien samanlaisuuden havaitsemista ei tässä tutkielmassa käsitellä. Toinen tärkeä syy fragmenttien välistä semanttista samanlaisuutta tunnistavien menetelmien runsauteen on se, että ongelmaa on tutkittu paljon ja monesta eri näkökulmasta, sillä useat eri sovellukset tarvitsevat samanlaisuutta tunnistavan komponentin. Sovelluksista kerrotaan lisää luvussa 1.2. Seuraavaksi esitellään taustoja tekstifragmenttien semanttista samanlaisuutta tunnistaville menetelmille. Luvussa 1.3 käydään lyhyesti läpi tutkielman rakenne.

1.1 Taustaa

Tekstifragmenttien välisen samanlaisuuden havaitsemisessa käytetyt menetelmät voidaan karkeasti jakaa päättelysääntöihin ja numeeriseen laskentaan perustuviin menetelmiin. Päättelysääntöihin perustuvat menetelmät pyrkivät jonkinasteiseen tekstin ymmärtämiseen kun taas numeeriseen laskentaan perustuvat menetelmät tyytyvät mittaamaan fragmenteissa esiintyvien sanojen samanlaisuutta.

Tutkielman päämääränä on esitellä tekstifragmenttien välistä samanlaisuutta tunnistavia numeeriseen laskentaan ja tekstin leksikaaliseen koheesioon perustuvia menetelmiä sekä teoreettisesta että käytännöllisestä näkökulmasta. Tutkielman kokeellisessa osuudessa verrataan toisiinsa erilaisia tapoja toteuttaa frag-

menttien välistä samanlaisuutta tunnistava komponentti kysymysvastausjärjestelmään. Tutkielma on tehty APPA- tutkimusprojektia[1] varten ja kokeellisessa osassa käytetty kysymysvastauskorpus on saatu käyttöön projektin ansiosta.

Tutkielmassa ei esitetä menetelmiä tekstin täydellisen semanttisen analyysin aikaansaamiseksi, vaan tyydytään kuvaamaan tekstiä sekä yksinkertaisella bag-of-words -lähestymistavalla[2] että sellaisilla menetelmillä, jotka hyödyntävät tekstin morfologista ja syntaktista analyysia sekä semanttisia verkkoja ja tekstin leksikaalisen koheesion tunnistamista.

Kattavan semanttisen kuvauksen muodostaminen tekstistä tarkoittaisi sitä, että tekstin sisältö kuvattaisiin jollakin kieliriippumattomalla esitystavalla kuten ensimmäisen kertaluvun logiikan lauseilla ja niihin liittyvillä päättelysäännöillä. Tämä olisi erittäin haastava tehtävä ja se edellyttäisi, että käytettävissä olisi jokin sellainen kieliriippumaton tapa, jolla maailmantietoa voisi yleisesti ja kattavasti kuvata. Käytännössä tämän tyyppistä lähestymistapaa on sovellettu joissakin tarkasti rajatuissa mikromaailmoissa sillä rajoituksella, että käsittelyn kohteena oleva teksti on muodostunut tarkasti rajatusta luonnollisen kielen osajoukosta. Tätä lähestymistapaa edustavat mm. Patrick Blackburn ja Johan Bos [BB99].

Fragmentilla (engl. fragment, block) tarkoitetaan tässä tutkielmassa sellaista tekstin osaa, jonka pituutta ei ole tarkasti määritelty. Maksimipituus tekstifragmentille on koko dokumentti ja minimipituus yksi sana. Fragmentin pituus vaihtelee tekstiä käsittelevän sovelluksen vaatimusten perusteella. Tässä tutkielmassa

[1] APPA (eli AsiakasPalautteen Prosessointi Automaattisesti) on Helsingin yliopiston Tietojenkäsittelytieteen laitoksen Doremi-tutkimusryhmän projekti, jonka rahoittajana on elinkeinoelämä. http://www.cs.helsinki.fi/research/doremi/

[2] *Bag-of-words* -menetelmät ovat yleinen nimitys sellaisille tavoille kuvata tekstin sisältöä, joissa kaikki tekstissä esiintyvien sanojen keskinäiseen järjestykseen liittyvä tieto on jätetty huomioimatta. Tällaisessa kuvaustavassa esim. *I eat what I see* ja *I see what I eat* saavat täysin identtisen kuvauksen [JM00].

esitetyissä sovelluksissa fragmentit eivät sisällä päällekkäisiä tekstinosia kuten esim. tekstin katkelmointisovellukissa käytetyt fragmentit [Hea97]. *Katkelma*lla (engl. segment, passage) tässä tutkielmassa tarkoitetaan useammasta kuin yhdestä fragmentista koostuvaa tekstin osaa. Tällöin oletetaan, että fragmentit eivät ole koko dokumentin pituisia.

1.2 Sovellukset

Tekstifragmenttien välistä semanttista samanlaisuutta tunnistavat menetelmät on usein kehitetty jotakin tiettyä sovellusta ja tekstityyppiä varten. Sovellukset ja tekstityyppi määrittävät sekä sen, minkä asteista samanlaisuutta tekstifragmenttien välillä pyritään tunnistamaan että sen, minkä pituisia tekstifragmentit milloinkin ovat. Kyseessä oleva tekstityyppi vaikuttaa lisäksi jonkin verran myös käytettyihin menetelmiin.

Tutkielman luvussa 6 on tutkittu erilaisia menetelmiä tekstifragmenttien välisen samanlaisuuden havaitsemiseksi kysymys-vastausjärjestelmässä. Kysymys-vastausjärjestelmässä tekstifragmentit ovat suhteellisen lyhyitä, sillä ne sisältävät useimmiten vain yhden kysymyksen tai yhden vastauksen. Toisaalta, kun uutta kysymystä verrataan vanhoihin kysymyksiin sen selvittämiseksi, onko kysymykseen jo olemassa vastaus, vaaditaan fragmenttien samanlaisuutta tunnistavalta komponentilta suhteellisen tarkkaa samanlaisuuden tai erilaisuuden tunnistamiskykyä.

Muita sovelluksia, joissa tekstifragmenttien välistä samanlaisuutta tunnistavia menetelmiä käytetään ovat tekstin luokittelu, ryvästäminen, tiivistelmän generointi ja tiedonhaku. Tekstin luokittelussa ja ryvästämisessä vertaillaan dokumentteja toisiinsa ja samaan luokkaan tai rypääseen kootaan ne, jotka ovat sisällöltään samanlaisimpia. Luokittelussa muodostettavat luokat ovat tiedossa etu-

käteen, mutta ryvästämisessä ei rypäiden aihepiirejä eikä välttämättä niiden lukumääräkään tiedetä ennen kuin ryvästäminen on suoritettu.

Tiivistelmän generoinnissa toimitaan usein siten, että ensin pilkotaan teksti eri aiheita käsitteleviin katkelmiin ja sitten poimitaan tiivistelmään kustakin katkelmasta sitä parhaiten kuvaavat lauseet. Katkelmien muodostamisessa teksti jaetaan pieniin fragmentteihin ja keskenään samanlaiset peräkkäin sijaitsevat fragmentit valitaan samaan katkelmaan.

Tiedonhaussa verrataan käyttäjän esittämän hakulauseen ja dokumenttien välistä samanlaisuutta. Hakutuloksena palautetaan lista niistä dokumenteista, jotka ovat samanlaisimpia hakulauseen kanssa. Jos dokumentit ovat pitkiä, katkelmoidaan ne ensin ja hakulausetta vertaillaan dokumenttien sijaan katkelmiin ja käyttäjälle palautetaan dokumenttilistan sijaan lista katkelmista.

Tekstifragmenttien välistä samanlaisuutta voidaan tunnistaa sekä samankielisten että eri kielisten dokumenttien välillä. Vaikka eri kielillä kirjoitettujen tekstifragmenttien välistä semanttista samanlaisuutta voidaan tunnistaa samoilla perusmenetelmillä kuin samaa kieltä olevien fragmenttienkin samanlaisuutta, on monikielisillä menetelmillä omat erityispiirteensä, ja niitä ei tässä tutkielmassa käsitellä.

1.3 Tutkielman rakenne

Alkuosan luvuissa 2, 3 ja 4 esitetään tutkielman teoriapohja ja luvussa 6 kuvaillaan käytettyä kysymysvastauskorpusta ja eri toteutusvaihtoehtoja fragmenttien välistä samanlaisuutta tunnistavan komponentin toteuttamiseksi sekä raportoidaan eri toteutusvaihtoehtojen evaluoinnin tulokset. Koska järjestelmien ja algoritmien evaluointimenetelmät ovat tärkeässä asemassa silloin kun kehitetään uusia menetelmiä vanhojen rinnalle tai kun tutkitaan, mikä olemassaolevista me-

netelmistä sopisi parhaiten tiettyyn sovellukseen, on evaluointimenetelmiä varten varattu oma luku. Luvussa 5 käydään läpi erilaisia tapoja menetelmien tehokkuuden evaluointiin. Tutkielman viimeisessä luvussa eli luvussa 7 esitetään lyhyesti tutkimuksen tulokset, vedetään niistä johtopäätökset ja kerrotaan, mihin suuntaan tutkimusta voisi saatujen tulosten pohjalta jatkaa.

Tutkielman teoriapohjan sisältävät kolme lukua on jaoteltu siten, että luvussa 2 käsitellään sekä yksittäisten sanojen että monisanaisten termien välistä samanlaisuutta ja niiden välisiä semanttisia suhteita. Luku 2 sisältää myös tekstin koheesiota ja koherenssia käsittelevän osuuden. Luvussa 3 esitellään sanojen morfosyntaktisiin piirteisiin perustuva menetelmä, jolla tekstistä voidaan poimia sisällön kannalta merkityksellisimmät sanat. Luvussa 4 esitellään numeeriseen laskentaan perustuvista samanlaisuuden havaitsemismenetelmistä yksityiskohtaisesti vektorimalli ja leksikaaliset ketjut.

2 Sanojen välinen semanttinen samanlaisuus ja semanttiset suhteet

Monet tekstifragmenttien välistä semanttista samanlaisuutta tunnistavista menetelmistä perustuvat siihen, että tarkastellaan tekstissä esiintyvien yksittäisten sanojen välistä semanttista samanlaisuutta ja niiden välisiä semanttisia suhteita. Koska käsite *sana* on moniselitteinen [Kar98], kerrotaan seuraavaksi missä merkityksessä sitä käytetäään tässä tutkielmassa sekä esitellään joitakin siihen läheisesti liittyviä käsitteitä. Sen jälkeen luvussa 2.3 kerrotaan sanoihin liittyvää tietämystä sisältävistä sanatietokannoista yleisesti, luvussa 2.4 perehdytään leksikaaliseen koheesioon, joka on jossakin muodossa pohjana kaikissa numeeriseen laskentaan perustuvissa tekstifragmenttien välistä semanttista samanlaisuutta tunnistavissa

menetelmissä. Luvussa 2.5 esitellään WordNet, joka on yleisesti käytössä englannin kielen semantiikkaan liittyvien sovellusten ja menetelmien tutkimuksessa.

2.1 Sana ja siihen liittyviä käsitteitä

Tässä tutkielmassa sanalla tarkoitetaan *ortografista sanaa* eli merkkijonoa, joka on erotettu ympäristöstään välilyöntien tai välimerkkien avulla. Sanaan läheisesti liittyvä käsite on *lekseemi* eli sanan leksikkomuoto. Sillä tarkoitetaan yhteenkuuluvien sananmuotoesiintymien luokkaa [Kar98]. Esimerkiksi sananmuodot *varvas, varpaan* ja *varpaillaan* edustavat lekseemiä *VARVAS*. Lekseemi on abstrakti käsite, joka ei itse voi esiintyä tekstissä, vaan se toteutuu jonain sananmuotona. Sanakirjoissa lekseemin sananmuotoja edustaa *sanakirjamuoto*. Sanakirjamuotona käytetty sananmuoto voi vaihdella kielestä toiseen. Esim. suomessa verbien sanakirjamuotona käytetään 1. infinitiiviä (*olla, nukkua*) kun taas latinassa käytössä on aktiivin indikatiivin preesensin 1. persoona (*olen, nukun*) [Kar98]. Lekseemi voi myös koostua useasta eri sanasta, esim. *atom bomb*. Lisäksi kaksi eri lekseemiä voivat olla joko kirjoitus- tai ääntämysasultaan tai molempien puolesta samanlaisia, esim. *kuusi* (puu) ja *kuusi* (numero). Tällaisia lekseemejä kutsutaan *homonyymeiksi*. Yhdellä lekseemillä voi olla enemmän kuin yksi merkitys, mutta tällöin merkitysten pitää liittyä kiinteästi toisiinsa, esim. *selkä* sanaliitoissa *yön selkä* ja *kirjan selkä*. Tällainen lekseemi on polyseeminen ja ilmiön nimi on *polysemia*. Polysemian ja homonymian välinen raja on useissa tapauksissa tulkinnanvarainen. Tutkielman kokeellisessa osassa esiintyy termi *lemma*. Lemma on yhteenkuuluvien sananmuotojen joukko, joka - toisin kuin lekseemi - on muodostettu ottamatta kantaa joukon perusmuodon määrittämisen ongelmaan.

2.2 Sanojen välinen semanttinen samanlaisuus

Sanojen välinen semanttinen samanlaisuus voi perustua siihen, että ne edustavat samaa leksikkomuotoa tai samaa merkitystä jos kyseessä on polyseeminen lekseemi, esim. sanojen *kalan* ja *kalassa* leksikkomuoto on *kala* tai sitten siihen, että sanojen leksikkomuotojen tai polyseemisten lekseemien leksikkomuotojen merkitysten välillä on jokin semanttinen suhde, esim. synonymia. Semanttisia suhteita käsitellään tarkemmin luvussa 2.4. Koska sekä sanan leksikkomuodon selvittäminen puhumattakaan kyseessä olevan polyseemisen leksikkomuodon merkityksen selvittämisestä ovat usein liian vaikea ongelma ratkaistavaksi automaattisesti, esitetään seuraavaksi kolme yksinkertaistavaa lähetymistapaa sanojen välisen semanttisen samanlaisuuden tunnistamiseksi.

Ensimmäisessä lähestymistavassa sanojen välistä samanlaisuutta tunnistetaan sanojen lemmojen ja mahdollisesti myös niiden välisten semanttisten suhteiden avulla. Tällöin käsitellään usein samaan sanaluokkaan kuuluvia homonyymejä aivan kuin ne edustaisivat samaa lekseemiä, sillä silloin säästytään sanojen semanttiselta yksiselitteistämiseltä. Semanttisen yksiselitteistämisen merkitys olisi kuitenkin sitä tärkeämpi mitä useampia ja hienojakoisempia semanttisia suhteita menetelmässä käytetään, sillä esim. polyseemisella lekseemillä saattaa olla jokaiselle sen eri merkitykselle erilaiset semanttiset suhteet. Homonyymeillä on aina keskenään erilaisia semanttisia suhteita.

Toinen tapa yksinkertaistaa sanojen välisen semanttisen samanlaisuuden tunnistamisongelmaa on se, että poistetaan sanoista päätteet ja pidetään samanlaisina niitä sanoja, jotka tämän jälkeen koostuvat samasta merkkijonosta. Tätä kutsutaan *stemmaukseksi* (engl. stemming), ja sen etu on yksinkertaisuus. Toinen hyvä puoli stemmauksessa saattaa olla se, että se voi nostaa kuvauksen abstraktiotasoa muuttamalla saman vartalon sisältävät sanat samaksi merkkijonoksi. Esimerkik-

si jos lekseemit *effect*, *effective* ja *effectiveness* stemmataan, saadaan tulokseksi yksi kaikkia lekseemejä kuvaava merkkijono *effect*. Jos suomen kielen sanoja stemmataan, sattaa tulos astevaihtelun takia olla esimerkiksi seuraavanlainen: *kotiansa* on stemmauksen jälkeen *koti* ja *kode+i+ssa kode*. Stemmaus on käytössä lähinnä joissakin englannin kieltä käsittelevissä sovelluksissa. Tunnetuin englannin stemmausalgoritmi on Porterin algoritmi [JM00].

Kolmas yksinkertainen tapa verrata sanojen välistä samanlaisuutta on se, että käsitellään ainoastaan yksisanaisia lekseemejä. Sellaisissa kielissä, joissa on vain vähän yhdyssanoja, kuten englanti, tällä on merkitystä, sillä esim. *atom bomb* olisi kaksi lekseemiä *atom* ja *bomb* kun se suomessa on käsittelytavasta riippumatta vain yksi lekseemi *atomipommi*. Toisaalta on aiheellista kysyä pitäisikö sellaisissa kielissä, kuten suomi ja saksa, jotka ovat tunnettuja tuotteliaasta yhdyssananmuodostuksesta, pilkkoa yhdyssanat mahdollisimman moneksi lekseemiksi tai jopa morfeemiksi. Tällainen lähestymistapa toimii, jos yhdyssanan merkitys on *kompositionaalinen* eli osiensa merkitysten summa. Yhdyssanojen pilkkominen lekseemeihin tai morfeemeihin olisi yksi tapa, jolla voitaisiin hieman vähentää käsiteltävänä olevien erilaisten sanojen määrää. Tästä voi olla hyötyä silloin, kun tekstifragmenttien välistä samanlaisuutta pyritään tunnistamaan vektorimallin avulla. Vektorimallia käsitellään tarkemmin luvussa 4.1.

2.3 Sanatietokannat

Jos tekstin samanlaisuutta ei tarkastella pelkkänä merkkijonojen samuutena, olivatpa sanat sitten stemmattuja tai eivät, tai jos tekstin samanlaisuuden havaitsemisessa halutaan käyttää hyväksi semanttisia suhteita, tarvitaan taustalle jonkinlainen sanatietokanta. Tällaisena sanatietokantana voivat toimia leksikko ja siihen liittyvät säännöstöt, tavallinen sanakirja, kielitieteellinen merkityssanakirja,

erikoiskielen sanasto eli terminologia, asiasanasto eli tesaurus, semanttinen verkko tai ontologiat.

Tässä tutkielmassa käytetään Karlssonin määritelmää leksikosta [Kar98]. Sen mukaan *leksikko* on kielen vakiintuneiden yksiköiden varasto. Se voi sisältää sanoja, yhdyssanoja, johdoksia ja morfeemeja. Jotta leksikkoa voisi käyttää, tarvitaan myös säännöstöt, jotka kertovat miten morfeemeja voi liittää toisiinsa ja mitä muunnoksia liitoksen yhteydessä pitää tehdä. Leksikko on käytössä lähinnä teoreettisessa kielentutkimuksessa, kun taas muut tässä esitellyt sanantietokannat ovat käytännön sovelluksia.

Tavallinen *sanakirja* on aakkosellinen luettelo kaikkiin sanaluokkiin kuuluvia lekseemejä luokitteluineen, määritelmineen, kuvauksineen ja esimerkkeineen [Kar98]. *Deskriptiivinen sanakirja* kuvaa vallitsevaa kielenkäyttöä ja normatiivinen sanakirja antaa ohjeita sanojen merkityksestä ja käytöstä [Tek89].

Kielitieteellinen merkityssanakirja (engl. thesaurus) on tavallisen sanakirjan käänteiskuvaus, sillä sanakirja kuvaa sanojen merkityksiä, mutta kielitieteellinen merkityssanakirja kertoo mitkä sanat parhaiten kuvaavat tiettyjä merkityksiä [MH91]. Kielitieteellisessä merkityssanakirjassa sanoja ei ole järjestetty aakkosjärjestykseen vaan niiden merkityksen mukaiseen järjestykseen. Näin samaa tarkoittavat sanat eli synonyymit löytyvät samasta kohdasta. Kielitieteellinen merkityssanakirja ilmaisee monia muitakin semanttisia suhteita synonymian lisäksi, mutta toisin kuin semanttisessa verkossa, niitä ei välttämättä ole nimetty [Fel98a].

Erikoiskielen sanasto eli *terminologia* on monessa suhteessa samanlainen kuin synonyymisanasto, sillä se on järjestetty käsitteiden eikä sanojen tai termien mukaan [Tek89]. Synonyymisanastosta terminologia eroaa siten, että siinä kuvataan vain johonkin tiettyyn erityiskieleen liittyviä termejä ja käsitteitä ja että termien selitykset on pyritty esittämään mahdollisimman yksikäsitteisesti ja selkeästi. Ter-

minologian pohjimmainen tarkoitus on kielenkäytön ohjaaminen ja standardointi.

Asiasanasto eli *tesaurus* on termiluettelo, jota käytetään dokumenttien tallennukseen ja hakuun [Tek89]. Se sisältää sekä asiasanoja että ohjaustermejä. *Ohjaustermit* ovat sanoja, jotka on haussa korvattava asiasanoilla. Asiasanoihin on tesauruksessa liitetty viittaukset niiden ylä- ja alakäsitteisiin. Esimerkki tunnetusta tesauruksesta on YSA eli Yleinen suomalainen asiasanasto, joka on käytössä mm. kirjastoissa. Monet tesaurukset kattavat vain jonkin tietyn alan. Esimerkki tällaisesta tesauruksesta on Korkeimman oikeuden asiasanasto, joka on osa Oikeusministeriön Finlex -verkkopalvelua [Fin01]. Seuraavat esimerkit ohjaustermeistä ja asiasanoista on poimittu Korkeimman oikeuden ohjaussanastosta: ohjaustermi: *herjaus*, asiasana: *kunnianloukkaus* ja ohjaustermi: *jäävi*, asiasana: *esteellisyys*.

Semanttinen verkko on verkko, jonka solmuina ovat käsitteet ja kaarina nimetyt suhteet solmujen välillä [JM00]. Käsitteisiin voi kielestä riippuen liittyä tai olla liittymättä niitä kuvaavia lekseemejä. WordNet, jota käsitellään tarkemmin luvussa 2.5 on esimerkki semanttisesta verkosta. Semanttiset verkot on usein luotu siten, että ne ovat helposti tietokoneohjelmistojen hyödynnettävissä kun taas kielitieteelliset merkityssanakirjat on alun perin tehty ihmisen käytettäväksi.

Ontologia on kuvaus johonkin tiettyyn mikromaailmaan (engl. microworld) kuuluvista olioista. *Taksonomia* on puumainen kuvaus ontologian olioista. Taksonomialle voidaan asettaa koko joukko rajoituksia jotka tekevät siitä huomattavasti muodollisemman kuvauksen kuin ontologiasta [JM00]. Ontologioiden ja taksonomioiden ero edellä esitettyihin sanatietokantoihin on se, että ne eivät pyri kuvaamaan yleiskieltä, vaan jollakin tietyllä alalla esiintyviä käsitteitä.

2.4 Tekstin koheesio ja koherenssi

Tekstin koheesio ja koherenssi ovat hyviä tekstin semanttisen samanlaisuuden ilmentäjiä. Seuraavaksi esitellään ensin koheesio ja koherenssi yleisesti ja sen jälkeen leksikaalinen koheesio tarkemmin. Leksikaaliseen koheesion keskittyminen johtuu siitä, että ainoastaan sen tunnistamiseksi on olemassa yleisesti tunnettuja automaattisia menetelmiä.

Koheesio tarkoittaa tekstin kuulumista yhteen. Se ilmenee sekä leksikaalisena koheesiona että *kieliopillisena koheesiona* [HH76]. Kieliopillisen koheesion muotoja ovat *viittaus, korvaus, ellipsi* ja *konjunktio*. Esimerkki viittauksesta on *anafora* eli viittaus aikaisempaan tekstin osaan, esim. *Hän* virkkeissä *Leena on syömässä. Hän oli hyvin nälkäinen.* Korvaus tarkoittaa, että jokin sana on korvattu toisella saman kieliopillisen kategorian sanalla, esim. *does* virkkeissä *You think Joan alredy knows? - I think everybody does.* Ellipsissä lauseesta on jätetty pois sellainen osa, joka ei ole välttämätön ymmärtämisen kannalta. Esim. *kirves* on jätetty pois jälkimmäisestä virkkeestä seuraavassa: *Kirveeni on liian tylsä. Minun täytyy hankkia terävämpi.* Konjunktiossa lauseet on yhdistetty toisiinsa jonkin konjunktion avulla, esim. *Hän oli iloinen, mutta ujo.*

Koheesioon läheisesti liittyvä tekstin ominaisuus on *koherenssi*. Koherenssi tarkoittaa sitä, että teksti muodostaa järkevän kokonaisuuden. Ilmiöinä koheesiota voidaan verrata syntaksiin ja koherenssia semantiikkaan [MH91]. Koherenssi muodostuu mm. selityksestä, tarkennuksesta, syy-seuraussuhteesta ja esimerkistä. Ilmiön moniuloitteisuutta ja monimutkaisuutta kuvaa se, että kielitieteen tutkijoiden parissa ei ole yksimielisyyttä siitä, mistä kaikista suhteista koherenssi muodostuu [MH91]. Esimerkki koherenssista: *Jussi osti sadetakin. Hän oli ostoksilla eilen Aleksanterinkadulla sateessa.* Esimerkkitapaus voitaisiin luokitella ostostapahtuman tarkennukseksi tai syy-seuraussuhteeksi.

2.4.1 Leksikaalinen koheesio

Leksikaalinen koheesio tarkoittaa lekseemien liittymistä toisiinsa semanttisilla suhteilla [MH91]. Halliday ja Hasan [HH76] jakavat leksikaalisen koheesion tunnistamisessa käytetyt semanttiset suhteet seuraaviin viiteen kategoriaan:

1. Toisto, joka sisältää viittauksen

   Esimerkki 1 Maija haukkasi *persikkaa*. Valitettavasti *persikka* ei ollut kypsä.

2. Toisto, joka ei sisällä viittausta

   Esimerkki 2 Maija söi *persikoita*. Hän pitää *persikoista*.

3. Toisto yläkäsitteen kautta

   Esimerkki 3 Maija söi *persikoita*. Hän pitää *hedelmistä*.

4. Systemaattinen semanttinen suhde

   Esimerkki 4 Maija pitää *vihreistä* omenista, mutta ei *punaisista*.

5. Epäsystemaattinen semanttinen suhde eli kollokaatio

   Esimerkki 5 Maija vietti kolme tuntia *puutarhassa* eilen. Hän *istutti* perunoita.

Ylläolevasta luokittelusta ensimmäiset kolme suhdetta perustuvat *toisto*on. Saman sanan toistamisen lisäksi voidaan käyttää hyperonyymeja, hyponyymeja ja synonyymeja. *Hyponymia* tarkoittaa, että kahden sanan välillä vallitsee sellainen epäsymmetrinen ja transitiivinen relaatio, jossa toinen sanoista on toisen yläkäsite, esim. *kala, taimen* ja *eläin, kala*. Yleisempää sanaa kutsutaan *hyperonyymiksi* ja erityisempää sanaa *hyponyymiksi*. *Synonymia* puolestaan määritellään siten, että se

on suhde, joka vallitsee kahden tai useamman lekseemin välillä silloin, kun kyseisillä lekseemeillä on sama tarkoite ja samat semanttiset suhteet [HO93]. Toisin sanoen sanat ovat synonyymeja, jos lauseilla, joissa ne on korvattu toisillaan on sama merkitys. *Täydellisiä synonyymeja* ovat sellaiset sanat, jotka voidaan kaikissa konteksteissa korvata toisillaan ilman, että lauseen merkitys muuttuu. *Osittaisia synonyymeja* ovat sellaiset sanat, jotka eivät ole täydellisiä synonyymeja ja joilla on olemassa sellainen konteksti, jossa ne voidaan korvata toisillaan ilmaisun merkityksen muuttumatta. Joidenkin käsitysten mukaan täydellisiä synonyymeja ei ole olemassa, vaan on vain eri asteisia osittaisia synonyymeja.

Hallidayn ja Hasanin luokittelun neljäs leksikaalisen koheesion ilmentymä on *systemaattinen semanttinen suhde*. Systemaattisia semanttisia suhteita ovat mm. *antonymia* eli toistensa vastakohtaa tarkoittavat sanat, meronymia ja ko-hyponymia. *Meronymia* on muuten hyponymiaan verrattavissa oleva suhde, paitsi että suhteen toisen käsitteen tarkoite on osa toisen käsitteen tarkoitetta. Esim. *paita, hiha* ja *hiha, lanka*. Kokonaisuuteen viittavaa sanaa kutsutaan *holonyymiksi* ja osaan viittaavaa *meronyymiksi*. *Ko-hyponymia* on systemaattinen semanttinen suhde, joka vallitsee sellaisten sanojen välillä, joilla on sama hyperonyymi. Esimerkiksi sanat *punainen, musta, valkoinen* ovat keskenään systemaattisessa semanttisessa suhteessa sillä perusteella, että niiden kaikkien hyperonyymi on *väri*. Ko-hyponyymejä kutsutaan myös toistensa *vieruskäsitteiksi*.

Viides leksikaalisen koheesion ilmentymä on *epäsystemaattinen semanttinen suhde*. Epäsystemaattisia suhteita on vaikea kuvata ja ne edellyttävät päättelyä suuresta tekstikorpuksesta. Epäsystemaattisessa suhteessa keskenään olevia sanoja kutsutaan paitsi kollokaatioiksi, myös *myötäesiintymiksi* (engl. co-occurring words) [HO93]. Usein epäsystemaattisessa suhteessa keskenään olevat sanat liittyvät johonkin tiettyyn tapahtumaan. Esimerkiksi postitoimistossa asioimisen kautta toisiinsa liittyviä sanoja voisivat olla: *postitoimisto, postimerkki, ostaa, lähteä*. Jos sanat

irroitetaan niihin liittyvästä tapahtumasta, ne eivät enää liity toisiinsa.

2.5 WordNet

WordNet on etenkin tutkimusmaailmassa paljon käytetty englannin kielen sanojen semanttisten suhteiden tietokanta. Myös useille muille kielille ollaan laatimassa WordNet-tyyppistä tietokantaa ja EuroWordNet -projektissa useille Euroopan kielille laadittiin WordNet-tyyppinen tietokanta [Vos98]. Alkuperäinen WordNet on kehitetty Yhdysvalloissa Princetonin yliopiston kognitiotieteen laitoksella ja työ aloitettiin vuonna 1985 [Fel98c]. WordNet on vapaasti saatavilla tutkimuskäyttöön ja se on myös käytettävissä WWW-lomakkeen kautta Princetonin yliopiston kognitiotieteen laitoksen WWW-sivuilla [Wor01]. Seuraavaksi esitellään WordNetin rakenne ja perusperiaatteet yleisesti ja luvussa 2.5.1 tutustutaan WordNetin kolmeen tietokantaan lähemmin.

WordNetin kehittämisessä oli päämääränä luoda sellainen sanojen semanttisten suhteiden tietokanta, joka paitsi kuvaisi semanttisten suhteiden järjestäytymistä ihmisen mielessä, olisi myös käyttökelpoinen tietokoneohjelmistoille. Aikaisemmin olemassa olevat sanakirjat ja sanastot oli suunniteltu ihmisen luettaviksi eivätkä sen takia sellaisenaan soveltuneet tietokoneohjelman käytettäviksi [Mil95].

WordNetin yhteydessä ei oikeastaan voida puhua sanoista siinä mielessä kuin sanan käsite on luvussa 2.1 määritelty ja sen takia tässä tutkielmassa käytetään termiä *perusmuoto* WordNetiin talletetuista yksiköistä. WordNetin perusmuoto voi muodostua useammasta kuin yhdestä sanasta, esim. *bad person, atom bomb, interpretive dancing* ja *break down*. WordNetin perusmuoto ei ole sama kuin lekseemi, sillä homonyymit ovat WordNetissä yksi ja sama perusmuoto, jos ne vain kuuluvat samaan tietokantaan, esim. *bass* (basso) ja *bass* (ahven). WordNet ei siis tee eroa homonymian ja polysemian välillä.

WordNetin semanttiset suhteet vallitsevat *perusmuotojen* välillä, *perusmuotojen yksittäisten merkitysten* välillä tai *synonyymijoukkojen* välillä. Kaikki WordNetissä esiintyvät semanttiset suhteet on esitetty taulukossa 1. Jokainen synonyymijoukko, *synset*, kuvaa ihmisen leksikaalisessa muistissa olevaa käsitettä. Taulukossa 2 on esitetty sananmuodolla *dog* substantiivitietokannasta löytyvistä kuudesta merkityksestä kolme ensimmäistä. Siitä käy hyvin ilmi, mitä WordNetissä tarkoitetaan sananmuodon yksittäisellä merkityksellä ja synonyymijoukolla. Taulukossa 2 esiintyvät synonyymijoukot ovat: { *dog, domestic dog, Canis familiaris* }, { *fromp, dog* } ja { *dog* }. Sananmuodon eri merkitykset on WordNetissä järjestetty yleisyyden perusteella siten, että useimmin esiintyvä on ensimmäisenä ja harvimmin esiintyvä viimeisenä.

Suhde	Tietokanta	Esimerkki
Synonymia	N, V, Adj&Adv	sad, unhappy
Antonymia	Adj&Adv, (N, V)	sad, glad (sell, buy)
Hyponymia hyperonyymi hyponyymi	N	 silver maple is a kind of *maple* *silver maple* is a kind of maple
Meronymia holonyymi meronyymi	N	 ship is a member of a *fleet* *ship* is a member of a fleet
Troponymia troponyymi	V	 to *whisper* is a particular way to speak
Implikaatio (engl. entailment)	V	 to win entails *competing*

Taulukko 1: *WordNetin semanttiset suhteet. N = substantiivit, V = verbit, Adj&Adv = adjektiivit ja adverbit.*

1. *dog*, domestic dog, Canis familiaris — (a member of the genus Canis (probably descended from the common wolf) that has been domesticated by man since prehistoric times; occurs in many breeds; “the dog barked all night”);
2. fromp, *dog* — (a dull unattractive unpleasant girl or woman; “she got a reputaion as a fromp”; “she’s a real dog”)
3. *dog* — (informal term for a man: “you lucky dog”)

Taulukko 2: *WordNetin substantiivitietokannan hakusanan dog kolme ensimmäistä merkitystä.*

WordNet sisältää yli 166 000 yksittäistä sananmuodon merkitystä, yli 118 000 perusmuotoa ja yli 90 000 merkitystä eli synonyymijoukkoa. Noin 17 %:lla WordNetin sananmuodoista on useampi kuin yksi merkitys ja noin 40 %:lla on yksi tai useampi synonyymi [Mil95]. WordNetissä on yhteensä 116 000 semanttista suhdetta osoittavaa viittausta perusmuotojen, yksittäisten perusmuotojen merkitysten tai synonyymijoukkojen välillä.

2.5.1 Substantiivi-, verbi- ja adjektiivi-adverbitietokannat

WordNet koostuu kolmesta erillisestä tietokannasta. Yhdessä ovat substantiivit, yhdessä verbit ja yhdessä sekä adjektiivit että adverbit. Tietokannassa on siis edustettuna kaikki *avoimet sanaluokat* eli sellaiset sanaluokat, joihin helposti syntyy uusia sanoja. *Sulkeisiin sanaluokkiin*, joihin syntyy uusia sanoja hyvin harvoin, kuuluvia sanoja ei WordNetissä ole ollenkaan. Sulkeisia sanaluokkia ovat pronominit, pre- ja postpositiot, artikkelit ja konjunktiot. Jos jokin perusmuoto kuuluu useampaan eri sanaluokkaan, on se erikseen tallennettu kaikkiin niihin tietokantoihin, joihin se sanaluokkansa perusteella kuuluu. Seuraavaksi kuvataan tarkemmin kutakin tietokantaa.

Kaikki WordNetissä olevat substantiivit on järjestetty hierarkisesti hyponymian

mukaan. Puun juurisolmu on *{entity}* ja sillä on 25 lapsisolmua, jotka on lueteltu taulukossa 3. Juurisolmun ja sen lapsisolmujen väliin on harkittu hierarkisempaakin rakennetta, mutta silloin olisivat puun yläosan solmut kuvanneet niin abstrakteja ja yleisluontoisia käsitteitä, että ne tuskin olisivat kuvanneet hyvin substantiivien järjestäytymistä ihmisen mielessä [Mil98]. Vaikka hierarkia puun juuresta yksittäiseen lehtisolmuun voisi teoriassa olla kuinka monikerroksinen tahansa, on se vain harvoin yli kymmenen tasoa.

{act, action, activity}	{animal, fauna}	{artifact}
{attribute, property}	{body, corpus}	{cognition, knowledge}
{communication}	{event, happening}	{feeling, emotion}
{food}	{group, collection}	{location, place}
{motive}	{natural object}	{natural phenomenon}
{person, human being}	{plant, flora}	{possession}
{process}	{quantity, amount}	{relation}
{shape}	{state, condition}	{substance}
{time}		

Taulukko 3: *WordNetin substantiivihierarkian juurisolmun { entity } lapsisolmut.*

Verbit on WordNetissä järjestetty pääosin verbiluokkien, merkityksen ja leksikaalisten implikaatiosuhteiden (engl. entailment relations) mukaan. Verbien väliset suhteet ja verbien järjestys WordNetissä ovat monimutkaisemmat kuin substantiivien, adjektiivien ja adverbien keskinäiset suhteet ja järjestys. Tyypillistä verbihierarkioille on, että ne ovat suhteellisen matalia, sillä niiden syvyys ylittää harvoin neljä tasoa.

Verbiluokkien perusteella verbit on jaettu siten, että ensin kaikki verbit on jaettu kahteen luokkaan: tapahtumiin ja toimintaan (engl. events and actions) sekä tiloihin (engl. states). Sen jälkeen tapahtuma ja toiminta -luokkaan kuuluvat verbit

on edelleen jaettu neljääntoista eri luokkaan [Fel98b]. Niitä vastaavat verbihierarkiapuun solmut on lueteltu taulukossa 4.

bodily care and functions	change	cognition
communication	competition	consumption
contact	creation	emotion
motion	perception	possession
social interaction	weather	

Taulukko 4: *WordNetin verbihierarkian tapahtuma ja toiminta -solmun lapsisolmut.*

Tila-luokkaan luokitellut verbit eivät muodosta mitään yhtenäistä semanttista aluetta, vaan niitä yhdistää se, että ne eivät kuulu mihinkään tapahtuma ja toiminta -luokan neljästätoista aliluokasta. Esimerkkejä tila-luokan verbeistä ovat *suffice*, *belong* ja *resemble*.

Merkityksensä perusteella verbit on järjestetty hierarkkisesti siten, että hierarkian ylimmällä tasolla on juuriverbejä, eli sellaisia verbejä, joiden merkitys on osa alemmilla tasoilla olevien verbien merkityksestä. Juuriverbejä ovat esimerkiksi {move, go} ja {change}, koska ne kuvaavat peruskäsitteitä. Esimerkikisi *walk* ja *run* sisältävät juuriverbien {move, go} merkitykset.

Kolmas tapa järjestää verbejä ja kuvata niiden välisiä suhteita on leksikaalinen implikaatio, jolla tarkoitetaan, että V_1:stä seuraa V_2. Leksikaalinen implikaatio on sama kuin propositiologiikan implikaatio $P \longrightarrow Q$. Esimerkki: *kuorsaaminen* implikoi *nukkumista*. Jos henkilö kuorsaa, niin siitä voidaan päätellä, että hän nukkuu. Leksikaalinen implikaatiosuhde on edelleen jaettu neljään osaan: troponymiaan, oletukseen ajassa taaksepäin (engl. backward presupposition), syyseuraukseen (engl. cause) ja muihin samanaikaisuutta sisältäviin implikaatiosuhteisiin. Nämä suhteet on esitetty kuvassa 1. Seuraavaksi kuvataan tarkemmin kutakin implikaatiosuhdetta.

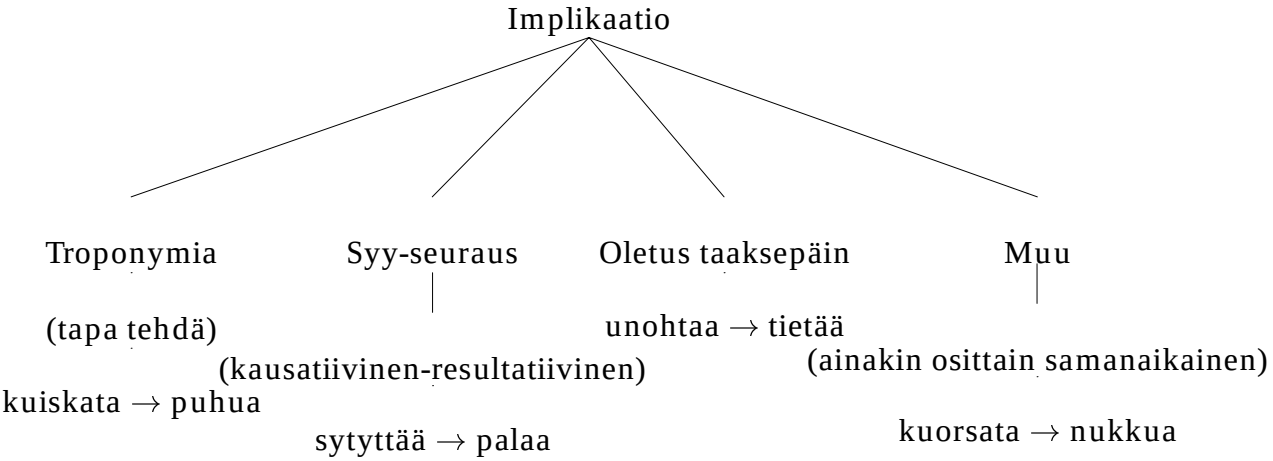
Verbien V_1 ja V_2 välinen implikaatiosuhde on

Troponymia jos V_1 ilmaisee tapaa tehdä V_2. Esimerkki: V_1 = nilkuttaa ja V_2 = kävellä.

Oletus ajassa taaksepäin jos V_1 ei voi tapahtua ilman että ennen sitä on tapahtunut V_2. Esimerkki: V_1 = palata takaisin ja V_2 = lähteä pois.

Syy-seuraus jos V_1 on kausatiivinen (engl. causative) ja V_2 on resultatiivinen (engl. resultative) verbi. Esimerkki: V_1 = rikkoa ja V_2 = mennä rikki.

Muu samanaikaisuutta sisältävä implikaatiosuhde jos V_1 ja V_1 tapahtuvat ainakin osittain samanaikaisesti eikä niiden välinen implikaatiosuhde kuulu mihinkään yllä mainituista suhteista. Esimerkki: V_1 = ostaa ja V_2 = maksaa.



Kuva 1: *Verbien leksikaalisten implikaatiosuhteiden luokittelu.*

Adjektiivit on WordNetissä jaettu kolmeen kategoriaan: kuvaileviin (engl. descriptive), suhdetta ilmaiseviin (engl. relational) ja väriadjektiiveihin. Kuvailevat adjektiivit muuntavat pääsanana toimivan substantiivin merkitystä. Ne on WordNetissä järjestetty antonymiaan ja synonymiaan perustuviin rypäisiin. Suhdetta ilmaisevat adjektiivit, jotka siis käyttäytyvät samalla tavoin kuin määritteenä toimivat substantiivit, on WordNetissä yhdistetty johonkin substantiiviin. Esim.

*dental hygiene*ssä *dental* on suhteessa sanaan *tooth* ja *fraternal twins*issä *fraternal* on suhteessa sanaan *brother* [GM98]. Värejä ilmaisevat sanat on WordNetissä kuvattu eri tavalla kuin muut adjektiivit, sillä värit eroavat adjektiiveista merkitykseltään siten, että niiden voidaan katsoa edustavan jatkumoa.

3 Tärkeiden sanojen tunnistaminen morfosyntaktisten piirteiden avulla

Lauseen sanojen morfologisten ja syntaktisten ominaisuuksien tutkiminen on hyödyllistä lauseen merkityksen selvittämisessä vaikka ei pyrittäisikään lauseen ymmärtämiseen. Sanojen morfosyntaktiset piirteet voivat auttaa hahmottamaan, mitkä sanat ovat merkityksen kannalta avainasemassa. Tästä esimerkkinä mainittakoon, että monet semanttista samanlaisuutta leksikaalisten ketjujen avulla laskevat menetelmät pitävät ainoastaan substantiiveja tekstin merkityksen kannalta oleellisina sanoina [BE97, HSO98]. Toisaalta, jos samanlaisuutta lasketaan vektorimallin avulla, voidaan painottaa niitä sanoja, joilla morfosyntaktisten piirteiden perusteella päätellään olevan enemmän painoarvoa [Lah00].

Tässä tutkielmassa sanan *morfosyntaktisilla piirteillä* tarkoitetaan sanan morfeemeita, sanaluokkaa ja syntaktista funktiota tietyssä kontekstissa. Luvussa 3.1 käydään läpi aiheen kannalta oleellisia käsitteitä ja luvussa 3.2 esitellään funktionaalinen dependenssikielioppi, johon perustuvaa jäsentäjää on käytetty tämän tutkielman kokeellisessa osassa. Luvussa 3.3 pohditaan, millaisia morfosyntaktisia piirteitä omaavat sanat voivat olla merkityksen kannalta tärkeitä sekä esitellään menetelmä näille sanoille ominaisten morfosyntaktisten piirteiden löytämiseksi.

3.1 Morfeemi, konstituentti ja syntaktinen funktio

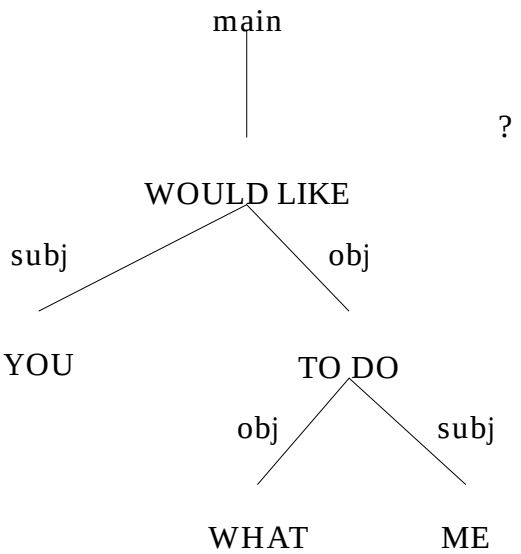
Morfeemi on abstraktio, joka edustaa useita samaan kategoriaan kuuluvia morfeja. *Morfi* puolestaan on kielen pienin rajattavissa oleva yksikkö, jolla on merkitys tai kieliopillinen funktio. Kaksi morfia voivat olla saman morfeemin ilmentymiä jos niillä on sama merkitys tai kieliopillinen funktio [Kar98]. Esim. sana *padat* koostuu kahdesta morfista: *pada* ja *t*, jotka voidaan esittää morfeemeina *pata* ja PL ja sana *padoissa* koostuu kolmesta morfista: *pado*, *i* ja *ssa*, jotka voidaan esittää morfeemeina *pata*, PL ja INESSIIVI.

Syntaktisten funktioiden määrittelemiseksi pitää ensin määritellä käsite *konstituentti* eli muodostin eli lauseke. Jokainen lausessa esiintyvä peräkkäisten sanojen ryhmä, joka voidaan irroittaa ja korvata yhdellä sanalla, on konstituentti. Esim. lauseen *Pekka sai sievän kissan*. konstituentit ovat: *Pekka*, *sai sievän kissan*, *sai*, *sievän kissan*, *sievän*, *kissan*. Konstituentit voivat olla esimerkiksi substantiivi-, verbi-, adjektiivi, adverbiaali-, pre- ja postpositiolausekkeita [Kar98].

Syntaktiset funktiot ilmaisevat lauseen konstituenttien välisiä suhteita tai niiden rooleja lauseessa. Se, kuinka monta syntaktista funktiota on olemassa ja miten syntaktisen funktion käsite määritellään, riippuu siitä, mitä syntaktista teoriaa tarkastelun pohjana käytetään [Kar98]. Tärkeimmät perinteiset syntaktiset funktiot ovat: subjekti, predikaatti, objekti, attribuutti, predikatiivi, adverbiaali ja predikatiiviadverbiaali. Seuraavassa luvussa esitetyt syntaktiset funktiot on kuvattu dependenssikieliopin mukaan. Tutkielmassa käytetyt funktionaalisen dependenssikieliopin mukaiset syntaktiset funktiot on lueteltu liitteessä 3. Ne on jaettu pintasyntaktisiin funktioihin ja funktionaalisiin riippuvuuskuvaimiin.

3.2 Funktionaaliseen dependenssikielioppiin perustuva jäsennin

Funktionaaliseen dependenssikielioppiin perustuva jäsennin eli FDG-jäsennin on ohjelma, joka saa syötteekseen luonnollista kieltä ja palauttaa funktionaalisen dependenssikieliopin mukaisen analyysin saamalleen syötteelle. FDG-jäsennin käyttää lauseen analysointiin sekä säännöllisiin lausekkeisiin perustuvaa mallia että puumallia, joka kuvaa lauseessa olevien ydinten (engl. nuclei) välisiä riippuvuussuhteita [Tap99]. Säännöllisiin lausekkeisiin perustuvan osan tehtävänä on suorittaa pintasyntaktista analyysia siten, että se karsii sanojen ylimääräisiä luentoja. FDG-jäsentimen esittelivät ensimmäisinä Tapanainen ja Järvinen [TJ97]. Säännöllisiin lausekkeisiin perustuvia jäsentimiä on ollut olemassa jo ennen FDG-jäsennintä, mutta lauseen elementtien välisiä syntaktisia riippuvuussuhteita dependenssikieliopin mukaan kuvaavaa formalisimia ei ole toteutettu ennen sitä. FDG-jäsennin perustuu strukturaalikielitieteilijän *Lucien Tesnière*n dependenssikielioppiin.



Kuva 2: *Lauseen "What would you like me to do?"dependenssipuu [Tap99].*

Dependenssikieliopin syntaktisen rakenteen peruselementti on *ydin*. Se koostuu

yhdestä tai useammasta syntaktisesti ja semanttisesti toisiinsa liittyvästä alkiosta. Ytimen sanojen ei tarvitse olla lauseessa peräkkäin. Lauseen ytimet voidaan järjestää puuksi, jonka juuri on lauseen pääverbi, sillä jokaisella ytimellä on vain yksi isäsolmu. Yhdellä solmulla voi olla useita lapsisolmuja [Tap99]. Kuvassa 2 on esimerkki dependenssipuusta. Lauseen ytimet ovat puun solmuja ja nimetyt kaaret kuvaavat solmujen välisiä suhteita.

FDG-jäsennin koostuu seuraavista komponenteista [Tap99]:

Selaaja (engl. tokenizer) suorittaa kielen alkioanalyysin. Alkioita voivat olla esim. sanat, monisanaiset yksiköt ja numeroita sisältävät ilmaisut, esimerkki:

- Syöte: Mitä haluat minun tekevän Puerto Ricossa 12.10.?
- Tuloste: Mitä, haluat, minun, tekevän, Puerto Ricossa, 12.10., ?

Leksikaalinen analysoija lisää jokaiseen alkioon tunnisteita, jotka ilmaisevat alkion mahdollisia sanaluokkia ja morfeemeita, esimerkki taulukossa 5.

Morfosyntaktinen yksiselitteistäjä (engl. disambiguator) poistaa aikaisemmassa vaiheessa alkioihin lisätyt ylimääräiset tunnisteet, esimerkki taulukossa 6.

Dependenssianalysoija muodostaa lauseesta puun, joka kuvaa lauseen syntaktisten elementtien välisiä suhteita, esimerkki taulukossa 7.

Taulukkojen 6 ja 7 analyyseissa käytetyt tunnisteet (engl. tag) ja muut merkinnät on selvitetty liitteessä 3. Se on sama merkkaustapa, jota tutkielman luvussa 6 kuvatussa kysymysvastauskorpuksen analyysissä on käytetty.

Syöte:	kuusi
Tuloste:	
"kuu"	N NOM SG 2SG
"kuu"	N GEN SG 2SG
"kuu"	N NOM PL 2SG
"kuusi"	N NOM SG
"kuusi"	PROP N NOM SG
"kuusi"	NUM NOM SG

Taulukko 5: *Esimerkki FDG:n leksikaalisesta analyysista.*

Syöte:	What would you like me to do?
Tuloste:	
“what”	@OBJ PRON
“would”	@+FAUX V
“you”	@SUBJ PRON
“like”	@-FMAIN V
“i”	@ OBJ PRON
“to”	@INFMARK>INFMARK>
“do”	@-FMAINV V

Taulukko 6: *Esimerkki FDG:n morfosyntaktisesen yksiselitteistäjän tulostuksesta [Tap99].*

Id	Alkuperäinen syöte	Analyysi
1	What	what obj:>7 @OBJ %NH PRON WH
2	would	would v-ch:>4 @+FAUXV %AUX V AUXMOD
3	you	you subj:>2 @SUBJ %NH PRON PERS NOM
4	like	like main:>0 @-FMAINV %VA V INF
5	me	i subj:>7 @OBJ %NH PRON PERS ACC SG1
6	to	to pm:>7 @INFMARK> %AUX INFMARK>
7	do	do obj:>4 @-FMAINV %VA V INF
8	?	?

Taulukko 7: *Esimerkki FDG:n dependenssijäsentimen tulostuksesta [Con01a].*

3.3 Termien painottaminen lingvististen tunnisteiden perusteella

Merkityksen kannalta oleellisten sanojen tunnistaminen ja painottaminen on tarpeellista kun verrataan tekstifragmenttien välistä samanlaisuutta. Tärkeiden sanojen painottamiseksi on kehitetty useita eri menetelmiä. Tässä luvussa käsitellään sanojen morfosyntaktisiin ominaisuuksiin perustuvaa menetelmää ja luvussa 4.1.2 tilastollisiin mittoihin perustuvia menetelmiä. Tämän luvun menetelmä olettaa, että käytettävissä on tekstikorpus, johon sisällön kannalta oleelliset sanat on merkitty.

Lahtinen [Lah00] on väitöskirjassaan etsinyt indeksointitermeille tyypillisiä tunnistejonoja (engl. tag combinations) ja laskenut niille tunnistepainoja (engl. tag weight). Tunnisteet tarkoittavat tässä FDG-jäsentimen tuottamia morfosyntaktisia tunnisteita ja sanojen sijaintia tekstissä ilmaisevaa merkkausta, esim. onko sana otsikossa tai kappaleen ensimmäisessä virkkeessä. Indeksointitermit ovat tekstin sisältöä kuvaavia termejä, joita käytetään tiedonhaussa. Tässä indeksointitermeinä on käytetty kunkin dokumentin lopussa olevassa hakemistossa esiintyviä termejä.

Korpuksena tutkimuksessa on käytetty englanninkielisiä sosiologian ja filosofian alaan kuuluvia tekstejä sekä kymmentä *The Grolier Encyclopedian* artikkelia. Koko korpuksessa on yhteensä 64 996 sanaa. Ensin korpuksen sanat on automattisesti merkattu FDG-jäsentimen syntaktisin tunnistein ja sijaintia osoittavin tunnistein. Sen jälkeen tekstiin on manuaalisesti merkattu kaikki indeksointitermit. Merkkauksen jälkeen korpus on jaettu kahteen osaan, joista toista eli opetuskorpusta on käytetty indeksointitermeille todennäköisten tunnisteiden löytämiseen ja tunnistepainojen laskemiseen ja toista eli testikorpusta tulosten testaamiseen.

Indeksointitermeille tyypillisten tunnistejonojen löytäminen tehdään käyttäen ling-

vististä tietämystä ja laskemalla tunnistepainoja opetuskorpuksesta. Tunnistepainot on muodostettu laskemalla ehdollinen todennäköisyys sille, että mielivaltainen termi t on indeksointitermi sillä ehdolla, että siihen liittyy tunnistejono tn. Jos ehdollinen todennäköisyys on suuri, kyseessä on indeksointitermille tyypillinen tunnistejono.

Esimerkki: Lasketaan todennäköisyys sille, että mielivaltainen termi *t* on indeksointitermi kun tiedetään, että siihen liittyvä tunnistejono tn on $< Proper > N\ NOM\ SG\ @SUBJ\ subj\ :>$. Kyseinen tunnistejono esiintyy korpuksessa yhteensä 43 kertaa ja 42:ssa tapauksessa kyseessä on indeksointitermi. Todennäköisyys lasketaan seuraavasti:

$$P(t \in indeksointitermit|tn) = \frac{P(t \in indeksointitermit\ \cap tn)}{P(tn)} = \frac{42}{43} = 0.977$$

Tutkimuksessa löydettiin yhteensä 89 erilaista indeksointitermien kannalta oleellista tunnistejonoa. Tutkimuksen indeksointitermit olivat sekä yksi- että monisanaisia ja tunnistejonot koostuivat tunnisteiden lisäksi operaattoreista *AND*, *OR* ja *NOT*. Taulukossa 8 on esimerkki tunnistejonosta.

tunnistejono: N AND subj: AND (<-> OR <?> OR <*>) AND NOT (<Proper> OR PL OR <DER:ism>)
paino: 0.654

Taulukko 8: *Esimerkki indeksointitermeille yleisestä tunnistejonosta. Esimerkki tarkoittaa, että yleisnimillä (N AND NOT <Proper>), jotka ovat subjekteja (subj:) ja joilla on jokin tunnisteista <->, <?> tai <*> ja jotka eivät ole monikkomuotoja (PL) ja jotka eivät pääty -ism (<DER:ism>) on paino 0.654.*

Kun kullekin tunnistejonolle on laskettu painoarvo, voidaan termin painoarvo laskea siten, että lasketaan yhteen kaikkien sen esiintymien tunnistejonojen pai-

noarvot. Näin laskettua termin painoarvoa kutsutaan STW:ksi eli summed tag weightiksi [Lah00]. Mitä suurempi termin STW on, sitä merkityksellisempänä tekstin sisällön kannalta sitä pidetään.

4 Samanlaisuuden laskeminen

Tässä tutkielmassa käsitellään ainoastaan numeeriseen laskentaan perustuvia menetelmiä tekstin samanlaisuuden toteamiseksi. Tutkielman ulkopuolelle jäävät mm. sellaiset symboliset menetelmät, joissa pyritään päättelysääntöjen avulla muodostamaan kuvaus tekstin merkityksestä. Luvussa 4.1 esitellään vektorimalli ja luvussa 4.2 leksikaaliset ketjut. Vektorimallin lisäksi muita klassisia tiedonhaussa käytettyjä samanlaisuuden laskemismalleja ovat boolean-malli ja probabilistinen malli [BYRN99]. Niitä ei kuitenkaan esitellä tässä, sillä vektorimallia pidetään tällä hetkellä tehokkaimpana hakumallina [BYRN99].

4.1 Vektorimalli

Semanttisen samanlaisuuden havaitseminen vektorimallin avulla perustuu siihen, että tarkastelun kohteena oleva teksti tai sen osa kuvataan vektorien avulla ja vektorien etäisyys toisiinsa lasketaan jollakin mitalla. Sovelluksesta riippuen tekstit voivat olla osittain päällekkäisiä tai täysin erillisiä ja niiden pituus voi eri sovelluksissa vaihdella paljonkin. Vektorimalli on yleisesti käytössä tiedonhaun sovelluksissa, sillä se on yksinkertainen, se toimii melko hyvin eikä se ole laskennallisesti raskas [BYRN99]. Esityksen selkeyden vuoksi käytetään tässä luvussa tekstistä ja tekstifragmentista nimitystä dokumentti.

Perusajatus vektorimallissa on, että kustakin dokumentista muodostetaan piirrevektori, joka kuvaa kaikkien siinä esiintyvien termien painoarvot *n*-ulotteisessa

termiavaruudessa. *Termillä* tarkoitetaan jokaista sanaa tai jokaista useampisanaista fraasia, joka esiintyy tekstissä. Piirrevektori voidaan esittää kaavalla

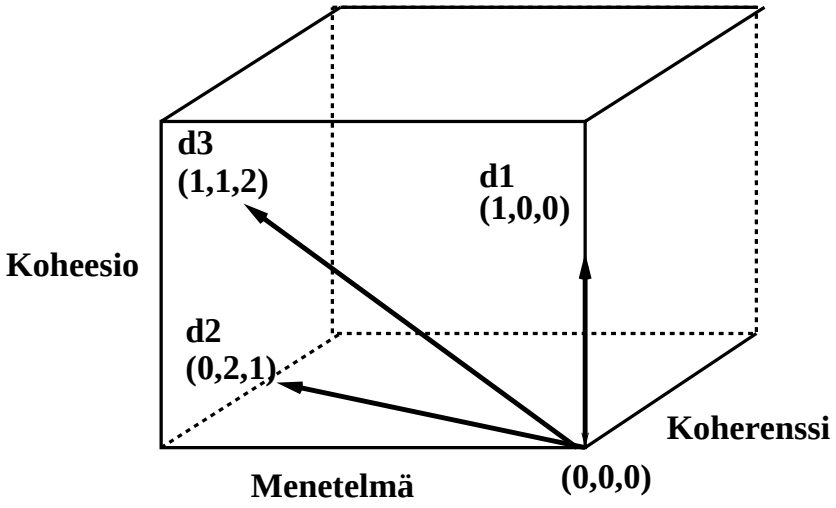
$$\vec{d_j} = (w_{1,j}, w_{2,j}, w_{3,j}, \ldots, w_{n,j}), \tag{1}$$

missä $\vec{d_j}$ kuvaa tiettyä dokumenttia j ja kukin piirre $w_{i,j}$ kuvaa termin i painoa dokumentissa j. Koko dokumenttikokoelman dokumentit voidaan kuvata painomatriisina, jonka rivit esittävät dokumenttikokoelmassa esiintyviä termejä ja sarakkeet dokumentteja. Taulukossa 9 on tästä esimerkki. Kuvassa 3 taulukon vektorit on esitetty kolmiulotteisessa vektoriavaruudessa.

$$\mathbf{A} = \left(\begin{array}{l|ccc} & \text{d1} & \text{d2} & \text{d3} \\ \hline koheesio & 1 & 0 & 1 \\ menetelmä & 0 & 2 & 1 \\ koherenssi & 0 & 1 & 2 \end{array}\right)$$

Taulukko 9: *Jokin tämän tutkielman luvuista voisi näyttää tältä jos se kuvattaisiin termeistä koheesio, menetelmä ja koherenssi muodostuvassa kolmiuloitteisessa vektoriavaruudessa. Termien painot on muodostettu perusmuotoon palautettujen sanojen frekvenssien perusteella.*

Yllä esitellyissä vektoreissa kukin vektorin ulottuvuus voi saada arvokseen jonkin reaaliluvun. Dokumenttien kuvaamiseen voidaan käyttää myös sellaista vektoriavaruutta, jossa kukin ulottuvuus voi saada vain arvot 1 tai 0 [MS00]. Yllä esiteltyihin vektoreihin verrattuna tällainen vain kaksi mahdollista painoarvoa sisältävä vektori eli *binäärivektori* on helpompi kuvata ja sille suoritettavat laskutoimitukset ovat yksinkertaisempia. Yksinkertaisin tapa kuvata binäärivektori on esittää se niiden ulottuvuuksien joukkona, joiden arvo ei ole 0. Esimerkiksi kuvan 3 dokumenttivektorit voidaan kuvata seuraavina joukkoina: dokumentti 1:



Kuva 3: *Taulukon 9 dokumentit esitettyinä kolmiulotteisessa vektoriavaruudessa.*

{koheesio}, dokumentti 2: *{menetelmä, koherenssi}*, dokumentti 3: *{koheesio, menetelmä, koherenssi}*.

4.1.1 Dokumenttivektoreiden välisen samanlaisuuden laskeminen kosinimitalla

Vektoreiden välisen samanlaisuuden laskemiseksi on olemassa useita erilaisia mittoja, mutta koska kosinimitta on niistä ylivoimaisesti yleisimmin käytetty käsitellään tässä ainoastaan sitä [MS00]. Muita vektoreiden välisen samanlaisuuden laskemiseksi käyettyjä mittoja ovat esim. Dicen ja Jaccardin mitat [vR80].

Binäärivektoreiden samanlaisuus voidaan laskea käyttämällä joukko-operaatioita, sillä kuten edellisessä luvussa todettiin, voidaan binäärivektorit kuvata joukkoina. Joukkojen X ja Y välinen samanlaisuus sim lasketaan kosinimitan avulla kaavalla

$$sim(X,Y) = \frac{|X \cap Y|}{\sqrt{|X||Y|}}, \tag{2}$$

missä $|X \cap Y|$ on niiden alkioiden lukumäärä, jotka kuuluvat sekä joukkoon X että joukkoon Y, $|X|$ on joukon X alkioiden lukumäärä ja $|Y|$ on joukon Y alkioiden lukumäärä.

Reaalilukuvektoreiden välinen samanlaisuus lasketaan kosinimitalla siten, että ensin lasketaan kahden vektorin välinen pistetulo ja sitten jaetaan se vektoreiden pituuksien tulolla kuten esitetään kaavassa

$$sim(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{|\vec{x}||\vec{y}|}. \tag{3}$$

Vektoreiden välinen pistetulo lasketaan kaavalla

$$\vec{x} \cdot \vec{y} = \sum_{i=1}^{N} x_i y_i, \tag{4}$$

missä N on vektoriavaruuden ulottuvuuksien lukumäärä ja x_i on vektorin i:nnen ulottuvuuden arvo. Vektorin pituus puolestaan lasketaan seuraavasti:

$$|\vec{x}| = \sqrt{\sum_{i=1}^{N} x_i^2}. \tag{5}$$

Toinen tapa laskea kahden vektorin välinen samanlaisuus kosinimitan avulla on, että ensin normalisoidaan vektorit, jolloin pelkän pistetulon laskeminen riittää. Vektorin normalisoiminen tehdään siten, että jaetaan jokainen vektorin ulottuvuus koko vektorin pituudella. Edellisen luvun taulukossa 9 esitettyjen vektorien pituudet ovat: $d1$: 1, $d2$: $\sqrt{5}$ ja $d3$: $\sqrt{6}$ ja vastaavasti samat dokumentit normalisoituina painomatriisin avulla esitettyinä löytyvät taulukosta 10.

Kosinimitalla laskettu dokumenttivektoreiden välinen samanlaisuusarvo vaihtelee välillä [0, 1]. Jos se on 0, ovat dokumenttivektorit mahdollisimman erilaiset ja jos se on 1, ne ovat täysin samanlaiset. Tämä perustuu siihen, että kun vektorien välinen kulma on 0 astetta - jolloin vektorit siis ovat samansuuntaiset - on kul-

$$\mathbf{A} = \left(\begin{array}{l|ccc} & \text{d1} & \text{d2} & \text{d3} \\ \hline koheesio & 1 & 0 & .41 \\ menetelmä & 0 & .89 & .41 \\ koherenssi & 0 & .45 & .82 \end{array}\right)$$

Taulukko 10: *Taulukon 9 painomatriisi normalisoituna.*

man kosini 1. Vastaavasti kun vektorien välinen kulma on 90 astetta, on kulman kosinin arvo 0.

	$d1$	$d2$	$d3$
d1	1	0	.41
d2	0	1	.73
d3	.41	.73	1

Taulukko 11: *Taulukon 9 dokumenttien väliset samanlaisuusarvot laskettuna kosinimitalla.*

4.1.2 Termien valitseminen ja painottaminen

Termien valitsemisella ja painottamisella on ratkaiseva vaikutus dokumenttien välisen samanlaisuuden tunnistamisessa. Kuvan 3 esimerkissä reaalilukuvektorien painotukseen käytettiin termien frekvenssiä dokumentissa eli tf:ia (engl. term frequency). Itse termit oli valittu täysin mielivaltaisesti dokumenttien perusmuotoon palautettujen sanojen joukosta.

Yksinkertainen kaava termien painottamiseksi on $tf * idf$, missä idf tarkoittaa käänteistä dokumenttifrekvenssiä (engl. inverse document frequency). Kaavassa otetaan termifrekvenssin lisäksi huomioon termien esiintyvyys koko dokumenttikokoelmassa. Intuitiivisesti voidaan ajatella, että sellaisia termejä, jotka esiinty-

vät usein koko tekstissä pitäisi painottaa vähemmän kuin sellaisia, jotka esiintyvät usein ainoastaan käsillä olevassa dokumentissa. Tällainen mitta on

$$idf_i = log\left(\frac{N}{n_i}\right),$$

missä N on kaikkien dokumenttien lukumäärä ja n_i on niiden dokumenttien lukumäärä, joissa termi *i* esiintyy. Koska kokoelmassa on usein suuri määrä dokumentteja, otetaan arvojen tasoittamiseksi saadusta luvusta vielä logaritmi.

Termin i paino w dokumentissa j lasketaan $tf * idf$-mitalla seuraavasti:

$$w_{i,j} = tf_{i,j} * idf_i,$$

missä $tf_{i,j}$ on termin i frekvenssi dokumentissa j ja idf_i on termin i käänteinen dokumenttifrekvenssi.

$tf * idf$-painotusmallista on kehitetty lukuisia erilaisia muunnelmia, joissa termifrekvenssi, kokoelmafrekvenssi (esim. idf) ja dokumentin pituuden normalisointi on suoritettu eri tavoilla tai ne on jätetty kokonaan pois. Salton ja Buckley ovat evaluoineet 1800 erilaista muunnelmaa, joista 287 antoivat erilaisen hakutuloksen [SB87]. Tutkimuksessaan he käyttivät termeinä ainoastaan yhdestä sanasta koostuvia yksikköjä. Silloin, kun sekä kyselyt että vastaukset ovat lyhyitä ja sama termi esiintyy vain kerran dokumentissa, mikä on tilanne tämän tutkielman kysymysvastausjärjestelmässä, on Saltonin ja Buckleyn tutkimuksen mukaan suositeltavaa käyttää seuraavanlaista termien painotusta ja vektoreiden normalisointia: termifrekvensseinä käytetään binäärisiä arvoja (1 jos termi esiintyy kyselyssä tai dokumentissa, muuten 0), kokoelmafrekvenssinä tulisi käyttää idf:ia ja vektoreita ei tulisi normalisoida.

Termivektoriin otetaan harvoin mukaan kaikkia dokumenttikokoelmassa esiintyviä sanoja sellaisinaan, vaan termien määrää ja samalla vektoriavaruuden ulottuvuuksien määrää pyritään pienentämään jollakin tavalla. Yleistä on, että sanat

palautetaan perusmuotoonsa. Lisäksi vektorista poistetaan kielen kaikkein yleisimmät sanat, sillä ne eivät tuo mitään uutta tietoa kyseessä olevan dokumentin aihealueesta. Tällaisia sanoja ovat suomen kielessä esim. *ja, tai* ja *on* ja ne sisältävää sanalistaa kutsutaan sulkulistaksi (engl. stop word list). Lisäksi sulkulistalle otetaan usein mukaan myös ne sanat, jotka esiintyvät jokaisessa tai ainakin lähes jokaisessa dokumenttikokoelman dokumentissa.

Yllä mainittujen perustoimenpiteiden lisäksi voidaan termiavaruutta pienentää ottamalla mukaan vain merkitykselliset sanat, jotka voidaan määrätä sanaluokan tai sanafrekvenssin mukaan esimerkiksi jättämällä pois ne sanat, jotka esiintyvät vain kerran koko dokumenttikokoelmassa. Termien vähentämiseen voi käyttää myös latenttia semanttista indeksointia, joka perustuu sille, että usein yhdessä esiintyvät termit kuvataan saman ulottuvuuden avulla. Menetelmässä käytetään pääkomponenttianalyysia [MS00]. Vektoriavaruuden ulottuvuuksien vähentäminen on aihe, jota on tutkittu paljon. Esimerkkejä tehtävän suorittamisessa käytetyistä mitoista ovat assosiaatiokerroin (engl. association factor), informaation lisäys (engl. information gain), mutual information ja χ-neliö (engl. CHI-square) [Seb02, YP97].

4.2 Leksikaaliset ketjut

Leksikaaliset ketjut (engl. lexical chains) ovat toisiinsa kohesiivisten suhteiden kautta liittyvien sanojen muodostamia ketjuja [HSO98, HH76]. Kohesiiviset suhteet on esitelty tutkielman luvussa 2.4, joka käsittelee tekstin koheesiota ja koherenssia. Koska ainoastaan leksikaalisen koheesion tunnistamiseen on olemassa yleisesti tunnettuja menetelmiä, perustuvat leksikaaliset ketjut käytännössä yleensä ainoastaan siihen. Leksikaalisia ketjuja on käytetty hyväksi tekstin katkelmoinnissa [MH91, KKM98], automaattisessa yhteenvetojen generoinnissa [SM00],

tiedonhaussa [MIO00] ja sanojen semanttisessa yksiselitteistämisessä [HSO98].

Leksikaalisten ketjujen muodostamiseksi tarvittava tieto sanojen välisistä semanttisista suhteista on useissa tapauksissa haettu WordNetistä. Tähän tarkoitukseen on varsinkin ennen WordNetin kehittämistä käytetty myös Rogets's Thesauruksen vuoden 1911 versiota, joka on ilmaiseksi saatavilla tutkimuskäyttöön sähköisessä muodossa [BDS93]. Semanttisten suhteiden lisäksi leksikaalisia ketjuja on muodostettu pelkän toiston perusteella ja kollokaatioiden perusteella.

Sekä leksikaalisten ketjujen muodostamiseksi että niiden vahvuuden laskemiseksi on esitetty useita toisistaan poikkeavia menetelmiä. Joissakin menetelmissä otetaan ketjuihin mukaan ainoastaan substantiivit [SM00, BE97], joissakin substantiivit, verbit ja adjektiivit [MIO00] ja joissakin substantiivit ja pronominit [KKM98]. Useissa tapauksissa substantiivit kattavat myös sellaiset substantiivilausekkeet, joissa määreenä on joko adjektiivi tai substantiivi, esim. *red wine* ja *atom bomb*. Molemmat esimerkit löytyvät WordNetin substantiivitietokannasta. Harvinaisempia tai johonkin tiettyyn aihealueeseen liittyviä substantiivilausekkeita on joissain menetelmissä tunnistettu pintalauseenjäsennyksen avulla [BE97]. Yleensä tiedonhaussa on substantiivilausekkeita pidetty sisältösanoina, eli sellaisina sanoina, jotka kaikkein parhaiten kuvaavat käsillä olevan tekstin merkitystä. Pronominien ottaminen mukaan leksikaalisiin ketjuihin on harvinainen ratkaisu. Jos olisi olemassa yleinen menetelmä, jolla pronominien viittaussuhteet saataisiin selville, voitaisiin pronominit leksikaalisissa ketjuissa korvata niillä substantiivilausekkeilla, joihin ne viittaavat. Joissakin leksikaalisten ketjujen muodostamismenetelmissä otetaan huomioon homonymia ja polysemia [MIO00, BE97], mutta joissakin ne jätetään huomioimatta [KKM98, MH91].

Seuraavaksi esitellään yksityiskohtaisemmin kaksi leksikaalisten ketjujen muodostamismenetelmää. Ensimmäisen menetelmän ovat kehittäneet Regina Barzilay ja Michel Elhadad ja siinä leksikaalisia ketjuja käytetään tiivistelmän gene-

roimiseen ja sen sivutuotteena syntyy sanojen semanttinen yksiselitteistäminen [BE97]. Toista menetelmää käytetään katkelmahaussa ja sen sivutuotteena syntyvät katkelmat, jotka sovellus palauttaa vastauksena esitettyyn kyselyyn. Menetelmän kehittäjät ovat Hajime Mochizukin, Makoto Iwayaman ja Manabu Okumura [MIO00].

4.2.1 Semanttinen yksiselitteistäminen ja tiivistelmän generointi

Regina Barzilayn ja Michael Elhadadin kehittämä leksikaalisten ketjujen muodostamismenetelmä käyttää hyväkseen WordNet-tietokantaa, sanaluokkien merkitsijää, substantiivilausekkeet tunnistavaa pintajäsentäjää sekä Marti A. Hearstin kehittämää TextTiling-katkelmointialgoritmia [Hea97]. Leksikaalisten ketjujen muodostamisen lisäksi he esittelevät menetelmän vahvojen ketjujen tunnistamiseksi. Heidän menetelmänsä perustuu pitkälti Graeme Hirstin ja David St-Ongen [HSO98] menetelmään. Hirst ja St-Onge käyttävät leksikaalisia ketjuja tekstin oikolukuun. Heidän sovelluksensa tavoitteena on tunnistaa ja korjata sellaiset sanat, jotka ovat sinänsä oikein kirjoitettuja, mutta jotka eivät sovi kontekstiinsa (engl. malapropism). Esimerkki: *ingenuous machine* korjattaisiin *ingenious machine*ksi.

Yleisesti ottaen leksikaalisten ketjujen modostaminen voidaan jakaa kolmeen vaiheeseen:

1. Valitaan mahdolliset ketjuihin kuuluvat sanat.
2. Käydään kaikki valitut sanat läpi ja katsotaan, kuuluvatko ne johonkin ketjuun.
3. Jos ketju löytyi, sana lisätään siihen. Jos ketjua ei löytynyt, luodaan uusi ketju, jonka ensimmäisenä sanana ko. sana on.

Yksityiskohtaisemmin algoritmi toimii seuraavasti:

1. Katkelmoidaan teksti käyttäen Hearstin TextTiling-algoritmia ja suoritetaan kohdat 2 - 4 erikseen jokaiselle katkelmalle.

2. Valitaan kaikki substantiivilausekkeet. Jos koko substantiivilauseke ei ole WordNetin substantiivitietokannassa, etsitään WordNetistä ainoastaan lausekkeen pääsanaa.

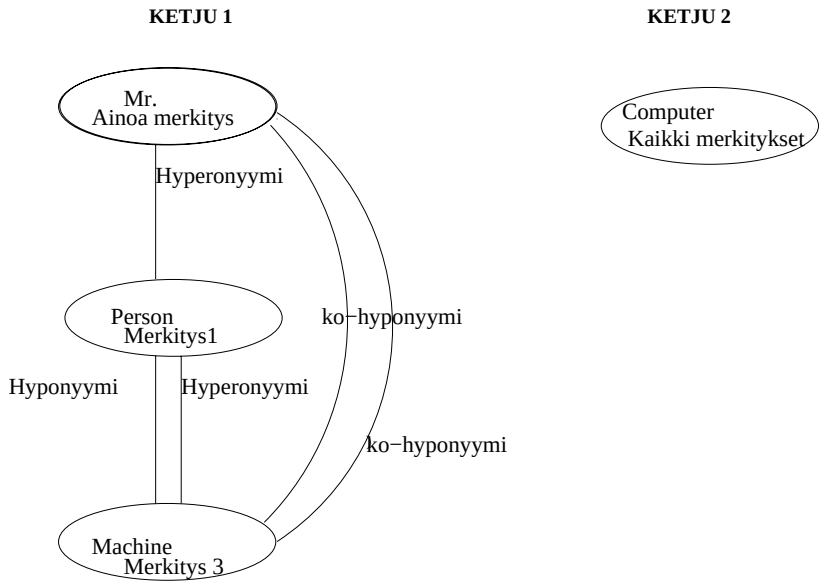
3. Sanan kuuluminen leksikaaliseen ketjuun riippuu sen etäisyydestä muista ketjun sanoista ja siitä, millainen semanttinen suhde sen ja muiden ketjun sanojen välillä vallitsee.

4. Jos sopiva ketju löytyi, sana lisätään siihen ja samalla merkitään, mikä sanan mahdollisesti monista eri merkityksistä on käytössä. Tarpeen vaatiessa myös muiden ketjussa esiintyvien moniselitteisten sanojen merkitys päivitetään. Jos sopivaa ketjua ei löytynyt ja luodaan uusi alkava ketju, on sanalla tässä ketjussa kaikki sen mahdolliset merkitykset.

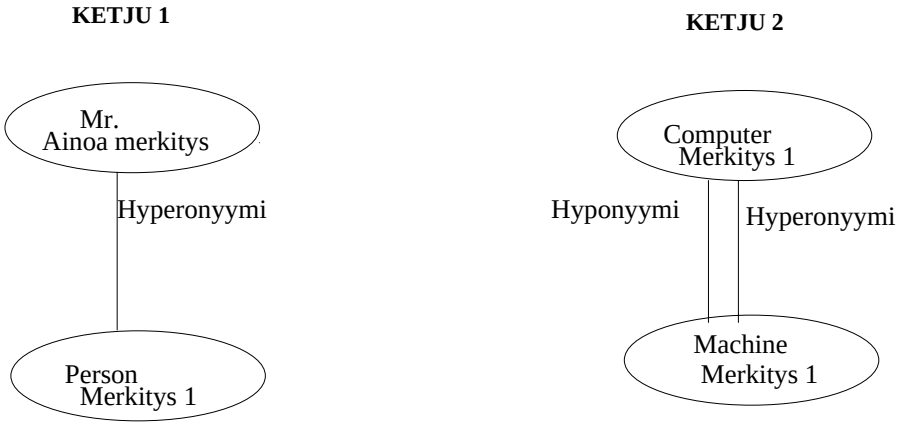
5. Katsotaan, jatkuuko jokin leksikaalinen ketju katkelmarajan yli. Tässä käytetään tiukempia kriteereitä kuin katkelman sisäisiä leksikaalisia ketjuja etsittäessä.

Sanojen yksiselitteistämismenetelmä ei ole ahne, eli sen jälkeen kun jonkin sanan merkitys on kiinnitetty, voidaan sitä muuttaa jos myöhemmin tulee runsaasti sellaisia sanoja, jotka liittyvät sanan johonkin toiseen merkitykseen. Käytännössä tämä on toteutettu siten, että leksikaalisia ketjuja muodostettaessa pidetään koko ajan yllä kaikkia mahdollisia ketjuja, joissa on kaikkien sanojen kaikki mahdolliset merkitykset. Laskennallisesti tehokkaammalla tavalla tämän osan algoritmista ovat toteuttaneet H. Gregory Silber ja Kathleen F. McCoy [SM00]. Kuvassa 4

esitetään yksi tapa muodostaa leksikaaliset ketjut sanoista *Mr., computer, machine* ja *person* ja kuvassa 5 toinen.



Kuva 4: *Sanoista Mr., computer, machine ja person muodostetut kaksi leksikaalista ketjua. Sanasta machine on kejussa 1 tehokasta ihmistä tarkoittava merkitys ja sanalla computer on sen molemmat merkitykset, sillä se on ketjun 2 ainoa sana.*



Kuva 5: *Sanoista Mr., computer, machine ja person muodostetut kaksi leksikaalista ketjua. Sanalla computer on ketjussa 2 sen tietokonetta tarkoittava merkitys ja sanalla machine sen konetta tarkoittava merkitys.*

Kuvat voi tulkita verkoiksi, joissa sanat ovat solmuja ja semanttisia suhteita kuvaavat viivat kaaria. Kun verkkojen lukumäärä kasvaa yli tietyn kynnysarvon, poistetaan ne samoja sanoja sisältävät verkot, joissa on vähiten kaaria ja joissa kaarien painoarvo on pienin. Kaarien painoarvot on määritelty seuraavasti: toisto ja synonyymi: 10, antonyymi: 7, hyperonyymi ja hyponyymi 4. Painoarvot perustuvat lingvistiseen intuitioon. Silber ja McCoy testasivat menetelmää eri painoarvoilla ja he päätyivät siihen, että ohjatut koneoppimismenetelmät voisivat olla hyvä tapa löytää sopiva painotus [SM00].

Leksikaalisten ketjujen vahvuuden arvioimiseksi Barzilay ja Elhadad analysoivat seuraavia ketjujen ominaisuuksia: ketjun pituus, ketjulla katetun tekstin pituus, ketjun jakautuminen tekstin sekaan, ketjun homogeenisuus eli samojen sanojen toistuminen, ketjun tiheys ja verkon halkaisija. Ketjun tiheys kuvaa sitä, kuinka paljon ketjuun kuulumattomia sanoja keskimäärin on ketjuun kuuluvien sanojen välissä. Barzilay ja Elhadad havaitsivat, että yhteenvedon generoinnin kannalta vahvoja olivat sellaiset ketjut, joiden pituus ja homogeenisuusarvot olivat suuria [BE97].

4.2.2 Katkelmahaku

Hajime Mochizuki, Makoto Iwayama ja Manabu Okumura ovat kehittäneet leksikaalisten ketjujen muodostamismenetelmän, joka perustuu paitsi saman sanan toistolle ja luvussa 2.5 esiteltyihin WordNetin tyyppisiin synonyymijoukkoihin, myös kollokaatioihin [MIO00]. Ketjujen luomisessa he käyttävät esikäsittelyvaiheessa morfologista analysoijaa ja ottavat ketjuihin mukaan vain substantiivit, verbit ja adjektiivit. Synonyymijoukkoihin perustuvat ketjut löydetään semanttisen yksiselitteistämisalgoritmin sekä kielitieteellisen merkityssanakirjan avulla. Kollokaatioista muodostuvat leksikaaliset ketjut löydetään aiemmin korpukses-

ta laskettujen sanojen samanlaisuusarvojen perusteella. Sanojen X ja Y samanlaisuusarvot on laskettu seuraavan kaavan avulla:

$$sim(X,Y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}, \tag{6}$$

missä x_i ja y_i ovat ovat sanojen X ja Y lukumäärät dokumentissa i ja n on korpuksessa olevien dokumenttien lukumäärä. Kyseessä on sama kosinimitta, joka esiteltiin luvussa 4.1.1, jossa sitä käytettiin dokumenttivektoreiden välisen samanlaisuuden laskemiseen. Tässä sitä käytetään sanavektoreiden välisen samanlaisuuden laskemiseen. Samanlaisuusarvon $sim(X,Y)$ avulla muodostetaan samanlaisten sanojen rypäitä eli leksikaalisia ketjuja. Kahden rypään C_i ja C_j välinen samanlaisuusarvo on kahden eri rypäisiin kuuluvan samanlaisimman sanan välinen samanlaisuusarvo eli:

$$sim(C_i, C_j) = \max_{X \in C_i, Y \in C_j} sim(X,Y) \tag{7}$$

Ketjujen muodostaminen kollokaatioiden perusteella on Mochizukin, Iwayaman ja Okumuran menetelmän kolmesta ketjuen muodostamistavasta monimutkaisin, joten vain se esitellään yksityiskohtaisemmin. Se tapahtuu seuraavalla tavalla:

1. Lasketaan sanojen väliset samanlaisuusarvot erikseen jokaisessa dokumentin virkkeessä.

2. Jokaisessa virkkeessä luodaan rypäät ja yritetään yhdistää niitä. Yhdistämistä jatketaan niin kauan kun rypäiden välinen samanlaisuusarvo ylittää tietyn kynnysarvon ja kun yhdistettävää löytyy. Näin syntyneet rypäät ovat virkkeen leksikaaliset ketjut.

3. Lopuksi yritetään muodostaa virkerajojen yli meneviä rypäitä. Jatkamisehto on sama kuin edellisessäkin kohdassa. Näin syntyneet rypäät ovat dokumentin leksikaaliset ketjut.

Koska yllä esitetyillä menetelmillä saattaa muodostua myös heikkoja leksikaalisia ketjuja, on seuraava vaihe niiden karsiminen. Heikkona ketjuna pidetään sellaisia ketjuja, jotka

1. ovat harvoja,
2. sisältävät pitkiä aukkoja tai
3. ovat liian lyhyitä.

Kun heikot ketjut on poistettu, lasketaan jäljelle jääneille ketjuille painoarvot. Koska Mochizuki, Iwayama ja Okumura ovat käyttäneet leksikaalisia ketjuja ainoastaan tiedonhaussa, on ketjujen painoarvomitta kehitetty palvelemaan erityisesti sitä eikä niinkään kuvaamaan jonkin tietyn leksikaalisen ketjun yleistä hyvyyttä. Mochizukin, Iwayaman ja Okumuran painotuskaavat perustuvat tutkielman luvussa 4.1.2 esiteltyyn $tf * idf$ -mittaan. Perusajatus kaavoissa on, että ketju on sitä tärkeämpi mitä enemmän siinä on termejä ja mitä harvemmin se esiintyy dokumenttikokoelman muissa dokumenteissa. Ketjulle C_d, joka perustuu saman sanan toistoon tai samaan synonyymijoukkoon kuulumiseen, painoarvo w lasketaan:

$$weight(C_d) = |C_d| * log(N/nC_d), \tag{8}$$

missä $|C_d|$ on ketjussa C_d olevien termien lukumäärä, N on dokumenttikokoelman dokumenttien lukumäärä, nC_d on niiden dokumenttien lukumäärä, joissa

leksikaalinen ketju C_d esiintyy. Kollokaatioihin perustuvan ketjun C_d painoarvo w lasketaan seuraavasti:

$$weight(C_d) = |C_d| * log(N / \max_{c \in C_d} n_c), \tag{9}$$

missä $\max_{c \in C_d} n_d$ on maksimi niiden dokumenttien lukumäärästä, jossa jokin ketjun C_d termi esiintyy.

Kun leksikaaliset ketjut on muodostettu, käytetään niitä sekä tietyn kyselyn suhteen oleellisten dokumenttien löytämiseen että dokumenttien katkelmoimiseen siten, että käyttäjälle palautetaan vastauksena kyselyyn vain se osa dokumentista, joka parhaiten sopii vastaukseksi.

5 Evaluointimenetelmiä

Semanttista samanlaisuutta tunnistavaa menetelmää voidaan arvioida useilla eri tavoilla. Sen tuloksien oikeellisuuden analysoimisen lisäksi voidaan myös tutkia itse menetelmää. Mitattavia menetelmän ominaisuuksia ovat esim. sen nopeus, skaalautuvuus eri kokoisille aineistoille ja tulkittavuus. Tulkittavuus ilmaisee, kuinka hyvin menetelmän käyttäytyminen on ihmisen ymmärrettävissä intuitiivisesti [HK01]. Menetelmän tuloksien oikeellisuutta voidaan tutkia myös analysoimalla ja todistamalla, että se on oikea ja täydellinen. Tämä kuitenkin edellyttää sitä, että ongelma, jonka menetelmä ratkaisee, on muotoiltavissa formaalisti, mikä ei päde tämän tutkielman aiheena olevaan semanttisen samankaltaisuuden tunnistamisongelmaan. Ainoa tapa arvioida semanttista samankaltaisuutta havaitsevan menetelmän hyvyyttä on siis kokeellinen evaluointi, jossa analysoidaan menetelmän tuloksia [Seb02]. Tässä luvussa kuvataan juuri tällaisia evaluointitapoja.

Tutkielman aiheena oleva samanlaisuuden havaitseminen voidaan mieltää binäärisenä luokitteluongelmana, jossa tekstifragmentti pyritään luokittelemaan samanlaiseksi tai ei-samanlaiseksi jonkin toisen tekstifragmentin kanssa. Näinollen menetelmän arviointiin voidaan soveltaa luokittelumenetelmien evaluointiin käytettyjä mittoja. Tämän lisäksi tekstifragmenttien välistä samanlaisuutta tunnistavan menetelmän evaluointiin soveltuvat tiedonhakujärjestelmien evaluoinnin mitat. Seuraavaksi luvussa 5.1 esitellään tiedonhakujärjestelmien evaluoinnissa käytettyjä perusmittoja ja luvussa 5.2 esitellään muita käyttökelpoisia mittoja. Luvussa 5.3 esitellään kaksi tapaa laskea menetelmän keskimääräistä suorituskykyä kuvaavat arvot ja luvussa 5.4 käsitellään erilaisten testikokoelmien merkitystä testitulosten vertailukelpoisuuden kannalta.

5.1 Saanti ja tarkkuus

Tiedonhaun tehokkuuden evaluoinnissa asemansa vakiinnuttaneet mitat ovat tarkkuus (engl. precision) ja saanti (engl. recall) [vR80]. Mittojen laskeminen perustuu siihen, että luokittelun tulos jaetaan taulukossa 12 esitettyihin luokkiin, jolloin tarkkuus voidaan määritellä seuraavasti:

$$tarkkuus = \frac{TP}{TP + FP} \tag{10}$$

ja saanti puolestaan seuraavasti:

$$saanti = \frac{TP}{TP + FN}. \tag{11}$$

Täydellisessä järjestelmässä sekä tarkkuuden että saannin arvo on 1. Huomattavaa on, että tarkkuuden ja saannin arvot ovat sidoksissa toisiinsa. Usein jos menetelmän tarkkuus paranee, niin sen saanti huononee ja jos sen saanti paranee,

		Asiantuntijan arvio	
		Kyllä	Ei
Luokittelijan arvio	Kyllä	TP	FP
	Ei	FN	TN

Taulukko 12: *Mahdollisuustaulukko (engl. contingency table) esittää asiantuntujan ja luokittelijan arviot siitä, kuuluuko dokumentti d_j luokkaan c_i. TP = true positives, FP = false positives, FN = false negatives ja TN = true negatives.*

niin tarkkuus huononee. Tämän takia käytetään usein saannin ja tarkkuuden yhdistävää F-mittaa (engl. F-measure), jossa saantia ja tarkkuutta painotetaan parametrin β avulla. F-mitta määritellään seuraavalla tavalla:

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}, \tag{12}$$

missä P tarkoittaa tarkkuutta ja R saantia. Kun β on yksi, tarkkuuden ja saannin painoarvot ovat yhtä suuret, jolloin kaava yksinkertaistuu alla esitettyyn muotoon:

$$F_1 = \frac{2PR}{P + R}$$

Kun β on suurempi kuin yksi, saannin painoarvo on suurempi ja kun se on pienempi kuin yksi, tarkkuudella on suurempi painoarvo. Painotuksen käyttö evaluoinnissa on hyödyllistä silloin, kun syystä tai toisesta joko tarkkuutta tai saantia pidetään tärkeämpänä. Sillon, kun luokittelun tulokset ohjataan ihmisasiantuntijalle jatkokäsittelyä varten, saattaa hyvä saanti olla tärkeämpi kuin tarkkuus. Jos luokittelijan pitää toimia ilman ihmisen apua, on tarkkuus usein vähintään yhtä tärkeä ominaisuus kuin saanti.

Joissakin tiedonhaku- ja luokittelujärjestelmissä tehokkuus ilmoitetaan saannin ja tarkkuuden yhtäsuuruuspisteen arvona (engl. break-even point) [Seb02]. Se on

piste, jossa saanti- ja tarkkuuskäyrät leikkaavat toisensa ja jossa saannin, tarkkuuden ja F_1:sen arvot ovat samat. Kaikissa järjestelmissä saanti- ja tarkkuuskäyrät eivät leikkaa toisiaan ollenkaan. Silloin yhtäsuuruuspisteen laskemiseen pitää käyttää erillistä kaavaa [Seb02].

5.2 Muut mitat

Jos luokat ovat merkityksensä mukaan jakautuneet hyvin epätasaisesti semanttiseen avaruuteen, pelkän saannin ja tarkkuuden mittaaminen ei anna kattavaa kuvaa järjestelmän tehokkuudesta. Tällaisessa tilanteessa jotkin väärät luokitukset ovat vähemmän väärin kuin toiset. Tällöin voidaan muiden mittojen lisäksi ottaa käyttöön osittainen arvotus (engl. partial credit) [JM00]. Silloin esim. vain vähän väärin menneestä luokituksesta voi saada puoli pistettä, täysin väärästä nolla ja täysin oikeasta yhden pisteen. Osittainen arvotus on käyttökelpoinen menetelmä silloin, kun asiantuntijoiden suorittamassa luokituksessa on suuria eroja eri asiantuntijoiden välillä.

On olemassa myös muita tilanteita, joissa on mielekästä antaa järjestelmän vastaukselle jokin muu arvo kuin vain joko oikein tai väärin. Tällainen tilanne syntyy mm. silloin, kun järjestelmä palauttaa järjestetyn joukon vastauksia. Jos ensimmäisenä on oikea vastaus, annetaan täydet pisteet eli 1 ja jos oikea vastaus löytyy myöhemmin listalta, annetaan vastauksesta vähemmän pisteitä. Nolla pistettä annetaan vain silloin, kun oikea vastaus ei ole ollenkaan vastauslistassa. Tällainen pistetytysjärjestelmä on käytössä mm. TREC:in kysymysvastauskilpailusarjassa [Voo99]. Siinä pisteet lasketaan siten, että kukin kysymys saa pisteet sen mukaan, monennellako sijalla oikea vastaus on. Pistemäärä on sijan käänteisluku. Esimerkiksi jos oikea vastaus on ensimmäisenä pisteet ovat 1 / 1, toisella sijalla 1 / 2, kolmannella sijalla 1 / 3 jne.

Dokumenttien luokittelumenetelmien evaluoinnissa on saannin ja tarkkuuden sijaan usein laskettu oikein ja väärin luokiteltujen määriä suhteessa koko aineiston kokoon [Seb02], jolloin täydellisessä järjestelmässä oikeat (engl. accuracy) saa arvon 1 ja virheelliset (engl. error) 0. Oikeat lasketaan seuraavan kaavan mukaan

$$oikeat = \frac{TP + TN}{TP + FP + FN + TN}, \tag{13}$$

missä muuttujat TP, TN, FP ja FN ovat samat kuin taulukossa 12. Vastaavasti virheelliset lasketaan seuraavalla tavalla:

$$virheelliset = \frac{FP + FN}{TP + FP + FN + TN}. \tag{14}$$

5.3 Keskiarvot

Yksittäisten tapausten tehokkuusmitat eivät riitä kuvaamaan koko menetelmän tehokkuutta, vaan tarvitaan menetelmiä järjestelmän keskimääräisen tehokkuuden laskemiseen. Yksittäisiä tapauksia ovat esim. tiedonhaussa tietyn kyselyn suorittamisen seurauksena löydetyt dokumentit ja luokittelussa tiettyyn kategoriaan luokitellut tekstifragmentit.

Keskimääräisten arvojen laskemiseksi on käytössä kaksi eri menetelmää: makroevaluointi (engl. macro-evaluation, macroaveraging) ja mikroevaluointi (engl. micro-evaluation, microaveraging) [vR80]. *Makroevaluoinnissa* kullekin yksittäiselle tapaukselle lasketaan tehokkuusmitta, esim. F_1. Koko järjestelmän tehokkuus on yksittäisten mittojen keskiarvo [Yan97]. Makroevaluoinnissa kukin kategoria tai kukin kysely saa saman painoarvon riippumatta siitä, kuinka yleinen ko. kategoria on koko kokoelmassa tai siitä, kuinka monta relevanttia dokumenttia kysely kattaa. Liitteessä 5 on laskettu tutkielman käytännöllisessä osassa to-

teutetun kysymysvastausjärjestelmän suorituskyky eri vektorityypeillä käyttäen makroevaluointia.

Mikroevaluoinnissa kaikkia järjestelmän tekemiä päätöksiä käsitellään yhtenä ryhmänä. Tämä tarkoittaa sitä, että kaikki yksittäisiä tapauksia kuvaavat taulukossa 12 esitetyt arvot yhdistetään samaan tauluun, josta sitten lasketaan yhteinen tehokkuusmitta. Tällainen laskutapa painottaa yleisten kategorioiden tai suuria määriä dokumentteja hakevien kyselyjen arvoja. Usein tiedonhakujärjestelmiä evaluoidessa käytetään makroevaluointia ja kategorisointia evaluoidessa mikroevaluointia [Lew91].

5.4 Testiaineistot

Luokittelijan tulosten mittaamista saattaa vaikeuttaa huomattavasti se, että ihmisasiantuntijoilla on eriäviä mielipiteitä siitä, mikä luokitus on oikein ja mikä väärin. Yksi tapa ratkaista tämä ongelma on se, että ennen kuin aloitetaan eri luokittimien tulosten arviointi ja vertailu keskenään, sovitaan mittaustehtävässä käytettävästä luokituksesta. Helpoiten se käy siten, että kaikki evaluoijat käyttävät samaa testimateriaalia, johon on valmiiksi merkattu oikea luokitus. Näin tehdään sekä TREC- [BYRN99] että MUC-konferensseissa [JM00] kun mitataan eri järjestelmien toimivuutta. Myös SENSEVAL-1-projektissa, jossa rakennetttiin evaluointiympäristö sanojen semanttisia luokittelijoita varten luotiin valmiiksi luokiteltu opetus- ja testiaineisto [KR00].

Jos luokitustehtävä on sen tyyppinen, että aineistoon ei ole luontevaa asettaa minkään yksittäisen luokitustavan mukaista luokitusta, voidaan useita ihmisiä pyytää luokittelemaan sama aineisto ja sitten ottaa evaluoinnissa huomioon ihmisluokittelijoiden väliset erimielisyydet. Tämän tyyppinen luokitustehtävä on esim. tekstin semanttinen katkelmointi, jossa teksti pyritään jakamaan toisiinsa

aihealueen puolesta liittyviin kokonaisuuksiin [Hea97]. Koska tällaisessa luokittelussa ei ole olemassa yhtä ainoaa oikeaa ratkaisua, on tämä syytä ottaa huomioon myös automaattisen katkelmoinnin tuloksia evaluoitaessa.

Toinen tapa välttää aineiston luokittelu evaluointia varten on luokittelun mittaaminen epäsuorasti sitä käyttävän järjestelmän suorituskyvyn evaluoinnin kautta [Hea97]. Tällainen mittaustapa edesauttaa tietyn sovelluksen tarpeisiin sopivan luokittelutavan kehittämistä, ja se mahdollistaa uudentyyppisten luokitusten kokeilua, koska siinä ei vaadita, että syntyvän luokittelun pitäisi vastata ihmisen tekemää luokitusta. Huono puoli luokittelun epäsuorassa evaluoinnissa on se, että itse luokittelua käyttävän järjestelmän evaluointi saattaa olla vaikeaa. Tämän lisäksi se, että luokittelija toimii hyvin tietynlaisissa sovelluksissa ei vielä kerro mitään sen toimivuudesta muissa sovelluksissa [Lew91].

Jos kategorisoinnin suorittava luokittelija on modostettu käyttäen jotakin ohjattua koneoppimismenetelmää, ei luokittelijan evaluointia voida luotettavasti suorittaa käyttämällä testiaineistona samaa aineistoa, jolla opettaminen on tapahtunut. Jos opetusvaiheessa on tapahtunut ylioppimista (engl. overfitting), antaa opetusaineistolla testaaminen aivan liian optimistisen käsityksen luokittelijan luokittelukyvystä [HK01]. Tämän ongelman välttämiseksi esitellään tässä kaksi tekniikkaa.

Ensimmäisessä menetelmässä saatavilla oleva aineisto jaetaan sattumanvaraisesti kahteen osaan, opetusaineistoon ja testiaineistoon. Yleensä opetusaineisto käsittää kaksi kolmasosaa koko aineistosta ja testiaineisto yhden kolmasosan. Opetusaineistoa käytetään luokittelijan opettamiseen ja testiaineistoa luokittelijan luokituskyvyn evaluoimiseen. Näin tehtiin luvussa 3.3 esitellyssä menetelmässä, joka oppi korpuksesta indeksointitermien todennäköisyyksiä.

Toisessa menetelmässä, jota kutsutaan *k-kertaiseksi ristiinvalidoinniksi* (engl. k-fold

cross-validation), on perusajatus sama kuin ensimmäisessäkin menetelmässä. Erona on se, että aineisto jaetaan *k* osaan ja opetus ja testaus suoritetaan *k* kertaa. Joka iteraatiokerralla yksi osa jätetään testiaineistoksi ja jäljelle jääneitä osia käytetään opetukseen. Luokittelijan oikeellisuus (ks. kaava 13) on kaikkien iteraatiokertojen oikeiden luokituksien summa jaettuna alkuperäisessä aineistossa olevien luokiteltavien olioiden lukumäärällä [HK01].

6 Kysymysten välisen samanlaisuuden tunnistaminen Tipu-kysymysvastausjärjestelmässä

Tekstifragmenttien välistä samanlaisuutta tunnistavia menetelmiä on testattu käytännössä osana APPA-tutkimusprojektia kehitettyä *Tipu*-nimistä kysymysvastausjärjestelmän prototyyppiä. Seuraavaksi luvussa 6.1 kerrotaan lyhyesti kysymysvastausjärjestelmistä yleensä ja sen jälkeen luvussa 6.2 esitellään Tipun arkkitehtuuri. Luvussa 6.3 kerrotaan, millaisia erilaisia vektorityyppejä kysymysten välistä samanlaisuutta tunnistavassa komponentissa on testattu. Luvussa 6.4 kerrotaan yleisellä tasolla minkä tyyppinen kysymysvastausaineisto Tipun käytettävissä on. Luvussa 6.5 esitellään Tipun evaluoinnissa käytetyt menetelmät sekä evaluoinnin tulokset.

6.1 Kysymysvastausjärjestelmät

Kysymysvastausjärjestelmä on sovellus, jonka syötteenä on käyttäjän luonnollisella kielellä esittämä kysymys ja tulosteena luonnollisella kielellä esitetty vastaus tai vastausjoukko. Jos tulosteena on vastausjoukko, on se järjestetty siten, että ensimmäisenä on järjestelmän mukaan paras vastaus ja viimeisenä huonoin. Kysymysvastausjärjestelmiä voidaan toteuttaa monella eri tavalla. Yksi tapa on

se, että järjestelmän käyttämä tieto on jossakin tietokannassa kuten relaatiotietokannassa, jolloin tutkimusongelmaksi muodostuu, kuinka luonnollista kieltä voidaan muuntaa jollekin kyselykielelle kuten SQL-kielelle. Tämä lähestymistapa on ollut tunnettu jo 1960-luvun lopulta lähtien, jolloin ensimmäiset kysymysvastausjärjestelmät toteutettiin [Woo73, GM89]. Tällaisessa järjestelmässä vastaus kysymykseen voidaan joko tulostaa käyttäjälle suoraan tietokannasta tai se voidaan ensin muuntaa luonnollisen kielen lauseiksi. Tässä tutkielmassa kysymysvastausjärjestelmiä lähestytään tiedonhaun näkökulmasta, sillä Tipu-prototyypillä ei ole käytössään mitään muuta tietokantaa kuin vapaana tekstinä ilmaistuja kysymysvastauspareja, joiden määrä lisääntyy jatkuvasti ja joiden aihepiirit muuttuvat. Yleiskatsaus kysymysvastausjärjestelmiin tiedonhaun näkökulmasta löytyy Reeta Kuuskosken laudaturtutkielmasta [Kuu01].

Tekstitietokantaan perustuvan kysymysvastausjärjestelmän toteutusta voisi lähestyä siten, että pyrittäisiin poimimaan vastauksista niiden sisältämä tieto, jolloin olisi mahdollista vastata esim. sellaisiin kysymyksiin, jotka edellyttävät useassa eri vastauksessa olevan tiedon yhdistämistä. Esimerkki: Kysymys: *Mitkä ovat Suomen kaksi suurinta kaupunkia?*, vastaus 1: *Suomen pääkaupunki ja samalla suurin kaupunki on Helsinki.*, vastaus 2: *Suomen toiseksi suurin kaupunki on Espoo.* Tässä tutkielmassa on kuitenkin lähdetty siitä, että vastauksiksi annetaan ainoastaan jo olemassaolevia vastauksia eikä luoda uusia vastauksia yhdistelemällä eri vastauksista poimittua tietoa tai poimimalla vastauksesta vain osa sen sisältämästä tiedosta. Tärkeä haaste Tipu-prototyypissä on myös se, kuinka tunnistetaan ne kysymykset, joihin ei käytettävissä olevassa tietokannassa ole vastausta.

Tiedonhaun tutkimuksen piirissä on kysymysvastausjärjestelmiä toteutettu mm. siten, että ensin etsitään ne kohdat dokumenttikokoelman dokumenteista, jotka todennäköisesti sisältävät vastauksen kysymykseen. Sen jälkeen dokumentit katkelmoidaan ja käyttäjälle annetaan vastauksina hänen esittämäänsä kysymyk-

seen paremmuusjärjestykseen järjestetty lista katkelmia. Tämä on perusajatuksena myös TREC:in kysymysvastausjärjestelmäsarjassa [VH00].

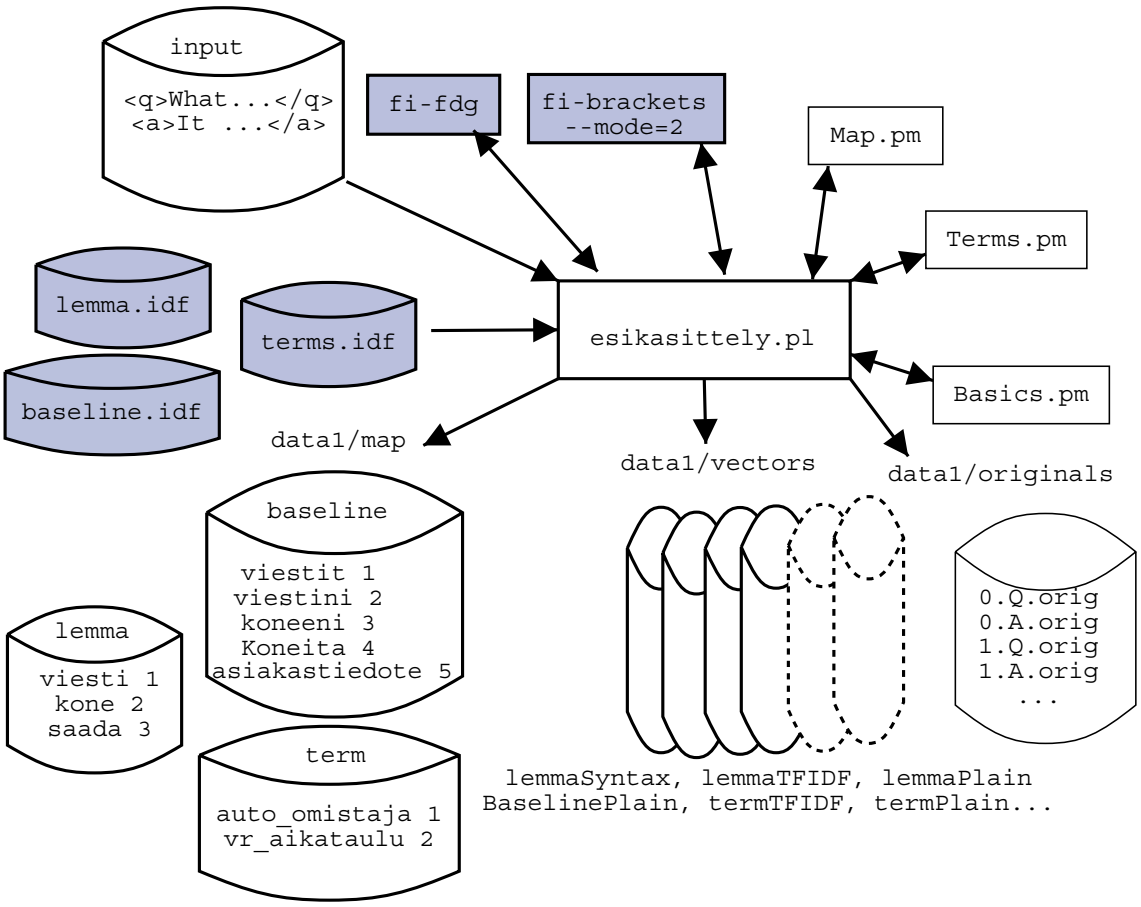
6.2 Järjestelmän arkkitehtuuri

Toteutettu kysymysvastausjärjestelmä koostuu kahdesta erillisestä ohjelmasta, jotka käyttävät hyväkseen samoja moduuleita ja tietokantoja [3]. Nämä erilliset ohjelmat ovat esikäsittelijä ja vuorovaikutteiseen käyttöön tarkoitettu kysymysvastausjärjestelmä nimeltä Tipu. Esikäsittelijää käytetään ainoastaan silloin, kun järjestelmä otetaan käyttöön. Seuraavaksi esitellään kummankin ohjelman arkkitehtuuri pääpiirteissään.

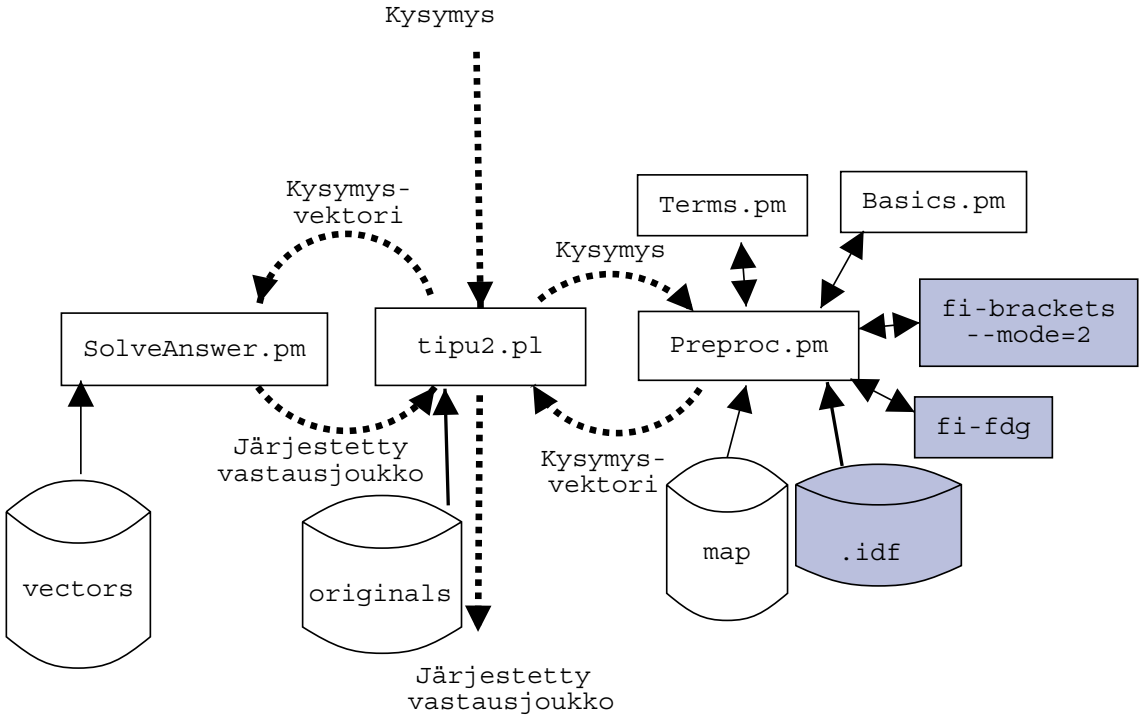
Esikäsittelijä saa syötteeksi kysymysvastauspareista koostuvan tiedoston. Ensin se numeroi kunkin parin ja tallentaa parit numeroidensa mukaisiin tiedostoihin. Sen jälkeen esikäsittelijä ottaa pareista mukaan jatkokäsittelyyn ainoastaan kysymykset, poistaa niistä välimerkit ja suorittaa alkioanalyysin. Alkiot eli termit ovat välilyönnein eroteltuja merkkijonoja (*baseline*), lemmatisoituja sanoja (*lemma*) ja termitunnistimella tunnistettuja termejä (*term*). Kullekin termityypille luodaan oma avaruutensa, jossa kukin termi saa kokonaisluvusta muodostuvan tunnuksen. Alkioanalyysin jälkeen kunkin kysymyksen termit painotetaan. Painotuksessa on kolme erilaista vaihtoehtoa: ei painotusta (*plain*), painotus termien syntaktisiin funktioihin perustuen (*syntax*) kuten luvussa 3.3 on esitetty sekä luvussa 4.1.2 esitetty $tf * idf$ -painotus (*tfidf*). Kaikenkaikkiaan muodostetaan yhteensä 7 erilaista vektorityyppiä. Plain- ja tfidf-painotusta käytetään kaikkien termityyppien kanssa ja syntax-painotusta ainoastaan lemma-termien kanssa.

[3] Tietokannalla tarkoitetaan tässä kokoelmaa toisiinsa liittyviä ja jotakin käyttötarkoitusta varten kerättyjä säilytettäviä tietoja. Tietokannan tekniseen toteutukseen ei tässä oteta kantaa. Se voi olla toteutettu esim. relaatiotietokantana, käyttöjärjestelmän hakemisto- ja tiedostojärjestelmää hyväksi käyttäen tai yksinkertaisena tekstitiedostona.

Esikäsittelyn tuloksena syntyy kolme tietokantaa: alkuperäiset kysymykset ja vastaukset pilkottuina omiin tiedostoihinsa, kolme erilaista termiavaruutta sisältävät muunnostaulukot termeistä kokonaisluvuiksi ja kysymykset esitettyinä vektoreina seitsemällä eri tavalla. Esikäsittelijä käyttää ulkopuolisen ohjelman generoimia tietokantoja *baseline.idf, lemma.idf* ja *terms.idf*, jotka sisältävät kunkin termityypin kokoelmakohtaiset idf-arvot. Kuvassa 6 on esitetty esikäsittelijän arkkitehtuuri.



Kuva 6: *Esikäsittelijän arkkitehtuuri. Sylinterit kuvaavat tietokantoja ja nelikulmiot ohjelmamoduuleja. Valkoiset kuviot ovat itse kirjoitettuja ohjelmia tai niiden generoimia tiedostoja ja värilliset ovat valmiita ohjelmia, joita vain kutsutaan tai jonkin muun ohjelman generoimia tiedostoja. Nuolten suunta osoittaa, mihin suuntaan tietoa moduulien ja tietokantojen välillä liikkuu. Pääohjelma sijaitsee keskellä ja sen nimi on esikasittely.pl.*



Kuva 7: *Tipu-kysymysvastausjärjestelmän arkkitehtuuri. Tietokannat, ohjelmamoduulit ja tiedon kulku moduulien ja varastojen välillä on kuvattu käyttäen samaa merkintätapaa kuin kuvassa 6. Tässä kuvassa tietokannat on kuitenkin kuvattu yleisemmällä tasolla kuin kuvassa 6. Pisteviivalla on esitetty, mitä muunnoksia käyttäjän syötteenä antamalle kysymykselle tehdään ja missä vaiheessa järjestelmän tuloste eli vastaus kysymykseen luodaan.*

Järjestelmän toinen osa on Tipu eli vuorovaikutteiseen käyttöön tarkoitettu varsinainen kysymysvastausjärjestelmä. Tipun arkkitehtuuri on esitetty kuvassa 7. Tipu käynnistetään siten, että ohjelmalle annetaan kysymys, johon halutaan saada vastaus. Tämän lisäksi käynnistysvaiheessa voi asettaa erilaisia parametreja. Parametreilla voi vaikuttaa seuraaviin asioihin: mitä vektorityyppiä laskennassa käytetään, millä algoritmilla laskenta suoritetaan, mikä on kynnysarvo tekstifragmenttien väliselle samanlaisuudelle, kuinka monta vastausta enintään tulostetaan ja missä muodossa vastaus halutaan. Oletusarvoisesti järjestelmä käyttää vektoreina *lemmaSyntax*ia eli lemmatisoituja syntaktisten funktioiden mukaan

painotettuja vektoreita, laskennassa oletusarvo on kosinimitta, oletusraja-arvo tekstifragmenttien samanlaisuudelle on 0.66, oletusmaksimimäärä vastauksille on 10 ja oletusarvoinen tulostusmuoto on XML eli eXtensible Markup Language. Lisäksi järjestelmälle voidaan syöteparametreilla kertoa, tulostaako se muutakin kuin vain järjestetyn vastausjoukon. Muuta tulostettavaa tietoa voi olla esimerkiksi vastauksiin tietokannassa liittyvät kysymykset ja kaikkien tulostuksessa esiintyvien kysymysten esitysmuodot sellaisina vektoreina joiden termit on esitetty kokonaislukujen sijaan merkkijonoina eli baseline-, lemma- tai term-muodossa.

Tipu toimii siten, että pääohjelma eli tipu2.pl kutsuu esikäsittelymoduulia *Preproc*, joka muuntaa kysymyksen halutunlaiseksi vektoriksi. Muunnoksessa Preproc.pm käyttää sekä omia alimoduulejaan että *map*-tietokantaa, joka sisältää eri alkiotyyppien muunnostaulut kokonaisluvuiksi. Lisäksi se käyttää tarvittaessa ulkoista termikohtaiset idf-arvot sisältävää tietokantaa ja ulkoisia ohjelmia *brackets-fi*, joka eristää termit kysymyksestä ja *fi-fdg*, joka sekä lemmatisoi kysymyksen että suorittaa sille syntaktisen analyysin funktionaalisen dependenssikieliopin mukaan. Esimerkit ohjelmien tulostuksesta löytyvät liitteistä 2 ja 4. Kun esikäsittelymoduuli on muuntanut kysymyksen vektoriksi, välitetään se edelleen *SolveAnswer* -moduuliin, joka laskee, mitä tietokannassa olevaa kysymysvektoria lähimpänä uusi kysymys on ja palauttaa tuloksena järjestetyn listan samanlaisten kysymysten vastauksista. Tämä lista palautetaan alkuperäisen kysymyksen esittäneelle käyttäjälle. Kysymysten vertailussa SolveAnswer-moduuli käyttää sopivia *vectors*-tietokannan vektoreita ja vastauksia palauttaessaan se käyttää alkuperäisiä kysymyksiä, jotka on talletettu *originals*-tietokantaan.

6.3 Testattavat menetelmät ja vektorityypit

Kysymysten välisen samanlaisuuden tunnistamisessa on järjestelmässä käytetty luvussa esitettyä 4.1 vektorimallia ja kosinimittaa.

Kysymysvektorien termit on muodostettu kolmella eri tavalla. Ne ovat:

baseline Termejä ovat kaikki tekstissä esiintyvät toisistaan välilyönneillä erotetut merkkijonot. Välimerkit eivät kuitenkaan sisälly termeihin. Esimerkkejä baseline-termeistä: *aikataulua, aikatauluja, AIKATAULUN, aikataulut, aikataulu* ja *AIKATAULU*.

lemma Termejä ovat sananmuotojen lemmat. Kaikki edellisessä kohdassa esimerkkinä esitetyt baseline-termit ovat lemmatisoituina *aika#taulu*. #-merkki tarkoittaa yhdyssanan osien välistä rajaa.

term Termejä ovat tekstifragmenteista termitunnistimella löytyneet termit. Termitunnistimena on käytetty fi-brackets -nimistä ohjelmaa. Esimerkkejä termitunnistimen löytämistä termeistä ovat *aikataulu* ja *chat_nimimerkki*. Alaviiva sanojen välissä tarkoittaa, että termitunnistin on yhdistänyt kaksi erillistä sanaa yhdeksi termiksi. Kaikki termit koostuvat sanojen lemma-muodoista.

Termivektoreiden painottamiseen on käytetty kolmea eri tapaa. Ne ovat:

plain Pelkkä *tf* -painotus. Koska korpuksen fragmentit ovat lyhyitä, ovat ykköstä suuremmat painot harvinaisia.

tfidf Luvussa 4.1.2 esitetty $tf * idf$ -painotus.

syntax Lingvistiseen intuitioon perustuva painotus, jossa painotetaan niitä termejä, joiden sanaluokka on substantiivi ja jotka ovat sellaisten substantiivilausekkeiden pääsanoja, jotka ovat joko lauseen subjekteja, objekteja tai

komplementteja. Esimerkejä: *Voiko* YliopistoEnergia *siirtää sähkölaskuni suoraveloitukseen?* (subjekti), *Miten etsin* tiedostoja*?* (objekti), *Mikä on* Gnome*?* (komplementti). Syntax-tyyppistä painotusta sovelletaan ainoastaan lemmatisoiduille termeille.

6.4 Kysymysvastausaineisto

Kuten tutkielman luvussa 1 lyhyesti mainittiin, vaikuttaa käytetty tekstityyppi siihen, millaisia menetelmiä tekstifragmenttien välisen samanlaisuuden tunnistamiseen kannattaa käyttää. Tämä tarkoittaa sitä, että tässä tutkimuksessa saadut kokeelliset tulokset eivät välttämättä aivan sellaisenaan päde muihin tekstityyppeihin.

Järjestelmän käytettävissä oleva kysymysvastausaineisto koostuu liitteessä 1 esitetyn kaltaisista tekstiviesteinä ilmaistuista kysymysvastauspareista. Jokaisen yksittäisen käyttäjän esittämät kysymykset on koottu omaksi ryhmäkseen. Kokonaisuudessaan korpus käsittää noin 24 000 kysymysvastausparia, jotka on esittänyt noin 13 000 eri käyttäjää. Yhden kysymyksen maksimipituus on 160 merkkiä. Vastaukset ovat vain harvoin yli kahden tekstiviestin eli yli 320 merkin pituisia.

Tutkimuksen kysymyksille on tyypillistä, että ne sisältävät keskimäärin yhden virkkeen, että niissä on käytetty paljon puhekielen ilmaisuja ja että ne sisältävät runsaasti yleiskieleen kuulumattomia kaupallisiin tuotteisiin viittaavia termejä. Kysymyksissä on myös runsaasti kirjoitusvirheitä. Vastaukset eroavat kysymyksistä siten, että ne ovat keskimäärin hieman pidempiä kuin kysymykset ja että niiden kieli on pääosin virheetöntä yleiskieltä, lukuunottamatta kaupallisiin tuotteisiin liittyvää terminologiaa.

Erilaisia baseline-tyyppisiä termejä aineistosta löytyi 39 957 kpl, lemmoja 17 321 kpl ja termitunnistimen termejä 15 060 kpl. Aineistossa olevista erilaisista lem-

moista yhteensä 5 069 kpl eli noin 29% oli sellaisia, joita ei löytynyt fi-fdg -jäsentimen leksikosta. Ne olivat pääosin vieraskielisiä, murteellisia tai väärin kirjoitettuja sanoja sekä tuotteiden nimiä tai erisnimiä.

6.5 Evaluointimenetelmät ja tulokset

Tutkimuksessa on verrattu toisiinsa seitsemää eri tapaa muodostaa termivektorit kysymyksistä. Tavoitteena on ollut selvittää, mikä vektorityyppi parhaiten soveltuu lyhyistä kysymyksistä koostuvien tekstifragmenttien välisen samanlaisuuden tunnistamiseen.

Eri vektoreiden soveltuvuutta on arvioitu sen perusteella, miten hyvin kysymys-vastausjärjestelmä suoriutuu tehtävästään. Jos kysymykseen tulee oikea vastaus, osoittaa se, että vertailussa käytetty vektorityyppi on tarkoituksenmukainen. Arvioinnissa on käytetty myös *negatiivisia kysymyksiä* eli kysymyksiä, joihin järjestelmällä ei ole edellytyksiä vastata. Tällöin järjestelmän odotetaan tunnistavan tilanne ja olevan palauttamatta ainuttakaan vastausta.

Evaluoinnissa on käytetty kahdenlaista mittaa. Toinen mitta ilmoittaa, kuinka monta Tipun palauttamista vastauksista on oikein ja toinen kertoo, miten hyvin Tipu löytää oikean vastauksen ja mille sijalle se on tuloksessa asetettu. Ensin mainittu mitta on luvussa 5.1 määritelty tarkkuus. Toinen mitta on TREC-kilpailujen kysymysvastaussarjassa käytetty pisteytys ja se on esitelty tutkielman luvussa 5.2. Saantia ei voida laskea, sillä se edellyttäisi että tiedettäisiin kysymysten kaikki oikeat vastaukset. Tämä edellyttäisi, että koko kysymysvastauskorpus olisi luokiteltu siten, että kukin luokka sisältäisi tiettyyn kysymykseen tai kysymysjoukkoon sopivat vastaukset. Keskimääräiset arvot on laskettu käyttäen luvussa 5.3 esiteltyä makroevaluointia.

Negatiivisten kysymysten evaluointiin tarkkuutta ja TREC-tyyppistä pisteytystä

on sovellettu siten, että oikeaksi vastaukseksi on katsottu vastauksen puuttuminen. Jos Tipu ei palauta yhtään vastausta, annetaan sekä tarkkuusmitassa että TREC-tyyppisessä pisteytyksessä tulokseksi 1. Jos Tipu antaa vain yhden vastauksen, on TREC-tyyppinen pisteytys 0.5 (eli oikea vastaus on toisella sijalla) ja tarkkuusmitan mukainen pisteytys 0. Jos vastauksia on yksikin kappale, on tarkkuusmitan arvo 0. TREC-tyyppinen pisteytys sitä vastoin pienee sitä mukaa mitä enemmän vastauksia Tipu palauttaa, sillä oikean vastauksen katsotaan olevan viimeisen vastauksen jälkeinen vastaus. Näin saadaan aikaan mitta, joka antaa sitä huonommat pisteet mitä enemmän vastauksia negatiiviseen kysymykseen palautetaan.

Suoritettu evaluointi on vain suuntaa-antava. Kattava evaluointi vaatisi, että koko kysymysvastauskorpus olisi luokiteltu, jolloin evaluointiin voitaisiin käyttää luvussa 5.4 esiteltyä ristiinvalidointia. Tällöin evaluoinnissa ei käytettäisi mitään muuta materiaalia kuin kysymysvastauskorpusta.

Nyt evaluoinnissa käytetyt kysymykset ovat testausta varten luotuja uusia kysymyksiä ja ne on lueteltu liitteessä 5. Kysymykset voisivat aivan hyvin olla testikorpuksen kysymyksiä, sillä ne on luotu tutkimalla tarkkaan valmiita kysymyksiä ja etsimällä sieltä yleisiä kysymystyyppejä. Näitä yleisiä kysymystyyppejä on sitten käytetty pohjana uusia kysymyksiä luotaessa. Koska evaluoinnissa on käytetty vain 33:a kysymystä ja koska korpuksessa on yhteensä noin 24 000 kysymystä, on selvää, että vaikka testikysymykset olisikin poimittu suoraan korpuksesta, olisi korpusta jouduttu tutkimaan tarkkaan edustavien kysymysten löytämiseksi. Korpuksesta valittujen kysymysten edustavuutta olisi yhtä vaikea mitata kuin itse luotujen kysymystenkin edustavuutta. Todettakoon vielä, että testauksessa käytetyt kysymykset eivät sisälly järjestelmän opetuskysymyksiin eli niitä ei ole käytetty järjestelmän vektoritietokantoja luotaessa. Tämä on omiaan antamaan realistisen kuvan järjestelmän suorituskyvystä, sillä jos testikysymykset sisältyi-

sivät opetusjoukkoon, saataisiin evaluoinnissa todennäköisesti epärealistisen hyviä tuloksia.

Termityyppi	Painotus	Keskiarvo	
		tarkkuus	TREC-pisteytys
Baseline	Plain	.27	.41
	tfidf	.30	.30
Lemma	Plain	.34	.51
	tfidf	.38	.42
	Syntax	.45	.61
Term	Plain	.20	.30
	tfidf	.22	.26

Taulukko 13: *Tipun evaluoinnin tulokset. Testatut termityypit ovat baseline, lemma ja term ja painotukset plain,* $tf * idf$ *ja syntax. Tulokset voivat vaihdella nollan ja ykkösen välillä. Tulos on sitä parempi, mitä suurempi luku on.*

Evaluoinnin tulokset ovat taulukossa 13. Evaluoinnissa kaksi Tipun parametria on asetettu oletusarvoista poikkeaviksi. Nämä ovat tekstifragmenttien välisen samanlaisuuden raja-arvo, joka on asetettu 0.5:een sekä tulostettavien vastausten maksimimäärä. Tulostettavien vastausten maksimimääräksi on asetettu viisi. Jos saman samanlaisuusarvon omaavien vastausten joukko ei katkea viiden vastauksen kohdalla, tulostamista jatketaan kunnes seuraavan vastauksen samanalaisuusarvo on eri kuin edellisen. Tässä tapauksessa kymmenen vastausta on ehdoton maksimimäärä. Koska saman samanlaisuusarvon omaavat kysymykset tulostuvat satunnaisessa järjestyksessä, saattaa järjestelmä palauttaa eri vastausjoukon samalle kysymykselle ohjelman eri ajokerroilla silloin, kun samanlaisuusarvo ei ole muuttunut vielä yhdennentoistakaan vastauksen kohdalla.

Tipun evaluointituloksista huomataan, että sekä tarkkuusarvon että TREC-pistey-

tyksen mukaan parhaisiin tuloksiin päästään kun käytetään lemmatisoituja termejä, jotka on painotettu syntaktisen funktion mukaan. Toinen sija on jaettu sija, sillä tarkkuusarvon mukaan se kuuluu lemmatisoiduille $tf * idf$ -painotetuille vektorille ja TREC-tyyppisen pisteytyksen mukaan lemmatisoiduille painottamattomille vektoreille.

Yksityiskohtaisesti evaluoinnin tulokset on kuvattu liitteessä 6. Ne kertovat esimerkiksi sen, että Tipun antamien vastausten määrä vaihtelee paljon eri vektorityyppejä käytettäessä. Vähiten vastauksia palautetaan kun käytetään baselineTfidf-vektoreita (yht. 48 kpl) ja eniten kun käytetään termPlain-vektoreita (yht. 190 kpl). Liitteestä löytyvät myös kysymyskohtaiset tarkkuusarvot ja TREC-pisteytyksen mukaiset arvot. Lisäksi jokaiselle kysymykselle on laskettu, kuinka hyvin siihen keskimäärin löydettiin vastaus. Luvuista huomataan, että kysymysjoukko oli heterogeeninen, sillä se sisälsi seitsemän kysymystä, joihin ei millään testatuista vektorityypeistä löydetty vastausta ja toisaalta neljään kysymykseen löytyi vastaus niin hyvin, että ne saivat täydet pisteet TREC-pisteytyksen mukaan kaikilla vektorityypeillä.

Kokonaisuudessaan voidaan todeta, että tämän alustavan evaluoinnin perusteella Tipun suorituskyky silloin kun se käyttää lemmaSyntax-tyyppisiä vektoreita on suhteellisen hyvä, eli .61 TREC-tyyppisellä pisteytystavalla mitattuna. TREC-9 konferenssin kysymysvastaussarjassa parhaiten menestyneen järjestelmän saama keskimääräinen pistemäärä on .76. Toiseksi parhaiten menestyneen järjestelmän saama pisteytys on .46 [Voo00]. Nämä tulokset on otettu TREC:in 250:n tavun (engl. byte) kilpailusarjasta, mikä tarkoittaa sitä, että oikean vastauksen on löydyttävä 250:n tavun pituisesta tekstifragmentista. TREC:issä on myös vaikeampi kilpailusarja, jossa vastauksen on löydyttävä enintään 50:n tavun kokoisesta fragmentista. Tässä sarjassa vastaavat tulokset olivat .58 ja .32. TREC-kilpailujen evaluointi on tehty samalla tavalla kuin Tipun TREC-tyyppinen evaluointi lukuunot-

tamatta kahta poikkeusta. Ensimmäinen on se, että TREC:issä annetaan pisteitä ainoastaan viidestä ensimmäisestä vastauksesta. Toinen ero on, että TREC:issä ei oteta ollenkaan huomioon negatiivista kysymyksiä, sillä kaikki kysymykset pyritään tekemään sellaisiksi, että niihin löytyisi dokumenttikokoelmasta oikea vastaus.

7 Johtopäätökset

Tutkielmassa tutustuttiin menetelmiin, joilla voidaan tunnistaa tekstifragmenttien välistä samanlaisuutta. Näistä toteutettiin ja evaluoitiin vektorimalliin perustuvia menetelmiä osana kysymysvastausjärjestelmää. Koska monet samanlaisuutta tunnistavat menetelmät perustuvat tekstin leksikaalisen koheesion ja sanojen välisen semanttisen samanlaisuuden tunnistamiseen, käsiteltiin niitä tutkielman luvussa 2. Luvussa 3 tutustuttiin sanojen morfosyntaktiseen analyysiin perustuvaan tapaan poimia lauseesta merkityksen kannalta oleelliset sanat. Luvussa 4 tutustuttiin yksityiskohtaisemmin kahteen menetelmään, joiden avulla tekstifragmenttien välinen samanlaisuus voidaan laskea numeerisesti. Luvun 5 menetelmät eri algoritmien tehokkuuden mittaamiselle loivat pohjan tutkielman kokeelliselle osalle, jossa menetelmiä sovellettiin todelliseen aineistoon ja mitattiin niiden paremmuutta toisiinsa nähden.

Tutkielman teoreettisen osan saavutus on, että se toimii johdatuksena leksikaaliseen koheesioon perustuviin tekstifragmenttien samanlaisuutta tunnistaviin menetelmiin. Lisäksi se viitoittaa useita eri suuntia jatkotutkimukselle. Tämä olikin tarkoitus, sillä APPA-projektin alkuvaiheen päämäärä oli kysymysvasvastausjärjestelmiin ja niihin liittyviin ongelmiin ja menetelmiin tutustuminen kysymysvastausjärjestelmän prototyypin toteutuksen ohella.

Tutkielman kokeellisessa osassa toteutettiin Tipu-niminen kysymysvastausjärjes-

telmän prototyyppi, jonka avulla evaluoitiin erilaisia vektoripohjaisia menetelmiä tekstifragmenttien välisen samanlaisuuden tunnistamiseksi käyttäen aineistona todellisia kysymyksiä ja vastauksia. Kokeellisen osan saavutus on, että teoreettisessa osassa esitettyjä menetelmiä testattiin käytännössä hyvin haastavalla materiaalilla, sillä aineisto koostui hyvin heterogeenisesta kielenkäytöstä ja sen aihepiiri oli erittäin erikoistunut. Tämän lisäksi kokeellisen osan ansioksi voidaan lukea se, että käytössä oli suomenkielistä eikä englanninkielistä aineistoa kuten suuressa osassa alan tutkimuksista. Eri menetelmien alustavasta evaluoinnista saadut tulokset kertonevat vähintäänkin yhtä paljon siitä, mitkä menetelmät sopivat parhaiten käyttämällemme tekstityypille kuin siitä, mitkä menetelmät ovat yleisesti ottaen parhaita.

Tutkielma viitoittaa kaksi mahdollista suuntaa jatkotutkimukselle ja APPA-projektille. Ensimmäinen on tutkielmassa esiteltyjen numeeriseen laskentaan ja tekstin koheesioon perustuvien menetelmien kehittäminen edelleen. Tekstin koheesioon perustuvat menetelmät hyötyisivät todennäköisesti ilmausten koreferenssiä eli samaviitteisyyttä[4] ratkovasta algoritmista, monisanaisten termien ja fraasien tunnistamisalgoritmista sekä menetelmästä, joka muodostaa semanttisen verkon tai ontologian dokumenttikokoelman käsitteistä ja termeistä. Jatkotutkimuksessa näitä kolmea komponenttia voitaisiin kehittää itse tai jos ne olisivat saatavilla valmiina, voitaisiin selvittää, parantaako niiden käyttö Tipu-prototyypin suorituskykyä. Koska Tipussa toteutettiin ainoastaan vektorimalliin perustuva fragmenttien välistä samanlaisuutta laskeva komponentti, olisi mielenkiintoista toteuttaa myös leksikaalisia ketjuja käyttävä komponentti ja verrata sen suorituskykyä jo toteutetun komponentin suorituskykyyn.

Toinen tekstifragmenttien semanttisen samanlaisuuden tunnistamiseen liittyvä

[4]Samaviitteisiä ovat esim. *Halonen* ja *hän* seuraavissa virkkeissä. Tapasin *Halosen*. *Hän* oli juuri palannut Brasiliasta.

jatkotutkimuksen aihe on jonkinasteinen tekstin ymmärtäminen päättelysääntöjen avulla ja niihin perustuvan komponentin toteuttaminen. Tällaisen komponentin sisältää mm. Falcon-kysymysvastausjärjestelmä, joka menestyi ylivoimaisesti parhaiten TREC-9:ssä [Voo00]. Falconin perustana on leksikaalista koheesiota tunnistava komponentti, jonka päälle on rakennettu loogista päättelyä suorittava moduuli [HMP+00]. Päättelysääntöihin perustuva lähestymistapa semantiikkaan on lähempänä perinteistä tekoälytutkimusta kun taas tässä tutkielmassa suurimman painoarvon saanut vektorimalliin pohjautuva lähestymistapa on peräisin tiedonhakututkimuksesta.

Tekstifragmenttien välisen samanlaisuuden tunnistaminen on monitahoinen tehtävä, jolle löytyy käyttöä monessa luonnollista kieltä käsittelevässä sovelluksessa. Vaikka aihetta on tutkittu jo melko kauan ja eri näkökulmista, on siinä vielä paljon selvitettävää. Tämä tutkimus osoitti, että fragmenttien välistä semanttista samanlaisuutta tyydyttävästi tunnistava menetelmä voi olla yksinkertainen. Jos kutenkin halutaan hyviä tai erinomaisia tuloksia, vaikuttaa siltä, että tehtävästä tulee hyvinkin monimutkainen ja haastava.

Lähteet

BB99 Patrick Blackburn and Johan Bos. *Representation and Inference for Natural Language, A First Course in Computational Semantics*, volume 1. September 1999. `http://www.coli.uni-sb.de/~bos/comsem/book1.html[8.11.2001]`.

BDS93 Richard Boyd, Jamse Driscoll, and Mien Syu. Incorporating semantics within a connectionist model. In *Proceedings of TREC-2*, 1993.

BE97 Regina Barzilay and Michael Elhadad. Using lexical chains for text

summarization. In *Proceedings of the intelligent scalable text summarization workshop*, Madrid, Spain, 1997. Association for Computational Linguistics.

BYRN99 Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, 1999.

Con01a Fdg with visual output, 2001. `http://www.conexoroy.com/products.htm[12.9.2001]`.

Con01b Output of Conexor analysers for Finnish, 2001. `http://www.conexoroy.com//docs/fi-tags.html[12.9.2001]`.

Fel98a Christiane Fellbaum. *Introduction*. In [Fel98c], 1998.

Fel98b Christiane Fellbaum. *A Semantic Network of English Verbs*, chapter 3. In [Fel98c], 1998.

Fel98c Christiane Fellbaum, editor. *WordNet : an electronic lexical database*. MIT Press, 1998.

Fin01 Korkeimman oikeuden asiasanasto, 2001. `http://www.finlex.fi/oikeus/asiasanasto.html[19.11.2001]`.

GM89 Gerald Gazdar and Chris Mellish. *Natural language processing in Prolog : an introduction to computational linguistics*. Addison-Wesley, Wokingham, 1989.

GM98 Derek Gross and Katherine J. Miller. *Modifiers in WordNet*, chapter 2. In Fellbaum [Fel98c], 1998.

Hea97 Marti A. Hearst. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64, March 1997.

HH76 Michael A. K. Halliday and Ruqaiya Hasan. *Cohesion in English*. Longman, 1976.

HK01 Jiawei Han and Micheline Kamber. *Data Mining, Concepts and Techniques*. Morgan Kaufmann Publishers, 2001.

HMP+00 S Harabagiu, D. Moldovan, M. Paşca, R. Mihalcea, M. Surdeanu, R. Bunescu, R. Gîrju, V. Rus, and P. Morărescu. Falcon: Boosting knowledge for answer engines. In *Proceedings of TREC-9*, 2000.

HO93 Auli Hakulinen and Jussi Ojanen. *Kielitieteen ja fonetiikan termistöä*. Suomalaisen Kirjallisuuden Seura, Helsinki, 1993.

HSO98 Graeme Hirst and David St-Onge. *Lexical chains as representation of context for the detection and correction of malapropisms*, chapter 13. In Fellbaum [Fel98c], 1998.

JM00 Daniel Jurafsky and James H. Martin. *Speech and language processing. An Introduction to natural language processing, computational linguistics and speech recognition*. Prentice Hall, 2000.

Kar98 Fred Karlsson. *Yleinen kielitiede*. Yliopistopaino, Helsinki, 1998.

KKM98 Min-Yen Kan, Judith L. Klavans, and Kathleen R. McKeown. Linear segmentation and segment significance. In *Proceedings of the Sixth Workshop on Very Large Corpora*, Montreal, Quebec, Canada, August 1998. Association for Computational Linguistics.

KR00 Adam Kilgarriff and Joseph Rosenzweig. English SENSEVAL: Report and results. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation*, Athens, Greece, 2000. The European Language Resources Association.

Kuu01 Reeta Kuuskoski. Kysymys-vastaus-järjestelmät. Master's thesis, Department of Computer Science, University of Helsinki, Finland, 2001.

Lah00 Timo Lahtinen. *Automatic indexing: an approach using an index term corpus and combining linguistic and statistical methods.* PhD thesis, Department of General Linguistics, University of Helsinki, Finland, 2000.

Lew91 Davis D. Lewis. Evaluating text categorization. In *Proceedings of the Speech and Natural Language Workshop*, February 1991.

MH91 Jane Morris and Graeme Hirst. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1):21–48, March 1991.

Mil95 George A. Miller. Wordnet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, November 1995.

Mil98 George A. Miller. *Nouns in WordNet*, chapter 1. In Fellbaum [Fel98c], 1998.

MIO00 Hajime Mochizuki, Makoto Iwayama, and Manabu Okumura. Passage-level document retrieval using lexical chains. In *Proceedings of the 6th Conference on content-based multimedia information access*, Paris, France, April 2000. Computer-Assisted Information Retrieval (RIAO).

MS00 Christopher D. Manning and Hinrich Schütze. *Foundations of statistical natural language processing.* The MIT Press, 2000.

SB87 Gerard Salton and Chris Buckley. Term weighting approaches in automatic text retrieval. Technical Report 87-881, Cornell University, Ithaca, New York, November 1987.

Seb02 Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 2002. Accepted for publication.

SM00 Gregory H. Silber and Kathleen F. McCoy. Efficient text summarization using lexical chains. In *Proceedings of the ACM Conference on Intelligent User Interfaces*, New Orleans, LA, January 2000.

Tap99 Pasi Tapanainen. *Parsing in two frameworks: finite-state and functional dependency grammar*. PhD thesis, Department of General Linguistics, University of Helsinki, Finland, December 1999.

Tek89 Tekniikan sanastokeskus, toim. Sanastotyön käsikirja. Soveltavan terminologian periaatteet ja työmenetelmät. Suomen standardisoimisliitto, Helsinki, 1989.

TJ97 Pasi Tapanainen and Timo Järvinen. A non-projective dependency parser. In *Proceedings of the 5th conference on applied natural language processing*, Washington, D.C., April 1997. Association for Computational Linguistics.

VH99 Ellen M. Voorhees and D. K. Harman, editors. *NIST Special Publication 500-246: The Eighth Text REtrieval Conference (TREC-8)*, Gaithersburg, Maryland, November 1999. Department of Commerce, National Institute of Standards and Technology.

VH00 Ellen M. Voorhees and D. K. Harman, editors. *NIST Special Publication 500-249: The Ninth Text REtrieval Conference (TREC-9)*, Gaithers-

burg, Maryland, November 2000. Department of Commerce, National Institute of Standards and Technology.

Voo99 Ellen M. Voorhees. The TREC-8 Question Answering Track Report. In Voorhees and Harman [VH99].

Voo00 Ellen M. Voorhees. Overview of the TREC-9 Question Answering Track. In Voorhees and Harman [VH00].

Vos98 Piek Vossen. Introduction to EuroWordNet. *Computers and the Humanities*, 32(2):73–89, 1998.

vR80 C. J. van Rijsbergen. *Information Retrieval*. Butterworths, 2nd edition, 1980.

Woo73 William A. Woods. Progress in natural language understanding: an application to lunar geology. In *AFIPS Conference Proceedings, 42*, pages 441–50, New York, June 1973. American Federation of Information Processing Societies.

Wor01 Wordnet a lexical database for the English language, 2001. `http://www.cogsci.princeton.edu/\~wn/[17.9.2001]`.

Yan97 Yiming Yang. An evaluation of statistical approaches to text categorization. Technical Report CMU-CS-97-127, Carnegie Mellon University, 1997.

YP97 Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, 1997.

Liite 1. Esimerkki kysymysvastauskorpuksesta

Esimerkki tutkielman kokeellisessa osassa käytetystä kysymysvastauskorpuksesta. Korpuksen muoto on säilytetty, mutta sisältö on korvattu osoitteesta http://sal-faq.sourceforge.net/other-formats/sal-faq.html poimituista kysymyksistä ja vastauksista.

```
<user id="0"><pair>
<question>
Mikä on Gnome?
</question>
<answer>
Gnome on käyttäjäystävällinen työpöytäympäristö Unixiin, siis myös
Linuxiin. Se tulee mukana useimmissa levityspaketeissa. Gnome
vaatii vähintään 32-64 megatavua muistia.
</answer>
</pair></user>
<user id="1"><pair>
<question>
Miten etsin tiedostoja?
</question>
<answer>
Tiedostoja etsitään käskyllä find.
</answer>
</pair></user>
```

Liite 2. Esimerkki jäsennetystä korpuksesta

Analyysin voi suorittaa Conexor Oy:n www-sivulla [Con01a].

```
1 Mikä mikä subj:>2 &NH PRON SG NOM
2 on olla main:>0 &+MV V ACT IND PRES SG3
3 Gnome gnome comp:>2 &NH <?> N SG NOM
4 ? ?
5 <s><s>>4
1 Gnome gnome subj:>2 &NH <?> N SG NOM
2 on olla main:>0 &+MV V ACT IND PRES SG3
3 käyttäjäystävällinen käyttäjä#ystävällinen attr:>4 &A> A SG NOM
4 työpöytäympäristö työ#pöytä#ympäristö comp:>2 &NH N SG NOM
5 Unixiin unix goa:>2 &NH N SG ILL
6 ,,
7 siis siis meta:>12 &ADV ADV
8 myös myös &ADV ADV
9 Linuxiin linux &NH N SG ILL
10 . .
11 Se se subj:>12 &NH PRON SG NOM
12 tulee tulla &+MV V ACT IND PRES SG3
13 mukana mukana &ADV ADV
14 useimmissa usea attr:>15 &A> A SUP PL INE
15 levityspaketeissa levitys#paketti loc:>12 &NH N PL INE
16 . .
17 <s><s>>16
1 Gnome gnome subj:>2 &NH <?> N SG NOM
2 vaatii vaatia main:>0 &+MV V ACT IND PRES SG3
```

```
3 vähintään vähintään ad:>4 &AD> ADV
4 32-64 32-64 qn:>5 &QN> NUM CARD
5 megatavua mega#tavu &NH N SG PTV
6 muistia muisti &NH N SG PTV
7 . .
1 Miten miten man:>2 &ADV ADV
2 etsin etsiä &+MV V ACT IND PRES SG1 &+MV V ACT IND PAST SG1
3 tiedostoja tiedosto obj:>2 &NH N PL PTV
4 ? ?
5 <s><s>>4
1 Tiedostoja tiedosto obj:>2 &NH N PL PTV
2 etsitään etsiä main:>0 &+MV V PASS IND PRES
3 käskyllä käsky loc:>2 &NH N SG ADE
4 find find &NH <?> N SG NOM
5 . .
6 <s><s>>5
```

Liite 3. Conexor Oy:n jäsentimen käyttämä merkkaus

Merkkauksen selitys löytyy myös Conexor Oy:n www-sivuilta [Con01b].

Suomen pintasyntaktiset funktiot:

Funktio	Selitys	Esimerkki
&NH	nominaalinen pääsana	*aurinko* paistaa
&A>	etuattribuutti	*valkoinen* hevonen
&ADV	adverbi	*tänään* sataa
&AD>	ad-adverbi	*liian* paljon
&CC	rinnastuskonjunktio	Teo *ja* Kai
&CS	alistuskonjunktio	*jos* sataa
&QN>	kvantifikaattori	*kolme* porsasta
&+MV	finiittinen verbi	Teo *juoksee*
&-MV	ei-finiittinen verbi	oli *juossut*
&PM	pre- tai postpositio	huvin *vuoksi*

Suomen funktionaaliset riippuvuuskuvaimet:

Kuvain	Nimi	*Esimerkki*
main	pääelementti	*Sataa* . *Todellako?*
pm	merkitsin	Sen *vuoksi* tiesin *että* sataisi.
phr	fraasi	Kielto *on voimassa* .
subj	subjekti	*Linnut* lentävät.
obj	objekti	Ostin *punaisen takin* .
comp	komplementti (predikatiivi)	Takki on *punainen* . Olen *juossut* .
dat	datiiviobjekti	Annoitko sen *hänelle* ?
oc	objektikomplementti	Hänet nimitettiin *maaherraksi* .
copred	kopredikatiivi	Juotko kahvisi *mustana* ?
voc	vokatiivi	*Pekka* , tulisitko tänne!
tmp	aika	*Tänään* ei syödä kalaa.
dur	kesto	Jaana oli lomalla *viisi viikkoa* .
frq	taajuus	Olen ollut siellä *viidesti* .
qua	määrä	Kurssit nousivat *kolme prosenttia* .
man	tapa	Hän pukeutuu aina *tyylikkäästi* .
loc	paikka	Eiffel-torni on *Pariisissa.*
sou	lähde, alkuperä	Puuvillaa tuodaan *Intiasta* .
goa	kohde, tulos	Vein hänet *kotiin* .
pur	tarkoitus	Emme elä *syödäksemme* .
rsn	syy	*Miksi* hän ei ole jo tullut?
cnd	ehto	Kysykää, *jos ette ymmärrä* .
meta	lauseadverbiaali	*Valitettavasti* emme olleet kotona.
qn	kvantifioiva määre	Jaksatko syödä *neljä* perunaa.
attr	etumääre	*Pöydän* pinta on *ruskeaa* tammea.
mod	jälkimääre	Se, *joka pelkää* , ei pelaa.
ad	ad-adverbi	Onnistuimme *aika* hyvin.
cc	rinnastus	Näin pari lokkia *ja yhden sorsan* .

Suomen substantiivien morfologisiset koodit:

Sanaluokka	Alakategoriat	Selitys	esimerkkisana
N		substantiivi	
– luku	SG	yksikkö	*takki, takkia, takissa*
	PL	monikko	*takit, takkeja, takeissa*
– sija	NOM	nominatiivi	*takki, takit*
	GEN	genetiivi	*takin, takkien*
	PTV	partitiivi	*takkia, takkeja*
	INE	inessiivi	*takissa, takeissa*
	ELA	elatiivi	*takista, takeista*
	ILL	illatiivi	*takkiin, takkeihin*
	ADE	adessiivi	*takilla, takeilla*
	ABL	ablatiivi	*takilta, takeilta*
	ALL	allatiivi	*takille, takeille*
	ESS	essiivi	*takkina, takkeina*
	TRA	translatiivi	*takiksi, takeiksi*
	ABE	abessiivi	*takitta, takeitta*
	COM	komitatiivi	*takkeineen*
	INS	instruktiivi	*takein*
	PRO	prolatiivi	*postitse, kirjeitse*
– possessiivisuffiksi	POSS:SG1	possessiivi, SG1	*takki**NI***
	POSS:SG2	possessiivi, SG2	*takki**SI***
	POSS:3	possessiivi, 3. pers.	*takki**NSA***
	POSS:PL1	possessiivi, PL1	*takki**MME***
	POSS:PL2	possessiivi, PL2	*takki**NNE***
– kliittiset muodot,	ks. viimeinen	taulukko	

Suomen adjektiivien, numeraalien ja pronominien morfologiset koodit:

Sanaluokka	Alakategoriat	Selitys	esimerkkisana
A		adjektiivi	
	CMP	komparatiivimuoto	*yleisempi, myöhemmät*
	SUP	superlatiivimuoto	*tuoreimpia, harmain*
	– luku	ja sija kuten	substantiiveilla
NUM		numeraali	
	CARD	perusluku	*26 400 000, miljoona,*
	ORD	järjestysluku	*kolmas, 3., III*
	– luku	ja sija kuten	substantiiveilla
PRON		pronomini	
– sija:	ACC	akkusatiivi	*hänet, meidät*
	– muuten	sija kuten	substantiiveilla
– numerus:	SG	yksikkö	*joku, muun*
	SG1	yksikön 1. persoona	*minut*
	SG2	yksikön 2. persoona	*sinua*
	SG3	yksikön 3. persoona	*häneen*
	PL	monikko	*jotkut, muiden*
	PL1	monikon 1. persoona	*me*
	PL2	monikon 2. persoona	*teille*
	PL3	monikon 3. persoona	*heistä*

Suomen verbien morfologiset koodit:

Sanaluokka	Alakategoriat	Selitys	esimerkkisana
V		verbi	
– pääluokka:	ACT	aktiivi	*pöllysi, muodostuisi, panostanut*
	PASS	passiivi	*tultiin, toteutettu, sijoitettava*
– modus:	IND	indikatiivi	*tulimmehan, ajatellaan*
	KOND	konditionaali	*tultaisi, olisitko*
	POT	potentiaali	*saapunee*
	IMP	imperatiivi	*ostettako, mene*
– tempus:	PRES	preesens	*tulevatko, avataan*
	PAST	imperfekti	*näimme, mentiinhän*
– numerus:	SG	yksikkö	*tehnyt, laskenut*
	SG1	yksikön 1. persoona	*teinkin, lasken*
	SG2	yksikön 2. persoona	*tekisit, laske*
	SG3	yksikön 3. persoona	*tehneekö, laskekoon*
	PL	monikko	*tehneet, laskeneet*
	PL1	monikon 1. persoona	*teimmekin, laskenemme*
	PL2	monikon 2. persoona	*tekisitte, laskekaa*
	PL3	monikon 3. persoona	*tekevät, laskisivat*
– infinitiivit:	INF1	1. infinitiivi	*osoittaa*
	INF2	2. infinitiivi	*säilyttäen, julkaistaessa*
	INF3	3. infinitiivi	*kokoamalla, selailemassa*
	INF4	4. infinitiivi	*jäämistä, siirtäminen*
	INF5	5. infinitiivi	*lähtemäisillään*
– partisiipit:	PCP1	1. partisiippi	*sisältävä, istuttava*
	PCP2	2. partisiippi	*tapahtunut, rakennettukaan*
– infinitiivien	ja	partisiippien luku	ja sija kuten substantiiveilla

Suomen adverbien, post- ja prepositioiden, konjunktioiden, interjektioiden ja muiden edellä mainitsemattomien sanojen morfologiset koodit:

Sanaluokka	Alakategoriat	Selitys	esimerkkisana
ADV		adverbi	*huoleti*
PSP		postpositio	*päiten, taa*
PRE		prepositio	*ilman, kesken, sitten, vastoin*
CS		alistuskonjunktio	*että, jos*
CC		rinnastuskonjunktio	*mutta, sekä*
INTERJ		interjektio	*jaaha, jee*
Muita koodeja	ja apupiirteitä		
Kliittiset muodot	-KIN		*juokseekin, talokin*
	-KA		*eikä*
	-KO		*onko, viekö*
	-PA		*olepa, etsipä*
	-PI		*ompi, viepi*
	-HAN		*olihan*
	-KAAN		*viekään*
	-S		*tules, tulepas*

Liite 4. Conexor Oy:n termitunnistimen tulostusta

```
Mikä on <maxterm base="gnome"> Gnome </maxterm> ?

<maxterm base="gnome"> Gnome </maxterm> on
<maxterm base="käyttäjäystävällinen\_työpöytäympäristö">
käyttäjäystävällinen työpöytäympäristö </maxterm>
<maxterm base="unix"> Unixiin </maxterm>,
siis myös <maxterm base="linux"> Linuxiin </maxterm> .

Se tulee mukana <maxterm base="usea\_levityspaketti">
useimmissa levityspaketeissa </maxterm> .

<maxterm base="gnome"> Gnome </maxterm> vaatii vähintään 32-64
<maxterm base="megatavu"> megatavua </maxterm>
<maxterm base="muisti"> muistia </maxterm> .
```

Liite 5. Evaluoinnissa käytetyt kysymykset

Evaluoinnissa käytetyt 33 kysymystä on muodostettu 24 000 kysymystä käsittävästä korpuksesta etsimällä edustavia kysymystyyppejä ja muokkaamalla niitä.

1 Paljonko maksaa soittaa Soneran liittymästä Radiolinjaan? Entä lankapuhelimeen?
2 Miten puheluista saa bonusta?
3 Miten ulkomailta soitetaan Suomeen?
4 Miksi mun WAP ei toimi?
5 Miten saan puhelimen saldon?
6 Paljonko maksaa vastaajan viestien purkamienen?
7 Mistä numerosta voi tilata logoja?
8 Tahtoisin siirtää laskuni suoraveloitukseen. Miten onnistuu?
9 Kenen numero on 040-1234567
10 Miksei FIND toimi?
11 Miten saan selville soittajan nimen?
12 Mikä on saldosopimus?
13 Voiko laskun määrää rajoittaa etukäteen? Miten?
14 Ottakaa saldoraja pois numerosta 040-2345678.
15 Miten pääsen eroon chat-kanavasta?
16 Voiko chat-nimimerkin henkilöllisyyden saada selville?
17 Kytkekää kotisoitto: 091234567 0403456789 0404567890. Kiitos!
18 Miksi mun nettiliittymä lopetti toimimasta?
19 Miten niitä lahjoja tilataan kännykällä?
20 Osoitteenmuutos: asiakasnumero 111111, Risto Reipas, uusi osoite Kaupunkikatu 1, 00100 Helsinki.
21 Onko kauas pitkä matka?
22 Maksaako soitonsiirto Soneran liittymästä Telian liittymään? Siirtäjälle vai Soittajalle?

23 Mikä on seuraavan laskuni saldo?

24 miten saan bussiaikataulun puhelimeeni?

25 mikä on mun puk-koodi?

26 Mikä on teidän tilinumero?

27 Paljonko maksaa tekstiviestin lähettäminen Makedoniaan?

28 Kuinka tilaan soittoäänen toiseen liittymään?

29 Mikä on vastaajani numero?

30 Haluaisin muuttaa numeroni salaiseksi. Kuinka se tehdään? Onnistuuko tätä kautta?

31 Miten estän numeron näkymisen vastaanottajalle?

32 Voiko numeron näkymisen tekstiviestissä estää?

33 Mikä on edullisin espanjalainen operaattori? Paljonko sen puhelut maksaa Suomeen?

Liite 6. Evaluoinnin tulokset yksityiskohtaisesti

Tarkkuus

Kysymykset 1 - 14. Numero kertoo, kuinka monta vastausta on oikein ja kuinka monta vastausta saatiin. Lisäksi jokaiselle kysymykselle on laskettu keskimääräinen tarkkuus. Jos kysymykseen ei saatu vastausta, merkitään positiivisille kysymyksille 0 ja negatiivisille kysymyksille 1.

Kysymys	Baseline		Lemma			Term		Kysymyksen
Nro	Plain	Tfidf	Plain	Tfidf	Syntax	Plain	Tfidf	Keskiarvo
1	0/1	0/1	0/5	0/2	0/8	1/10	0/5	.01
2	3/10	4/4	4/10	5/5	4/5	3/10	4/5	.66
3	0	0	2/5	0/5	2/5	0/10	0/5	.11
4	0/4	0	0/9	0/1	1/5	0	0	.03
5	3/7	4/4	3/6	3/5	3/5	0	0	.45
6	1/4	0	4/5	0	3/7	0/7	0	.21
7	1/5	4/5	1/10	2/6	1/5	2/7	4/10	.33
8	0	0	0/1	1/1	1/10	1/10	2/5	.23
9	6/6	0	9/9	0	10/10	5/10	4/10	.56
10	0/3	0/3	1/6	0/10	0/5	0	0	.02
11	3/5	5/5	4/5	3/5	4/5	5/6	4/5	.78
12	0/10	0/3	0/10	0/7	3/8	2/10	2/10	.11
13	0	0	0	0	0/1	0	0	0
14	0	0	0	0	0/6	0/6	0/6	0

Tarkkuus

Kysymykset 15 - 33 ja eri vektorityyppien tarkkuuden keskiarvo.

Sulkuihin merkittyyn keskarvoon ei ole otettu mukaan negatiivisia kysymyksiä.

Kys.	Baseline		Lemma			Term		Kys.
Nro	Plain	Tfidf	Plain	Tfidf	Syntax	Plain	Tfidf	Keskiarvo
15	0	0	0/5	0	0/3	0	0	0
16	0	0	0/4	0/3	0/1	0	0	0
17	0	0	0/8	2/10	4/5	4/10	3/5	.29
Neg. 18	1	1	1	1	1	1	1	1
19	0	0	0/7	0/4	0/2	0/10	0/5	0
20	0	0	2/10	0	5/5	0	0	.17
Neg. 21	1	1	0/1	1	0/1	0/5	0/5	.43
22	0	0	4/6	1/2	2/5	2/4	1/1	.44
23	7/10	0	10/10	0/6	9/9	8/10	0/5	.5
24	0/9	0	0/8	0	0/5	0/10	0	0
25	4/10	10/10	5/10	10/10	5/5	0	0	.56
26	2/10	4/4	3/5	5/5	5/5	4/5	4/5	.77
27	2/6	0	2/7	0	2/5	1/10	0	.16
28	0	0	0/4	0/5	0/10	0/10	0/10	0
29	4/10	5/5	10/10	10/10	8/8	6/10	1/10	.73
30	0	0	6/6	5/5	6/6	1/10	0/8	.44
31	3/3	3/3	4/5	5/5	5/6	0	0	.66
32	0	0	1/7	2/3	1/5	0	0	.14
33	1/1	0/1	0/5	1/2	1/5	0/10	0	.24
yht.	40/114	39/48	75/119	55/117	85/176	45/190	29/115	-
Keskiarvo	.27	.30	.34	.38	.45	.20	.22	.30
(ei neg.)	(.22)	(.25)	(.33)	(.34)	(.44)	(.18)	(.18)	(.28)

TREC-tyyppinen pisteytys Kysymykset 1 - 14.								
Kysymys	Baseline		Lemma			Term		Kysymys
Nro	Plain	Tfidf	Plain	Tfidf	Syntax	Plain	Tfidf	Keskiarvo
1	0	0	0	0	0	0.17	0	.02
2	1	1	1	1	1	1	1	1
3	0	0	1	0	1	0	0	.29
4	0	0	0	0	0.2	0	0	.03
5	1	1	0.5	1	1	0	0	.64
6	0.5	0	1	0	1	0	0	.36
7	0.5	1	0.14	0.33	0.25	1	0.5	.53
8	0	0	0	1	1	1	1	.57
9	1	0	1	0	1	1	0.14	.59
10	0	0	0.33	0	0	0	0	.05
11	1	1	1	1	1	1	1	1
12	0	0	0	0	0.33	0.14	0.5	.14
13	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0

TREC-tyyppinen pisteytys Kysymykset 14 - 33 ja eri vektorityyppien keskiarvot.								
Kysymys	Baseline		Lemma			Term		Kysymys
Nro	Plain	Tfidf	Plain	Tfidf	Syntax	Plain	Tfidf	Keskiarvo
15	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0
17	0	0	0	0.5	1	0.33	0.33	.31
Neg. 18	1	1	1	1	1	1	1	1
19	0	0	0	0	0	0	0	0
20	0	0	1	0	1	0	0	.29
Neg. 21	1	1	0.5	1	0.5	0.17	0.17	.62
22	0	0	1	1	1	0.5	1	.64
23	1	0	1	0	1	0.5	0	.5
24	0	0	0	0	0	0	0	0
25	1	1	1	1	1	0	0	.71
26	1	1	1	1	1	1	1	1
27	0.5	0	1	0	0.33	0.1	0	.28
28	0	0	0	0	0	0	0	0
29	1	1	1	1	1	0.33	0.14	.78
30	0	0	1	1	1	0.5	0	.5
31	1	1	1	1	1	0	0	.71
32	0	0	0.5	0.5	0.5	0	0	.21
33	1	0	0	0.5	1	0	0	.36
Keskiarvo	.41	.30	.51	.42	.61	.30	.26	.40
(ei neg.)	(.37)	(.26)	(.50)	(.38)	(.60)	(.28)	(.21)	(.37)