Erkki Mäkinen (toim.)

Tietojenkäsittelytieteellisiä tutkielmia
Kevät 2015

INFORMAATIOTIETEIDEN YKSIKKÖ
TAMPEREEN YLIOPISTO

INFORMAATIOTIETEIDEN YKSIKÖN RAPORTTEJA 36/2015

TAMPERE 2015

TAMPEREEN YLIOPISTO
INFORMAATIOTIETEIDEN YKSIKKÖ
INFORMAATIOTIETEIDEN YKSIKÖN RAPORTTEJA 36/2015
HUHTIKUU 2015

Erkki Mäkinen (toim.)

Tietojenkäsittelytieteellisiä tutkielmia
Kevät 2015

INFORMAATIOTIETEIDEN YKSIKKÖ
33014 TAMPEREEN YLIOPISTO

ISBN 978-951-44-9810-7 (pdf)

ISSN-L 1799-8158
ISSN 1799-8158

Sisällysluettelo

Käyttäjien osallistamisen hyödyt ohjelmistosuunnittelussa

Veera Kuusikko

Tiivistelmä.
Käyttäjät tuntevat parhaiten käyttämänsä ohjelmistot ja niiden käyttökontekstit. Käyttäjien osallistamisella suunnitteluun onkin suuri vaikutus ohjelmiston menestykseen sekä käyttäjien tyytyväisyyteen ohjelmistoa kohtaan. Lisäksi sillä on suuri vaikutus myös vaatimusten laatuun. Käyttäjien osallistaminen ei kuitenkaan ole ongelmatonta, vaan siinä on omat haasteensa. Tässä tutkielmassa perehdyn osallistamisen hyötyihin sekä siihen, miten näitä haasteita voidaan ehkäistä.

Avainsanat ja -sanonnat: käyttäjien osallistaminen, ohjelmistokehitys, osallistamisen hyödyt, käytettävyys.

1. Johdanto

Käyttäjien osallistamisella tarkoitetaan perinteisesti käyttäjien ”edustamista” (representing), jolloin käyttäjät eivät osallistu aktiivisesti ohjelmiston kehitykseen [Iivari, 2006]. Käyttäjien edustamisella tarkoitetaan asiantuntijan käyttäjästä laatimaa kuvausta, joka perustuu hänen aiempaan tietämykseensä ja oletuksiin järjestelmän käyttäjistä ja heidän käyttötavoistaan. Tällaisia kuvauksia ovat esimerkiksi persoonat sekä käyttötapaukset ja -kuvaukset (skenaariot). Vaikka käyttäjät voivat usein tarjota suunnittelijoille tietoa ja kommentoida aiemmin tehtyjä suunnitteluratkaisuja sekä osallistua testaukseen testihenkilöinä, eivät he voi osallistua aktiivisesti suunnitteluun tai vaikuttaa päätöksen tekoon. Riittääkö se, vai tulisiko käyttäjät ottaa oikeasti mukaan suunnitteluun?

Pankowskan [2012] mukaan on olemassa neljä syytä, miksi käyttäjät kannattaa ottaa mukaan suunnitteluun. Ensimmäiseksi, käyttäjät eivät ole hyviä kertomaan, mitä he tekevät ja haluavat. Tämän takia tarvitaan avuksi asiantuntijoita, jotka keräävät tietoa käyttäjistä kysymysten avulla. Toiseksi, käyttäjät haluavat olla avuksi ja raportoida järjestelmän käytössä ilmenevistä ongelmista ja hankaluuksista. Kolmanneksi, he eivät ole kehitysryhmän jäseniä, koska heillä on omat työnsä, jolloin he eivät ole lähellä kehitystiimiä. Tällöin heidän mielipiteensä ovat heidän omiaan, eikä sidottuja kehitystiimin mielipiteisiin. Neljänneksi, järjestelmän oikeat käyttäjät ovat kehitykselle arvokkaampia kuin markkinoinnin avulla saavutettavat ”korvikekäyttäjät”.

Käyttäjät nähdään usein tärkeänä sidosryhmänä, sillä he maksavat järjestelmästä [Kujala *et al.*, 2005; Heiskari and Lehtola, 2009]. Käyttäjien tavoite on yleisesti ostaa järjestelmä, joka tukee heidän tehtäviään. He ovat myös järjes-

telmän todellisia käyttäjiä, jolloin heidän huomioon ottaminen suunnittelussa on tärkeää. Myös Abeleinin ja muiden [2013; myös Bano and Zowghi, 2014] mukaan käyttäjien osallistaminen on äärimmäisen tärkeää, koska käyttäjät tuntevat niin työ- kuin kotikäyttökontekstit, joita kyseisen ohjelmiston tulisi tukea.

Käyttäjien osallistamisella on keskeinen rooli kehitysprosessin vaiheissa, joissa kartoitetaan käyttäjien tarpeita, toimintoja tai parannetaan käytettävyyttä [Hess *et al.*, 2013]. Yleisesti, käyttäjien odotetaan auttavan ohjelmistoja kehittäviä yrityksiä ymmärtämään heidän tarpeitaan, tarjoavan erilaisia palveluita, vähentämään kehitykseen tarvittavaa aikaa, johtamaan käyttäjien opastusta sekä kehittämään innovaatioiden leviämistä [Pankowska, 2012].

Käyttäjien osallistaminen voidaankin nähdä erilaisten käyttäjien ja suunnittelijoiden välisenä suhteena, joka tähtää käyttäjien voimaantumiseen (empowerment) [Iivari, 2006]. Voimaantuminen voi olla joko demokraattista tai toiminnallista. Demokraattisella voimaantumisella tarkoitetaan sitä, että käyttäjällä on mahdollisuus osallistua päätöksentekoon, ja toiminnallisella voimaantumisella sitä, että käyttäjä pystyy suorittamaan tehtävänsä tehokkaasti [Iivari, 2006].

Iivarin [2006] mukaan käyttäjien keskeisestä roolista huolimatta ohjelmistoja kehittävät yritykset ovat usein tietämättömiä ohjelmistojensa käyttäjistä ja käyttökonteksteista eli tilanteista, joissa ohjelmistoa käytetään. Usein kehitys tapahtuukin ilman käyttäjien osallistumista siihen tai käyttäjät otetaan kehitykseen mukaan liian myöhään. Tällöin käyttäjillä ei enää ole mahdollisuutta vaikuttaa lopulliseen ohjelmistoon. Näihin ongelmiin käyttäjien osallistaminen voi tarjota vastauksia.

Miksi osallistamista ei sitten tehdä? Onko se liian työlästä? Periaatteessa sen ei pitäisi olla työlästä, esimerkiksi Iivarin [2006] mukaan käyttäjien osallistamista suunniteltaessa vastataan vain kahteen kysymykseen: miksi ja kuinka? Myös Hess ja muut [2013] sekä Bano ja Zowghi [2014] päätyivät samankaltaisiin kysymyksiin. Hessin ja muiden mukaan pitää miettiä, ketkä käyttäjät, milloin sekä miksi. Banon ja Zowghin mukaan pitää tunnistaa käyttäjät, osallistamisen näkökulma, osallistamisen taso, suunnittelun vaiheet, joissa käyttäjät otetaan mukaan sekä suunniteltavan järjestelmän tyyppi.

Tämä tutkielma on kirjallisuuskatsaus, jonka tarkoituksena on antaa lukijalle yleiskatsaus käyttäjien osallistamisen hyödyistä ja sen ongelmista. Lisäksi esittelen erilaisia tapoja osallistaa käyttäjiä. Tutkielmassani osallistamisen hyödyllisyyttä tarkastellaan ohjelmistokehityksen näkökannalta, eli tutkielmassani ei käsitellä niin sanottujen fyysisten tuotteiden suunnittelua. Luvussa 2 esittelen menetelmät, joilla lähdemateriaalit ja tutkielman näkökulmat on valittu. Luvussa 3 kerron, mitä käyttäjien osallistaminen tarkoittaa, mitä hyötyä siitä on, min-

kälaisia haasteita se asettaa kehitykselle sekä millä eri tavoin käyttäjiä pystyy osallistamaan. Lopuksi, luvussa 4, esitän yhteenvedon ja keskustelua aiheesta.

2. Menetelmä

Aineistoa tutkielmaan haettiin neljästä eri tieteellisestä tietokannasta, jotka olivat ACM, ScienceDirect, IEEE Xplore ja Springer Link. Näistä tietokannoista aineistoa haettiin avainsanojen ja erilaisten rajausten avulla, kuten esimerkiksi rajaamalla julkaisuvuotta. Avainsanojen rajaaminen oli haastavaa, sillä käyttäjien osallistamiselle löytyy runsaasti eri käännöksiä, jotka periaatteessa tarkoittavat samaa asiaa, mutta merkitys ei kuitenkaan ole sama. Taulukossa 1 on esitetty yleisimpiä käyttäjien osallistamisesta käytettyjä termejä ja niiden merkitykset.

Termi	Merkitys
User participation	Osallistuminen fyysistä: käyttäjä suorittaa tehtäviä kehitysprosessin aikana ja osallistuu näin suunnitteluun [Bano and Zowghi, 2013].
User involving / involving users	Osallistuminen neuvovaa: käyttäjä kertoo ja kommentoi järjestelmän tärkeyttä ja vastaavuutta hänen tarpeisiinsa [Bano and Zowghi, 2013].
End-user development	Kehitysmalli, jossa loppukäyttäjille annetaan työkalut, jotta he voivat itse suunnitella ja toteuttaa haluamansa ohjelmiston.
Participatory design	Vanhempi käsite, jota ei juurikaan käytetä enää. Suunnittelumetodi, joka kuvaa toimintaa, jossa yrityksen työntekijät osallistuvat omien järjestelmiensä suunnitteluun.
Co-design	Synonyymi Participatory designille.

Taulukko 1: Osallistamisesta käytetyt termit ja niiden merkitykset

Taulukossa 2 on esitetty osallistamisesta käytettyjen termien jaottelu osallistumistavan ja aikajänteen mukaisesti. Taulukossa termit ovat jaoteltu sen mukaan, lähestyykö käyttäjä asiantuntijan vai asiantuntija käyttäjän roolia ja näkökantaa. Tällä tarkoitetaan sitä, täytyykö käyttäjän omaksua suunnittelijan rooli ja ymmärtää, miten suunnittelu tapahtuu vai riittääkö käyttäjän oma tietämys aiheesta.

Aikajänne	Suunnittelija lähestyy käyttäjää	Käyttäjä lähestyy suunnittelijaa
Nykyhetki ↕	User participation	Participatory design
		End-user development
Tulevaisuus	User involving / involving users	Co-design

Taulukko 2: Termien jaottelu osallistamistavan mukaisesti [Tiainen *et al.*, 2013]

Taulukossa 2 aikajänteellä tarkoitetaan sitä, keskittyykö osallistaminen siihen, mitä sovellus on tällä hetkellä vai siihen, mitä se voisi olla tulevaisuudessa. Nykyhetkeen keskittyvissä osallistamistavoissa käyttäjä arvioi sovellusta sen nykytilassa käyttämällä nykyistä, jo olemassa olevaa ohjelmistoa. Tulevaisuuteen keskittyvissä osallistamistavoissa käyttäjä osallistuu suunnittelemaan tulevaisuudessa tehtävää sovellusta, esimerkiksi kertoo, miten hän käyttää sovellusta ja mitä hän siltä haluaa.

Vaikka termeillä ”user participation” ja ”user involving” on merkitysero, niitä käytetään silti toistensa synonyymeinä. Valitsin tutkielmaani aineistoja, joissa käytettiin termejä ”user participation” tai ”user involving”, koska nämä aineistot kertovat yleisesti käyttäjien osallistamisesta eivätkä keskity esimerkiksi jonkin tietyn alustan tai metodin toimivuuteen osallistamisessa.

Toinen tiedonhaussani käyttämä tärkeä termi oli ohjelmistokehitys, josta käytin termejä ”software design”, ”software development”, ”software engineering” sekä joissain tapauksissa ”product design”. Taulukossa 3 on esitetty tietokantakohtaisesti käytetyt hakusanat sekä asetetut rajaukset.

Taulukon 3 mukaisilla hauilla saaduista tuloksista valitsin tutkielman aineistoksi materiaalit, joiden otsikoissa ja tiivistelmissä kerrottiin käyttäjistä, käyttäjäkeskeisyydestä tai käyttäjien osallistamisesta. Lisäksi materiaalin otsikoiden ja tiivistelmien tuli kertoa ohjelmistokehityksestä tai tuotekehityksestä. Aineistosta karsiutuivat pois myös materiaalit, jotka koskivat erikoisryhmille, kuten esimerkiksi sokeille, suunniteltavia sovelluksia sekä ylipäänsä terveydenhoitoon liittyviä sovelluksia. Nämä jätin pois, sillä niillä on omat erityiset vaatimuksensa ja ne suunnitellaan asiantuntijakäyttöön. Tämän rajauksen tarkoituksena oli varmistaa, että aineistossa olevat materiaalit koskevat niin sanottuja peruskäyttäjiä, joilla ei ole paljoa kokemusta ohjelmistokehityksestä. Rajaamalla pois asiantuntijakäyttöön suunniteltavat sovellukset jäivät jäljelle materiaalit, jotka käsittelivät osallistamista yleisesti ja tällöin tutkielmastani tulisi yleistietoa antava kirjallisuuskatsaus.

Tietokanta	Osallistamisen termi	Ohjelmistokehityksen termi	Rajaukset	Hakutuloksia
Science Direct	User participation TAI user involvement TAI participatory design	Software development TAI software design TAI software engineering	Julkaisuvuosi 2000–2014. Tieteenala tietojenkäsittely.	18
ACM	User participation TAI user involvement TAI participatory design TAI end-user development TAI co-design	Software development TAI software design TAI software engineering TAI product design	Julkaisuvuosi 2000–2014. Julkaisutyyppi lehti tai konferenssijulkaisu.	8
IEEE Xplore	User involvement		Julkaisuvuosi 2000–2014.	84
Springer Link	User participation TAI user involvement TAI participatory design TAI end-user development	Software development TAI software design TAI software engineering	Julkaisuvuosi 2000–2014. Tieteenala tietojenkäsittely.	20

Taulukko 3: Käytetyt hakusanat ja asetetut rajaukset

Edellä kuvatun valikoinnin jälkeen aineistoon jäi 17 materiaalia. Osa näistä materiaaleista karsiutui pois analysointivaiheessa, kun huomasin, että ne eivät olleet relevantteja aiheeni kannalta. Karsiutuneissa materiaaleissa keskityttiin joko jonkin tietyn ohjelmiston tai toimintavan käyttöön osallistamisessa tai materiaali ei tarjonnut mitään aiheen kannalta hyödyllistä tietoa. Analysoinnin jälkeen aineistossa oli yhdeksän materiaalia, jotka luokittelin sen mukaan, kertoivatko ne osallistamisen keinoista, hyödyistä, haasteista vai osallistamisesta yleisesti. Kyseinen luokittelu toimi tämän tutkielman runkona.

3. Käyttäjien osallistaminen

Seuraavaksi esittelen kirjallisuuskatsauksen tulokset. Aloitan esittelemällä käyttäjien osallistamisen tavat kohdassa 3.1. Katsauksessa nousi vahvasti esille käyttäjien osallistamisen positiivinen vaikutus järjestelmän onnistumiseen sekä vaatimusten laatuun. Osallistamisen hyödyistä kerron tarkemmin kohdassa 3.2. Käyttäjien osallistamisesta nousi myös esille erilaisia haasteita, jotka tulee ottaa

huomioon osallistamista suunnitellessa. Erilaisista haasteista kerron kohdassa 3.3.

3.1. Osallistamisen tavat

Käyttäjät voivat osallistua suunnitteluprosessiin heti vaatimuksia määriteltäessä tai tulla mukaan vasta myöhemmin, esimerkiksi rautalankamalleja mietittäessä tai kun tarvitaan arvioivampaa lähestymistapaa [Hess *et al.,* 2013]. Mitään syytä ei ole myöskään esitetty sille, miksei osallistaminen voisi olla kokoaikaista. Sen toteuttaminen käytännössä saattaa kuitenkin olla vaikeaa.

Iivarin [2006] sekä Heiskarin ja Lehtolan [2009] mukaan osallistamista on kolmea eri tyyppiä: informatiivista, konsultoivaa tai osallistavaa. Nämä tyypit eroavat toisistaan käyttäjien roolin ja tehtävien perusteella. Informatiivisessa ja konsultoivassa tyypissä käyttäjät tarjoavat tietoa ja osallistuvat testaukseen tarkkailun kohteina sekä kommentoivat ennalta määritettyjä suunnitteluratkaisuja. Tällöin he toimivat ”human factorina” eli passiivisina, tarkkailtavina kohteina. Osallistavassa tyypissä käyttäjät puolestaan toimivat aktiivisina osallisina suunnittelussa ja heillä on valtaa osallistua myös päätöksentekoon. Tällöin he toimivat ”human actoreina” eli aktiivisina agentteina. Informatiivisessa ja konsultoivassa osallistamisessa käyttäjät eivät välttämättä osallistu kehitykseen itse, vaan määrätty välitoimija (esim. käytettävyysasiantuntija) voi haastatella ja olla yhteydessä käyttäjiin ja saatujen tietojen perusteella edustaa (represent) heitä ohjelmistokehityksessä [Iivari, 2006].

Samankaltaiseen jaotteluun päätyivät myös Kujala ja muut [2005] tutkiessaan käyttäjien roolia osallistavassa suunnittelussa. Tutkimuksensa tuloksissa he erittelivät neljä erilaista metodia: osallistuminen, vierailu asiakkaiden luona ja työpajat, tarkastelu sekä käytettävyystutkimus.

Heiskarin ja Lehtolan [2009] mukaan myös osallistamisen taso voi vaihdella. He esittävätkin seitsemän osallistamisen tasoa, jotka ovat: osallistamattomuus, symbolinen osallistaminen, neuvova osallistaminen, osallistaminen vähällä vaikutusvallalla, osallistaminen tekemällä sekä osallistaminen vahvalla vaikutusvallalla. Osallistamattomuudessa käyttäjät eivät ole halukkaita osallistumaan tai heitä ei kutsuta mukaan. Symbolisessa osallistamisessa käyttäjät kutsutaan mukaan, mutta heidän palautteensa jätetään huomiotta. Neuvovassa osallistamisessa käyttäjien neuvot tai vinkit kerätään kyselyillä. Kolmessa viimeisessä tasossa käyttäjät osallistuvat suunnitteluun. Osallistamisessa vähällä vaikutusvallalla käyttäjillä on ”viimeinen sana” kehityksen eri vaiheissa, eli heidän vastuulla on hyväksyä vaiheen tuotos ennen seuraavaan vaiheeseen siirtymistä. Tekemällä osallistaessa käyttäjä toimii kehitystiimin jäsenenä ja osallistuu itse kehitykseen. Viimeisellä tasolla eli vahvalla vaikutusvallalla osal-

listaessa käyttäjällä on paljon valtaa ja hän pystyy esimerkiksi rahoittamaan kehitystä ja täten vaikuttamaan kehityksen kulkuun.

Tulevaisuudessa käyttäjien osallistaminen voi tulla helpommaksi jatkuvasti kehittyvien sosiaalisten teknologioiden, kuten esimerkiksi blogien ja wikien, avulla, jotka mahdollistavat osallistumisen suunnitteluun etänä ja myös reaaliaikaisesti [Hess *et al.,* 2013]. Jotkin organisaatiot tutkivat mahdollisuuksia luoda alustoja, esimerkiksi verkkosivuja, joiden kautta käyttäjät voiva luoda ja muokata sisältöä, luoden yhteistyötä käyttäjien ja organisaatioiden välillä [Pankowska, 2012].

3.2. Osallistamisen hyödyt

Käyttäjien osallistamisella on yleisesti positiivinen vaikutus järjestelmän onnistumiseen (system success) ja käyttäjien tyytyväisyyteen [Kujala *et al.,* 2005; Abelein and Paech, 2013]. Abelein ja muut [2013] löysivät kuusi järjestelmän onnistumisen osa-aluetta, joista yleisimmin käytetyt olivat käyttäjän tyytyväisyys, järjestelmän käyttö ja järjestelmän laatu. Osa-alueet selityksineen on esitetty kuvassa 1.

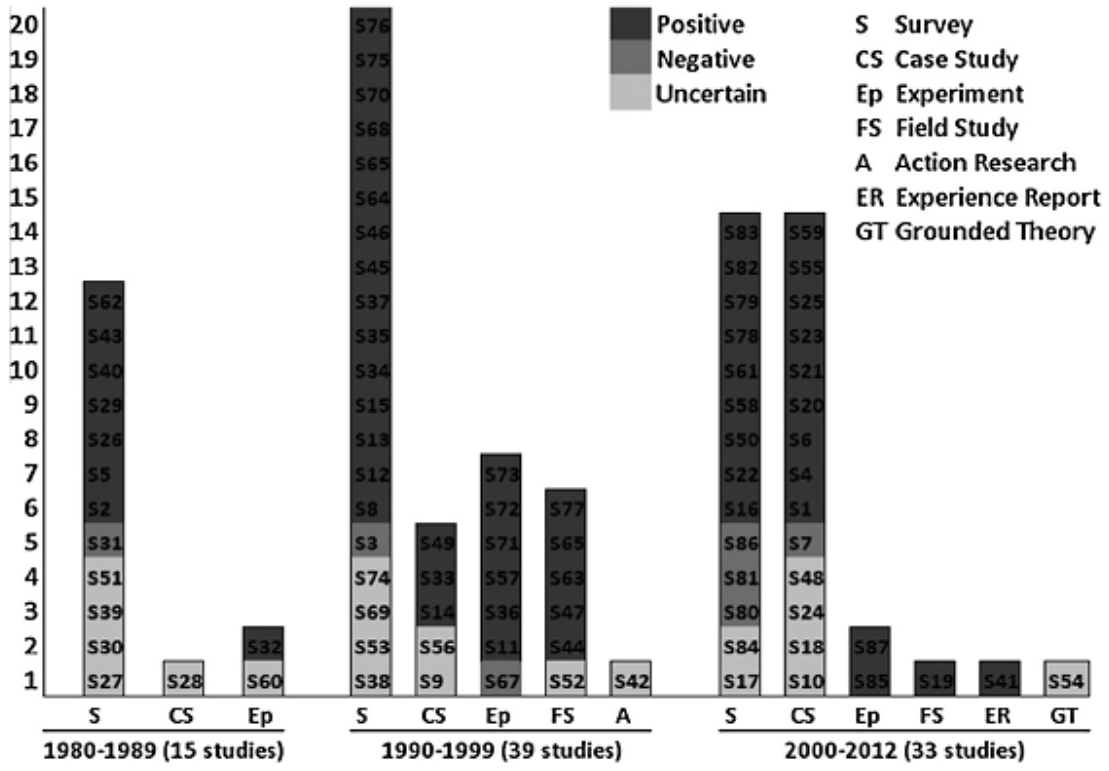
Myös Bano ja Zowghi [2014] toteavat käyttäjien osallistamisella olevan positiivinen vaikutus järjestelmän onnistumiseen. Kirjallisuuskartoituksessaan he toteavat 68 prosenttia (59/87 tutkimusta) kartoituksessa olleista tutkimuksista päättyneen positiivisesta tulokseen osallistamisen vaikutuksesta. Vain seitsemässä tutkimuksessa päädyttiin negatiiviseen lopputulokseen. Kaikista tutkimuksista 21:ssa ei saatu selvää negatiivista eivätkä positiivista lopputulosta. Tulokset on esitetty kuvassa 2. Kuvassa 2 pystypalkit esittävät aikakaudella tehtyjen tietyntyyppisiä tutkimusten määriä, esimerkiksi aikakaudella 1990–1999 tehtiin 39 tutkimusta, joista kaksikymmentä oli kyselytutkimuksia (survey). Näistä kahdestakymmenestä tutkimuksesta 15 päätyi positiiviseen tulokseen (tummanharmaa osa palkkia), yksi negatiiviseen (harmaa osa) ja neljän tulokset olivat kyseenalaisia (vaaleanharmaa osa).

TABLE 2

Overview of aspects of system success.

Measurement of system success	Definition for study	Examples of original measurement	Number of studies
User satisfaction	User's degree of favorableness with respect to the system and the mechanics of interaction	End user computing satisfaction, end user satisfaction, information satisfaction, outcome satisfaction, perceived system usefulness, perceived usefulness, system acceptance, system satisfaction, usefulness, user assessment, user information satisfaction, user satisfaction	50
System use	Frequency of use of the developed system	Intention to use, system impact, system usage, time spent using	18
System quality	Structured set of characteristics such as a system's functional suitability, reliability, usability, performance efficiency, compatibility, security, maintainability, and portability	Accessibility, accuracy, completeness, flexibility, perceived system quality, product success	17
Project in time and budget	Project efficiency and effectiveness in terms of schedule, budget, and work quality	Project success, overall success, process satisfaction, project completion, project performance, project success, successful implementation	8
Ease of use	Degree to which a user expects the target system to be free of effort; also refers to system friendliness and handling in system's use	Perceived impact on work, system friendliness	4
Data quality	Degree to which the characteristics of data satisfy stated and implied needs when used under specified conditions; accuracy, consistency, and availability of data	Appropriateness of format, availability of historical data, data accuracy, data consistency, data sufficiency	1

Kuva 1: Järjestelmän onnistumisen osa-alueet [Abelein et al., 2013]

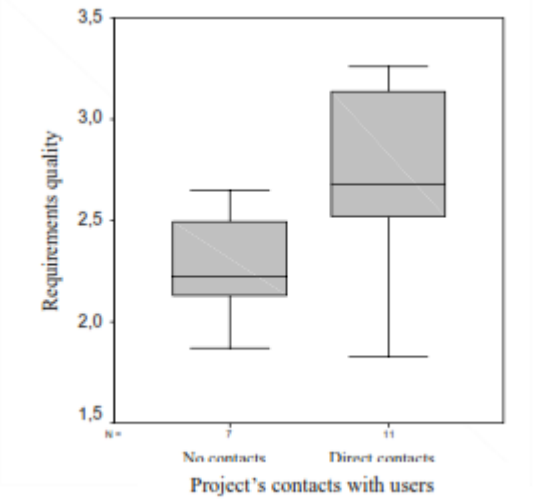


Kuva 2: Osallistamisen vaikutus järjestelmän menestykseen [Bano and Zowghi, 2014]

Järjestelmän onnistumisen lisäksi löytyy todisteita siitä, että vaatimusten muodostaminen on tehokkaampaa käyttäjien ollessa pääasiallinen tiedonlähde. Käyttäjien osallistamisen sanotaan parantavan vaatimusten laatua, lisäävän ymmärrystä käyttäjän ja tehtävien asettamista vaatimuksista sekä auttavan kehittämään hyödyllisiä ja käytettäviä järjestelmiä [Kujala *et al.*, 2005]. Vaatimusten laadun paranemisella tarkoitetaan sitä, että vaatimukset ovat tarkemmin määritelty, paremmin dokumentoitu sekä kuvaavat tarkemmin kohderyhmän ohjelmistolle asettamia tarpeita. Lisäksi käyttäjät, jotka tuntevat olevansa mukana järjestelmän suunnittelussa, suhtautuvat siihen positiivisemmin, arvioivat sen hyödyllisemmäksi ja ovat siihen tyytyväisempiä [Abelein and Paech, 2013]. Tällöin käyttäjät hyväksyvät järjestelmän paremmin sekä ymmärtävät sitä paremmin, jolloin he osallistuvat aktiivisemmin päätösten tekoon ja yrityskulttuurista tulee demokraattisempi.

Samankaltaisiin huomioihin päätyivät myös Pankowska [2012] sekä Heiskari ja Lehtola [2009]. Heidän mukaansa käyttäjien osallistamisella on edellä mainittujen lisäksi seuraavia hyötyjä: se vähentää tutkimuksen ja kehityksen kuluja, antaa mahdollisuuksia erilaistaa markkinointitarjouksia ja lopullisia tuotteita, vähentää markkinointiin tarvittavaa aikaa, tarjoaa mahdollisuuksia käyttäjien koulutuksen tukemiseen, lisää tiedon jakoa tuotteesta käyttäjien joukossa, antaa mahdollisuuden kehittää pitkäkestoisia asiakassuhteita, lisää käyttäjien uskollisuutta sekä parantaa yrityksen imagoa.

Kujalan ja muiden [2005] mukaan projekteissa, joissa vaatimukset perustuivat käyttäjiltä kerättyyn tietoon, järjestelmän onnistuminen arvioitiin korkeammaksi ja vaatimusten laatimisesta muodostuvat kulut olivat alhaisemmat. Yleisesti tällaiset projektit vaativat siis vähemmän resursseja. Käyttäjien asettamat vaatimukset ovat tärkeitä, sillä ne kuvaavat, miten tehtävä tuote tulee auttamaan käyttäjiä tavoitteiden saavuttamisessa ja kuinka tuote vastaa käyttökontekstin asettamia tarpeita. Tämän takia vaatimusten tarkkuus on tärkeää ja ne kannattaa kerätä käyttäjiltä. Kujalan ja muiden [2005] tekemän kyselytutkimuksen mukaan vaatimusten laadun keskiarvo oli huomattavasti korkeampi projekteissa, joissa käyttäjät olivat mukana (kuva 3).



Kuva 3: Käyttäjien osallistamisen vaikutus vaatimusten laatuun [Kujala et al., 2005]

Kujalan ja muiden [2005] mukaan suoralla kontaktilla käyttäjiin on vaatimusten laadun korkeamman keskiarvon lisäksi kokonaisuudessaan positiivinen vaikutus suureen osaan vaatimusten laatua koskeviin väittämiin, jotka ovat esitetty kuvassa 4.

Table 5. Respondents' mean assessment of the requirements quality issues in their development projects.

Requirements quality statement	Mean response	
	No direct contacts with users	Direct contacts with users
Requirements describe a system that meets the user needs	2.6	**2.8**
The benefits of user roles are described	2.3	**2.5**
Understandable requirements	2.0	**2.6**
A white box view of the system	2.6	**2.8**
Essential customer issues are documented	2.1	**3.0**
The correctness of requirements checked with users	1.4	**2.7**
The differences among user groups are described	1.1	**2.5**
The differences among customers are described	1.7	**2.8**
Usability requirements are documented	2.0	**3.1**
Requirements are completely defined	**2.7**	2.6
There are moderately few errors in the requirements	2.1	**2.6**
Essential user issues are documented	2.1	**3.0**
Sum of the 23 statements	2.3	**2.7**

A 4-point scale was used, ranging from 1=strongly disagree to 4=strongly agree, + no opinion option

Kuva 4: Kontaktin vaikutus vaatimusten laatuun [Kujala et al., 2005]

Kuten kuvassa 4 näkyy, kontaktilla on positiivinen vaikutus vaatimusten laatua koskeviin väittämiin. Suurimmat muutokset löytyvät niiden väittämien

kohdalta, jotka koskevat erilaisten käyttäjien ja käyttäjäryhmien huomioon ottamista sekä vaatimusten dokumentointia. Voidaan siis todeta, että käyttäjien ollessa mukana erilaiset käyttäjät huomioidaan paremmin, vaatimukset ovat täsmällisempiä ja vaatimukset dokumentoidaan varmemmin.

Järjestelmän onnistumisen ja vaatimusten laadun lisäksi myös projektin onnistuminen arviointiin korkeammaksi projekteissa, joissa käyttäjät olivat mukana. Kuva 5 esittää väittämiä koskien projektin onnistumista sekä osallistujien arviot niistä. Projektin onnistumisen kohdalla keskiarvioissa ei ole suuria eroja, toisin kuin vaatimusten laadun kohdalla.

Table 6. Respondents' mean assessment of the success issues in their projects.

Project success statement	Mean response	
	No direct contacts with users	Direct contacts with users
The project was a success	2.4	**2.8**
The organization considers the project a success	2.4	**2.7**
The result of the project meets the business goals	2.4	**3.0**
The result of the project meets requirements	2.9	**3.1**
The result of the project is a success according to customer and user feedback	2.3	**2.5**
The project was completed on schedule	1.8	**2.2**
The success of the project was above average	1.7	**2.4**
The requirements engineering was performed successfully in the project	1.7	**2.6**
There were several differences among the opinions of the parties in the project	2.3	**2.3**
Sum	2.2	**2.6**

A 4-point scale was used, ranging from 1=strongly disagree to 4=strongly agree, + no opinion option

Kuva 5: Kontaktin vaikutus projektin onnistumiseen [Kujala et al., 2005]

Kuten kuvasta 5 näkyy, suurimmat erot löytyvät kohdista, jotka koskevat vaatimusten onnistumista, yritystoiminnan tavoitteiden saavuttamista ja projektin onnistumista verrattuna keskitasoon. Voidaan siis todeta, että käyttäjien osallistamisella on monelta kannalta huomattava merkitys vaatimusten laatuun ja tarkkuuteen.

3.3. Osallistamisen haasteet

Vaikka käyttäjien osallistamisen tärkeys onkin yleisesti hyväksytty, on käyttäjien osallistaminen osoittautunut haastavaksi erityisesti tuotekehityksen kontekstissa [Iivari, 2006]. Haastetta tuo esimerkiksi kehitysprosessin lyhyys, jolloin aikaa ei vain ole käyttäjien osallistamiseen. Lisäksi haastetta tuo puutteellinen käsitys käyttäjistä ja käyttökonteksteista. Usein kehitystä jatketaankin ilman käyttäjien palautetta tai vaihtoehtoisesti käyttäjät otetaan mukaan liian

myöhäisessä vaiheessa, jolloin tuloksilla ei ole suurta vaikutusta suunnitteluun [Iivari, 2006]. Ajankäyttö nousi haasteeksi myös tilanteessa, jossa käyttäjiltä tuli runsaasti toiveita ja toimintoehdotuksia harkittaviksi ja analysoitaviksi [Kujala *et al.*, 2005].

Toinen haasteellinen osa-alue on käyttäjien rooli. Kujalan ja muiden [2005] mukaan käyttäjän osallistuessa täysivaltaisesti vaatimusten määrittelyyn, on hän vaarassa menettää alkuperäisen näkemyksensä ja siirtyä teknisempään näkökantaan. Tällöin käyttäjät käsittääkseni miettivät enemmän järjestelmän rakennetta ja teknistä toteutusta, kuin sitä, mitä he haluavat ja tarvitsevat järjestelmältä.

Myös Hess ja muut [2013] raportoivat käyttäjien roolin ongelmallisuudesta. Heidän tutkimuksessaan seurattiin erään televisioyhtiön palvelun kehittämistä. Kehityksessä esiintyi ongelmia tilanteissa, joissa käyttäjät saivat tehdä ehdotuksia ja ideoida uusia toiminnallisuuksia palveluun. Välillä näissä tilanteissa käyttäjillä oli liian korkeat odotukset sen suhteen, mitä pystytään toteuttamaan, ja he hermostuivat, koska heidän ideoitaan ei otettu toteutettaviksi. Ongelmia ilmeni myös käyttäjien innostuksessa osallistua. Osallistujat olivat projektin aluksi innokkaampia kommentoimaan ja tekemään ehdotuksia, mutta innostus hiipui projektin edetessä.

Hessin ja muiden [2013] tutkimuksessa ongelmia olisi voinut vähentää osallistamisen tekeminen niin, että käyttäjät olisivat olleet vuorovaikutuksessa suunnitteluryhmän kanssa. Toteutetulla tavalla osasta osallistujia tuntui, ettei heillä ole yhteyttä päätöstentekoon. Tutkitussa kehitysprojektissa osallistaminen tapahtui pääasiallisesti keskustelupalstan ja wikin kautta, joihin käyttäjät kirjoittivat ideoitaan ja keskustelivat niistä. Keskustelupalstan käytössä oli myös muita ongelmia, kuten se, että käyttäjät eivät aina verranneet kirjoitustaan aiempiin kirjoituksiin, jolloin keskustelupalstalla esiintyivät samat ideat useaan kertaan. Jos osallistamista halutaan toteuttaa keskustelupalstan avulla, kannattaa rajata keskusteltavia asioita ja jakaa keskustelunaiheet niin, että käyttäjät pääsevät keskustelemaan aiheista joista heillä on tietämystä. Lisäksi keskustelijoille kannattaa selittää myös päätöksentekoprosessi eli millä perusteella toteutettavat ideat valitaan. Tämä voi vähentää päätösten aiheuttamaa suuttumusta hylättyjä ideoita koskien.

Myös Heiskari ja Lehtola [2009] raportoivat samanlaisia käyttäjän rooliin liittyviä ongelmia. He esittivät ongelman roolien muodossa, jotka ovat ”hostage” ja ”propagandist”. Ensimmäinen näistä kuvaa roolia, jossa käyttäjä on osa kehitystiimiä, mutta hän ei saa osallistua itse kehitykseen. Jälkimmäinen rooli puolestaan kuvaa tilannetta, jossa käyttäjä osallistuu kehitykseen opettelemalla kehityksen metodeja ja täten unohtaa roolinsa ja näkökantansa käyttäjänä.

Vaikka käyttäjän roolin ongelmallisuus nousikin usein esille, Banon ja Zowghin [2014] mukaan merkittävimmät ongelmat aiheutuivat kommunikaatio-ongelmista ja väärinkäsityksistä käyttäjien ja kehitystiimin välillä. Heidän mielestään osallistamisen suurin haaste koskee käyttäjien motivointia, koska motivaation menetyksellä on suora vaikutus asenteeseen ja käytökseen.

4. Keskustelu ja johtopäätökset

Käyttäjien edustaminen on hyvin yleistä ohjelmistokehityksessä. Siinä on omat ongelmansa ja käytettävien välikäsien roolit tuleekin miettiä tarkkaan [Iivari, 2006]. Edustamisen yksi suurimmista ongelma on näkökanta, koska asiantuntija ei aina osaa ”laskeutua” tavallisen käyttäjän tasolle ja arvioida asioita tältä näkökannalta. Samaan tapaan kuin käyttäjien rooli voi vaihdella, myös välikäden rooli voi vaihdella. Välikäden roolin tulisi olla osallistuva, mutta se voi olla myös informatiivinen tai konsultoiva [Iivari, 2006]. Mielestäni parhaimman tuloksen saisi aikaan silloin, kun käyttäjät ja asiantuntijat toimivat yhdessä ja miettivät suunnitteluun liittyviä asioita yhteistyössä.

Banon ja Zowghin [2014] mukaan käyttäjien osallistaminen aiheuttaa vakavia ongelmia hyötyjen sijasta, mikäli sitä käytetään väärin. Käyttäjien osallistamisessa onkin runsaasti haasteita, joista suurimmat ovat ajankäyttö ja käyttäjien roolin muodostaminen. Ajankäytön luoman haasteen voisi pystyä korjaamaan laatimalla kehitysprosessin, jossa on varattu riittävästi aikaa vaatimusten määrittelylle ja laadunvarmistukselle. Käyttäjien osallistamiseen kannattaa mielestäni panostaa ja käyttää aikaa, sillä on olemassa todisteita siitä, että osallistamista käyttävät projektit ovat onnistuneempia ja käyttävät vähemmän resursseja.

Käyttäjiä osallistaessa tulee käyttäjien rooli ja osallistumisen aste miettiä tarkkaan [Kujala *et al.,* 2005]. Jos käyttäjälle antaa liian ”vaikean” roolin, hänen näkökantansa muuttuu ja hänen tarjoamansa tieto ei välttämättä ole enää puhtaasti käyttäjän näkökulmasta. Tällöin järjestelmästä ei välttämättä tule käyttäjän tarpeita vastaavaa. Yksi lupaava ratkaisu tähän on kenttätutkimus, jossa käyttäjiin ja heidän toimintaansa tutustutaan heidän omassa ympäristössään. Tällöin käyttäjän rooliksi jää toimia tiedonantajana ja havainnoitavana kohteena eikä toimia roolissa, joka ei hänelle kuulu.

Yksi tärkeä kysymys osallistamista suunniteltaessa on mielestäni kysymys siitä, mikä on haluttu lopputulos? Esimerkiksi Hessin ja muiden [2013] tutkimuksessa lopputuloksena oleva sovellus jakoi mielipiteitä siten, että kehityksessä mukana olleet käyttäjät eivät pitäneet siitä, mutta uudet käyttäjät, jotka käyttivät sovellusta ensimmäisen kerran, olivat erittäin tyytyväisiä siihen. Oliko tällöin osallistamisesta hyötyä ja onnistuiko projekti? Mielestäni vastaus on kyllä. Osa projektiin osallistuneista käyttäjistä ei pitänyt valmiista sovelluksesta,

koska siinä ei ollut toteutettu heidän ideoitaan. Mielestäni projekti onnistui hyvin, sillä uusi sovellus houkutteli uusia käyttäjiä ja oli heidän mielestään hyvä.

Yhteenvetona Kujala ja muut [2005] esittävät, että jos perustat vaatimukset käyttäjiltä kerättyyn aitoon tietoon, säästät aikaa ja vaivaa myöhemmin. Tämä pitää mielestäni paikkansa hyvin, koska jos vaatimuksia ei mietitä kunnolla projektin alussa, voi lopussa tulla ongelmia mikäli järjestelmä ei vastaakaan käyttäjien tarpeisiin. Pahimmassa tapauksessa täytyy aloittaa järjestelmän kehitys alusta. Tässä vaiheessa muutosten tekeminen on vaikeaa ja kallista. Heiskari ja Lehtola [2009] toteavatkin käyttäjien tai käyttäjäryhmien palvelemisen ja ymmärtämisen tuovan kilpailuetua, jota ei kannata jättää huomiotta.

Tutkielman tulokset olisivat voineet olla kattavammat, jos olisin ottanut mukaan myös aiemmin tehtyjä tutkimuksia. Osallistava suunnittelu oli muodissa 1970- ja 80-luvuilla ja on jälleen nousemassa takaisin suosioon. Nyt kuitenkin vaikuttaa olevan hiljaisempi jakso, mikä näkyy myös valitsemassani aineistossa. Toinen aineistoon vaikuttanut päätös oli avainsanojen määrä. Aineistoista olisi voinut tulla kattavampi ja laajempi, jos olisin valinnut enemmän avainsanoja.

Tutkielman aineistosta nousi esille kysymyksiä, joita olisi mielenkiintoista tutkia. Yksi tällainen kysymys oli haluttu osallistamisen lopputulos eli minkälaiseen lopputulokseen tulisi päätyä, jotta osallistaminen olisi onnistunut. Aineistoni tutkimuksissa tähän ei ollut perehdytty ja olisi mielenkiintoista tutkia, miten haluttu lopputuloksen määrittäminen vaikuttaa projektin etenemiseen ja lopputuloksen olevaan sovellukseen.

Viiteluettelo

[Abelein *et al.*, 2013] U. Abelein, H. Sharp and B. Paech, Does involving users in software development really influence system success?, *IEEE Software*, **30**, 6 (Nov.-Dec. 2013), 17–23.

[Abelein and Paech, 2013] U. Abelein and B. Paech, Understanding the influence of user participation and involvement on system success – a systematic mapping study, *Empirical Software Engineering*, **20**, 1 (December 2013), 1-54.

[Bano and Zowghi, 2014] M. Bano and D. Zowghi, A systematic review on the relationship between user involvement and system success, To appear in *Information and Software Technology*.

[Heiskari and Lehtola, 2009] J. Heiskari and L. Lehtola, Investigating the State of User Involvement in Practice, In: *Proc. of the Asia-Pasific Software Engineering Conference (APSEC '09)*, 2009, 433–440.

[Hess *et al.*, 2013] J. Hess, D. Randall, V. Pipek and V. Wulf, Involving users in the wild—Participatory product development in and with online communities, *International Journal of Human-Computer Studies*, **71**, 5 (May 2013), 570–589.

[Iivari, 2006] N. Iivari, 'Representing the User' in software development—a cultural analysis of usability work in the product development context, *Interacting with Computers*, **18**, 4 (July 2006), 635–664.

[Kujala *et al.*, 2005] S. Kujala, M. Kauppinen, L. Lehtola and T. Kojo, The role of user involvement in requirements quality and project success, In: *Proc. of the 13th IEEE International Conference on Requirement Engineering*, 2005, 75–84.

[Pankowska, 2012] M. Pankowska, User participation in information system development, In: *Proc. of the International Conference on Information Society (i-Society 2012)*, 396–401.

[Tiainen *et al.*, 2013] T. Tiainen, A. Ellman, and T. Kaapu, Workers' tacit knowledge transferred to conceptual design: the case of mobile work machine. In: *Proc. of the Boundary-Crossing Conference on Co-Design in Innovation*, 709-720. *Science + Technology*, 2013.

Tiedon aistillistamisesta: aistien hyödyntämisestä tiedon esittämiseen

Esa Lempiäinen

Tiivistelmä.
Tiedon visualisoinnin historia ulottuu kauas ennen tietokoneiden mahdollistamia vuorovaikutteisia visualisointeja. Multimodaalisuus puolestaan on yksi nopeimmin kehittyvistä ihminen-teknologia -vuorovaikutuksen osa-alueista. Tämä tutkielma tarkastelee näiden kahden alan leikkauspistettä, eli eri aistien hyödyntämistä tiedon esittämiseen.

Avainsanat ja -sanonnat: moniaistillisuus, multimodaalisuus, visualisointi, sonifikointi, haptiikka, näköaisti, kuuloaisti, tuntoaisti, hajuaisti, makuaisti

1. Johdanto

Informaation visualisoinnilla tarkoitetaan visuaalisen esityksen muodostamista abstraktista datasta. Tällöin tarkoitus on, että datan numeeriset piirteet muodostavat visuaalisia piirteitä, jotka visualisointia tarkasteleva ihminen kykenee hahmottamaan luontaisesti tehokkaan visuaalisen havaintojärjestelmänsä avulla.

Juuri visuaalisen havaintojärjestelmän tehokkuus, eli sen käsittelemän informaation määrä, nopeus ja tarkkuus suhteessa muihin aisteihin, on syy sen suosioon informaation aistillistamisen välineenä. Vastaavasti muiden aistimodaliteettien, kuten kuulon ja tuntoaistin, käyttö tähän tarkoitukseen on huomattavasti vähäisempää.

Tämä tutkielma tarkastelee kirjallisuuskatsauksen kautta muiden aistien kuin näköaistin käyttömahdollisuuksia ja haasteita datan esittämisessä.

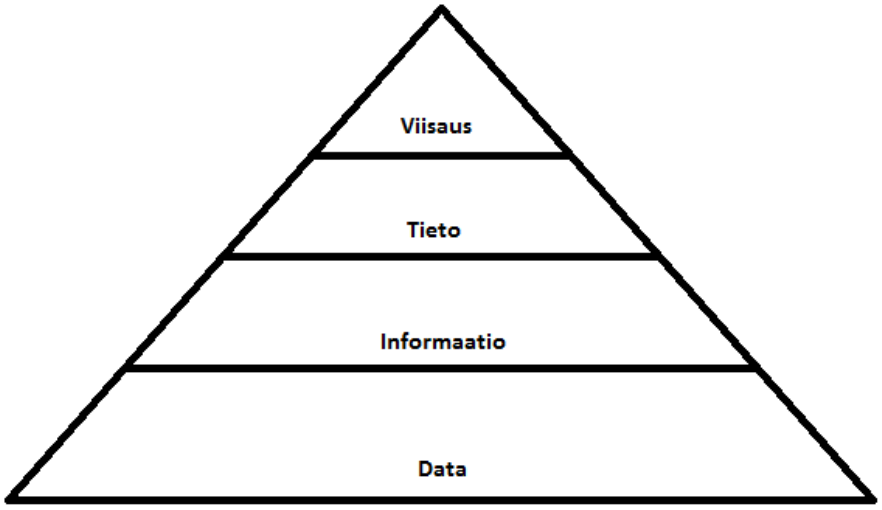
Luku 2 käy läpi tiedon jaottelua sen hienojakoisuuden mukaan. Tarkoituksena on selventää tutkielman keskeisiä tietoa kuvaavia käsiteitä. Luku 3 sisältää yleisen esittelyn keskeisten aistijärjestelmien toiminnasta ja luku 4 tiedon visualisoinnin käsitteellistä perustaa ja sen käytön syitä. Luku 5 esittelee modaliteettien ja multimodaalisuuden peruskäsitteitä ihmisen ja teknologian vuorovaikutuksessa. Luku 6 tarkastelee tieteellisten kirjallisuuden sisältämiä teoreettisia perustuksia kuulo, tunto-, haju- ja makuaistin hyödyntämiselle tiedon esittämiseen. Luku 7 sisältää yhteenvedon ja pohdintaa kirjallisuuskatsauksen tuloksista.

2. Data, informaatio, tieto ja viisaus

Ennen käsitteen *tieto* enempää käyttöä se on syytä määritellä. Tämä arkikielessäkin yleinen sana on merkitykseltään moninaisempi ja ongelmallisempi kuin arkisesti saattaisi ajatella.

Eräs usein toistettu määritelmä tiedolle on Platonilta peräisin oleva ”hyvin perusteltu tosi uskomus”. Menemättä osien ”hyvin perusteltu” ja ”tosi” määrittelemisen ongelmiin tämä määritelmä on aivan riittävä tämän tutkielman tarpeisiin.

Tuo määritelmä koskee kuitenkin vain yhtä niistä asioista, joita suomen kielessä kutsutaan tiedoksi. Selvyyden vuoksi tässä kohdassa on hyödyllistä ottaa esiin englannin sanat *data*, *information*, *knowledge* ja *wisdom* sekä niiden sisältöä käsittelevä malli, niin kutsuttu *DIKW-pyramidi* (kuva 1) [Rowley, 2007].



Kuva 1. DIKW-hierarkia.

DIKW-hierarkiamallista on olemassa eri yksityiskohtia sisältäviä versioita, mutta niille on yhteistä tiedon käsitteiden jaotteleminen siten, että tietosisällöltään vähäisemmät kokonaisuudet muodostavat hierarkiassa ylös siirryttäessä yhä suuremmalla tietosisällöllä ja siten suuremmalla hyödyllisyydellä varustettuja kokonaisuuksia.

Ackoff [1989] määrittelee keskeiset käsitteet seuraavasti:

- *Data* määritellään symboleiksi, jotka esittävät ominaisuuksia olioissa, tapahtumissa ja niiden ympäristöissä. Ne ovat *tarkastelun* tulosta. Ne eivät kuitenkaan ole hyödyllisiä ennen kuin ne ovat käytettävässä (tarkoituksenmukaisessa) muodossa. Ero *datan* ja *informaation* välillä on toiminnallinen, ei rakenteellinen.
- *Informaatio* sisältyy kuvailuun; vastauksiin kysymyksiin, jotka alkavat sanoilla kuten kuka, mitä, missä ja kuinka paljon. Informaatiojär-

jestelmät tuottavat, säilövät, palauttavat ja käsittelevät *dataa*. *Informaatio* on johdettu *datasta*.

- *Tieto* on tietotaitoa, ja se mahdollistaa informaation muuttamisen toimintaohjeiksi. *Tieto* voi olla saatu toiselta, jolla se on tai muodostettu kokemuksesta.
- *Älykkyys* on kykyä lisätä tehokkuutta.
- *Viisaus* on kykyä lisätä tuloksellisuutta. *Viisaus* tuottaa arvoa, mikä vaatii arvostelukykyä. Eettiset ja esteettiset arvot, jotka tähän sisältyvät, ovat toimijan sisäisiä, yksilöllisiä ja henkilökohtaisia.

Tämän tutkielman kannalta on olennaista tehdä ero käsitteiden *data*, *informaatio*, *tieto* ja *viisaus* välille. Suomen kielessä sanaa *tieto* käytetään yleisesti tarkoittamaan DIKW-mallin mukaisia *dataa*, *informaatiota* ja *tietoa*. Aistimisen ja ihmisen ja teknologian välisen kommunikoinnin näkökulmasta keskeisimpiä näistä ovat *data* ja *informaatio*.

Data (latinasta, monikko sanasta *datum*, annettu) on käsitettävissä ihmisen aistien ja koneiden käsittelemiksi fysikaalisiksi ilmiöiksi (fotoneja verkkokalvolla, ääniaaltoja tärykalvolla, painetta iholla, kemikaaleja makureseptoreissa) tai symbolijärjestelmien pienimmiksi yksiköiksi (kirjaimet, numerot). *Informaatio* on *datan* kokonaisuuksia, joista on mahdollista tulkita merkityksiä. *Tieto* on *informaatiosta* tulkittuja merkityksiä ja yhteyksiä, eli se on havainnoijan, ihmisen tai tietoa käsittelevän koneen *informaatiosta* tuottamaa. *Viisaus* on *tiedon* kokonaisuuksia, joilla on arvoa päätöksenteossa.

Tässä tutkielmassa käytetään vastedes sekä sanoja *data, informaatio* ja *tieto* edellä kuvatuissa merkityksissä että sanaa *tieto* viittaamaan niihin kaikkiin. Syy tälle on noudattaa vakiintunutta käytäntöä: esimerkiksi käsitteen *information visualization* yleinen suomennos on *tiedon visualisointi*.

3. Aistit ja havaitseminen

Tutkielman kannalta ei ole tarpeellista käydä läpi aistien toimintaa kovin syvällisesti, sillä vuorovaikutteinen teknologia on sisältämänsä havaintopsykologian osalta lähinnä soveltavaa. Aistien toimintaa on kuitenkin hyödyllistä esitellä tässä sen verran, että myöhemmin esitellyt aistijärjestelmiä hyödyntävät teknologiat saavat viitekehystä. Katsaus perustuu Purvesin ja muiden [2008] esitykseen.

Aistit ovat ihmisen fysiologis-kognitiivisia järjestelmiä, joiden kautta ihminen saa tietoa ympäröivästä maailmasta. Niiden toiminta perustuu reseptorieli vastaanotinsolujen ja hermosolujen toimintaan. Vastaanotinsolut reagoivat fysikaalisiin tai kemiallisiin *ärsykkeisiin* ja tuottavat sähkökemiallisia hermoimpulsseja, jotka kulkeutuvat hermosoluratoja pitkin eri aistijärjestelmien signaa-

lien käsittelyyn erikoistuneille aivokuoren osille. Varsinainen aistiminen, eli vastaanotinsolujen tuottaman tiedon käsittely havaintokokemuksiksi, tapahtuu siis hermostossa, lähinnä aivoissa.

Klassiset viisi aistia, *näkö*, *kuulo*, *tunto*, *maku* ja *haju*, eivät ole ominaisuuksiltaan aivan yhtä yksiselitteisiä kuin millaisiksi ne arkipuheessa mielletään. Esimerkiksi aistin määritteleminen sisältää omat sudenkuoppansa: kullakin aistilla on omat reseptorisolunsa, mutta joillain aisteilla niitä on useita erilaisia (esim. näköaistin tappi- ja sauvasolut ja hajuaistin reseptorit kemikaalien eri piirteille). Aisteja ei siis jaotella minkään yksittäisen solutyypin kautta esim. hämärä- ja kirkasnäön aistiksi. Lisäksi aistiaivokuoren alueet eivät ole jyrkkärajaisia: keskushermoston *plastisuuden* eli muovautuvuuden ansiosta esimerkiksi sokeutuneen ihmisen näköaivokuoren osat alkavat vähitellen prosessoida muiden aistijärjestelmien tuottamaa dataa. Aisteja koskeva kielenkäyttö on kuitenkin vakiintunutta, eikä aistin käsitteen problematisointi ole mielekästä tämän tutkielman puitteissa, varsinkin kun näkö, kuulo, tunto, maku ja haju ovat tiedon aistillistamisen piirissä käytettyjä käsitteitä.

Kullakin aistijärjestelmällä on oma *havaintoalueensa*, jolle voimakkuudeltaan sijoittuviin ärsykkeisiin ne kykenevät reagoimaan. Intensiteetiltään (voimakkuudeltaan) vähäisintä, arvaustodennäköisyyttä luotettavammin havaittavissa olevaa ärsykevoimakkuutta kutsutaan ärsykkeen *kynnysarvoksi*.

Ärsykkeiden intensiteetistä puhuttaessa on syytä mainita myös *adaptaatio*. Se tarkoittaa aistijärjestelmän sopeutumista käsittelemiinsä ärsykkeisiin siten, että sen reagointi usein toistuviin ärsykkeisiin heikkenee. Tämä edistää poikkeavien ärsykepiirteiden havaitsemista, mikä takaa, että aistijärjestelmä käyttää kapasiteettiaan parhaansa mukaan ympäristöstä toiseen. Esimerkiksi näköaisti sopeutuu ympäristön valon määrään seurauksin, jotka kuka tahansa toimivalla näköaistilla varustettu on saanut kokea tullessaan valoisista ulkotiloista hämärään eteiseen.

On syytä huomioida myös, etteivät aistit hahmota ärsykkeiden ominaisuuksia lineaarisesti toisin kuin keinotekoiset havaintolaitteet. Aistit ovat ennemmin *logaritmisesti* skaalautuvia: on esim. helppoa huomata ero yhden ja kahden kilon painoissa, mutta ei 20 ja 21 kilon välillä. Kun ärsykkeen intensiteetti kasvaa, samalla kasvaa pienin kahden ärsykkeen välillä havaittavissa oleva ero.

3.1. Näkö

Näköaisti on aisteista tutkituin, mikä ei ole ihme, sillä se käsittelee enemmän tietoa kuin muut aistit yhteensä. Oma osoituksensa sen merkityksestä on myös primäärin näköaivokuoren koko suhteessa muille aisteille omistautuneisiin aivoalueisiin.

Näköaisti vastaanottaa valoa ja tulkitsee ärsykkeestä piirteitä, joista mainittavan arvoisia ovat *kirkkaus/vaaleus*, *väri*, *muoto*, *syvyys*, *liike* ja *hahmot*. Näiden piirteiden havaitseminen tapahtuu useassa vaiheessa verkkokalvolta hermoratojen kautta useille eri aivoalueille levittäytyvässä järjestelmässä.

3.2. Kuulo

Nuorilla ja terveillä ihmisillä kuuloaisti vastaanottaa sisäkorvan kuuloelinten kautta suunnilleen taajuusalueelle 20–20000 Hz sijoittuvaa värähtelyä ja tunnistaa siitä ominaisuuksia kuten *äänenvoimakkuuden*, *taajuuden* ja *sävyn*. Kuuloaistin osalta on syytä huomioida sen läheinen suhde kielelliseen prosessointiin, jota ilmentää mm. nk. *cocktail party -ilmiö*, jossa tietoisuuden nappaa puoleensa jokin tietoisen tarkkaavaisuuden ulkopuolella ollut kielellinen kuuloärsyke [von der Malsburg and Schneider, 1986].

Toinen huomionarvoinen piirre kuuloaistissa on nk. *mismatch negativity* [Rinne et al., 2000]. Se on elektroenkefalografiassa (EEG) kuuloaivokuorella havaittava vaste, joka syntyy, kun rytmisessä ja samantaajuisesti toistuvassa äänisarjassa tapahtuu muutos. Se osoittaa, kuinka syvällä aivotoiminnassamme äkillisten äänierojen havaitseminen on.

3.3. Tunto

Tuntoaistin toiminta perustuu iholla mm. *painetta* ja *lämpötilaa* aistiviin hermosoluihin. Keskushermostoa tarkastelemalla selviää, että erillisiksi mielletyt *asento-* ja *tasapainoaisti* ovat myös tiiviisti sidoksissa tuntoaivokuoreen ja siten tosiasiallisesti tuntoaistin alaosia.

Ihon tuntoreseptorit eivät ole läpi kehon tasaisesti jakautuneita, vaan niissä on suhteellisesti huomattavan suuri tiheys käsissä ja kasvoissa. Nämä alueet siis ovat useita kertaluokkia tuntoherkempiä kuin esim. selkä ja käsivarret. Sama painotus on havaittavissa myös näiden ruumiinosien tuntoärsykkeitä käsittelevien aivokuoren osien suhteellisen suurena kokona.

3.4. Haju ja maku

Aiemmin mainituista aisteista poiketen *haju-* ja *makuaisti* (olfaktorinen ja gustatorinen aisti) perustuvat kemikaalien kanssa reagoiviin soluihin nenäontelossa ja suussa. Käytännössä erilaiset reseptorit ottavat vastaan kolmeulotteiselta muodoltaan eroavia molekyylejä. Kukin solutyyppi välittää tiettyä hajun tai maun piirrettä, ja varsinaiset kokemukset hajusta tai mausta syntyvät monimutkaisista ärsykkeiden kokonaisuuksista.

4. Tiedon visualisoinnista

Puhuttaessa *tiedon visualisoinnista* on syytä ensin määritellä käsitteen sisältö. Sanalla *visualisointi* voidaan tarkoittaa visuaalisen kuvan luomista mielessä tai, kuten tässä tutkielmassa, visuaalisen esityksen luomista tiedon kommunikointiin.

Tiedon visuaalisen esittämisen historia on ihmisen kulttuurin näkökulmasta pitkä, mikä ei ole ihme ottaen huomioon kohdassa 3.1 kuvatun ihmisen näköaistin merkityksen. Esimerkiksi kartat ja kirjoitus ovat varhaisia tiedon visuaalisen esittämisen muotoja.

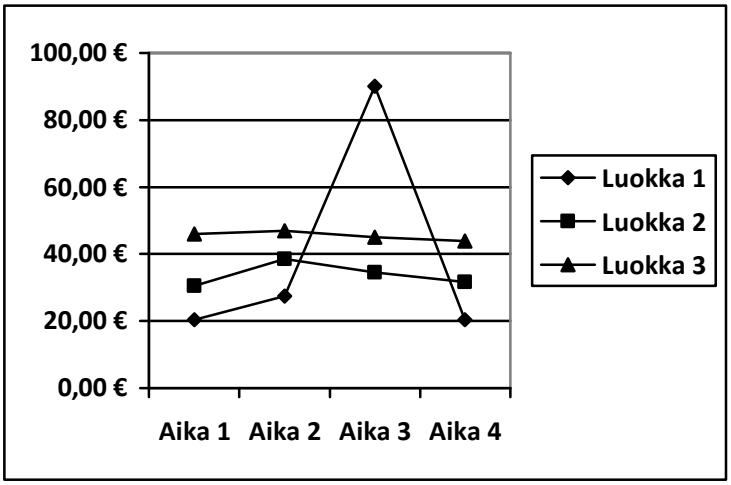
Karttojen ja kirjoitusmerkkien kautta on mahdollista esitellä eräs tiedon visualisoinnille olennainen kysymys: *aistillisten* ja *mielivaltaisten* symbolien suhde. Siinä missä kartat ovat visuaalisia jäljitelmiä maaston fyysisistä piirteistä, kirjaimet ovat mielivaltaisia symboleja, jotka muodostavat sanoja. Sanat ovat niin ikään enemmän tai vähemmän mielivaltaisia edustuksia käsitteistä. Esimerkiksi sanalla ”kissa” ei ole mitään suoraa vastaavuutta kuvaamansa eläimen kanssa, vaan sen merkitys on pitänyt oppia, jotta nuo viisi kirjainta merkitsisivät lukijalleen muutakin kuin satunnaista merkkijonoa. Sen sijaan kuva kissasta välittää visuaalista tietoa, jonka kuka tahansa näkemiseen kykenevä pystyy vastaanottamaan ilman ennakkotietoja. Teknisesti siis niin kuvat kuin kirjoituskin ovat tiedon visuaalista esittämistä, mutta tiedon visualisoinnin käsite viittaa ennemmin visualisointeihin, jotka välittävät merkityksiä ensisijaisesti visuaalisen havaitsemisen luontaisten mekanismien, eivätkä mielivaltaisten symbolien kautta [Ware, 2012].

Karttoja sivuten on selvennettävissä myös *tieteellisen visualisoinnin* ja *tiedon visualisoinnin* suhde. *Tieteellinen visualisointi* tarkoittaa fysikaalisten ilmiöiden visuaalista esittämistä. Esimerkiksi ihmisen anatomian kolmeulotteinen mallintaminen analyysiä varten on tieteellistä visualisointia. Sana tieteellinen visualisointi ei implikoi, että tiedon visualisointi ei olisi tieteellistä tai että tieteellinen visualisointi ei sisältäisi tietoa. Nimien ristiriitaisuus on historiallista painolastia siitä, että tieteellinen visualisointi on käsitteenä vanhempi. Fyysisestä maailmasta hahmonsa saavat kartat ovat läheistä sukua juuri tieteellisille visualisoinneille.

Tiedon visualisointi sen sijaan on visuaalisen esityksen luomista datasta, jolla ei ole luonnostaan fysikaalista hahmoa. Tyypillisiä tiedon visualisointeja ovat erilaiset diagrammit. Esimerkiksi pylväs- ja ympyrädiagrammit ovat mediassa yleisesti käytettyjä datan esittämisen välineitä.

Kuva 2 on hyvin yksinkertainen staattisen visualisoinnin periaatteita havainnollistava viivadiagrammi. Siinä x- ja y-akseli kuvaavat datan eri ulottuvuuksia, kuten myös se, millaisella symbolilla kutakin tietoalkiota merkitään.

Se on siis muodostettu kolmeulotteisesta datasta, jossa jokainen tietoalkio sisältää kolme arvoa: Viivalla toisiinsa yhdistetyillä symboleilla koodatun luokan, x-akselille koodatun aika-luokan ja y-akselille koodatun euromääräisen arvon.



Kuva 2. Esimerkki tiedon visuaalisesta esityksestä.

Taulukko 1. Esimerkki tiedon numeerisesta esityksestä.

	Aika 1	Aika 2	Aika 3	Aika 4
Luokka 1	20,40 €	27,40 €	90,00 €	20,40 €
Luokka 2	30,60 €	38,60 €	34,60 €	31,60 €
Luokka 3	45,90 €	46,90 €	45,00 €	43,90 €

Taulukko 1 sisältää täsmälleen saman datan kuin kuva 2, mutta siinä datan suhteelliset ominaisuudet eivät erotu lainkaan yhtä nopeasti ja selkeästi. Verratakseen tietoalkioita taulukossa 1 lukija joutuu kiinnittämään katseensa kerrallaan kuhunkin taulukon soluun, lukemaan sen arvon ja pitämään mielessään sekä tuon arvon, että rivin ja sarakkeen koodaamat luokat lukiessaan toisen solun arvon. Vastaavasti kuvassa 2 tietoalkioiden arvojen keskinäiset suhteet erottuvat pikaisella vilkaisulla, varsinkin kun samaan luokkaan kuuluvat alkiot on yhdistetty toisiinsa viivoilla, jolloin ne muodostavat hyvin nopeasti havaittavia hahmoja.

Voidaankin sanoa, että tiedon visualisoinnissa datasta muodostuu informaatiota, kun se saa visuaalisen hahmon. Sama informaatio, joka on taulukossa 1 implisiittistä ja vasta järkeilyn kautta saavutettavaa, on välittömästi nähtävissä kuvassa 2.

Datan tarkkojen arvojen lukeminen ei kuitenkaan onnistu kuvasta 2 lainkaan yhtä hyvin kuin taulukosta 1. Ihmisen ja teknologian vuorovaikutuksen tapauksessa visualisoinnit ovatkin usein vuorovaikutteisia, jolloin käyttäjä ky-

kenee hallitsemaan visualisointia eri tavoilla, kuten rajaamalla siinä esitettyjä tietoalkioita tai noutamalla lisätietoja reaaliaikaisesti jostain visualisoinnin yksityiskohdasta [Shneiderman, 1996]. Esimerkiksi kuvan 2 tapauksessa käyttäjä voisi viedä hiiren kursorin kaaviossa olevien merkkien päälle ja saada näkyviin y-akselille koodatun muuttujan tarkan arvon.

5. Modaliteeteista ja multimodaalisuudesta

Multimodaalisuus on yksi nopeimmin kehittyvistä ihminen-teknologia-vuorovaikutuksen alueista. Bernsen [2008] määrittelee sanan *modaliteetti* seuraavasti: ”Modaliteetti, tai tarkemmin informaation esittämisen modaliteetti, on tapa esittää informaatiota jonkin fyysisen väylän kautta. Siten modaliteettia määrittää sen fyysinen väylä ja sen nimenomainen esittämisen tapa”. Bernsen huomauttaa, että tämän määritelmän puitteissa modaliteetin ei tarvitse olla ihmisen havaittavissa oleva. Ihminen-teknologia-vuorovaikutuksen näkökulmasta kiinnostavia ovat kuitenkin sellaiset modaliteetit, joiden kautta ihminen ja teknologia kykenevät viestimään keskenään.

Vuorovaikutteisten järjestelmien modaaliteettien tapauksessa on huomionarvoista, että modaalisuus on kaksisuuntaista. Esimerkiksi puhelimen kautta toimiva puhekäyttöliittymä, joka ottaa syötteitä vastaan puheena ja niin ikään vastaa puheena, on niin kutsuttu *unimodaalinen järjestelmä*.

Bernsen antaa multimodaalisuudelle seuraavan määritelmän: ”*Multimodaalinen vuorovaikutteinen järjestelmä* on järjestelmä, joka käyttää vähintään kahta eri modaliteettia syötteitä ja/tai tulosteita varten. Siten [SM1, TM2], [SM1, SM2, TM1] ja [SM1, TM1, TM2] ovat joitakin yksinkertaisia esimerkkejä multimodaalisista järjestelmistä, joissa *S* tarkoittaa syötettä, *T* tulostetta ja *Mn* tiettyä modaliteettia *n*”. Näin ollen esimerkiksi perinteiset graafiset käyttöliittymät, kuten ikkunoihin, kuvakkeisiin, hiireen ja näppäimistöön perustuvat työpöytäkäyttöliittymät, ovat multimodaalisia; ne ottavat syötteitä *haptisesti* (käsien kautta) ja antavat *visuaalisia* tulosteita.

Bernsen jatkaa määrittelemällä muutamia eri modaliteetteja hyödyntävän tiedonvälityksen käsitteitä. Niistä tässä yhteydessä olennaisimpia ovat *staattinen/dynaaminen* ja *analoginen esitys*. Staattinen/dynaaminen merkitsee käyttäjän mahdollisuuksia tarkastella esitystä suhteessa aikaan. Staattinen esitys on esimerkiksi graafisen käyttöliittymän ruutu, joka on tarkasteltavissa niin kauan kuin käyttäjä haluaa. Dynaaminen esitys ei tarjoa tätä vapautta, vaan se on sidottu aikaan, kuten esim. soiva puhelin. Analoginen esitys tarkoittaa samankaltaisuutta esityksen ja esitettävän asian välillä. Luvussa 4 kuvatussa erossa kissaa nimittävän sanan ja kissan kuvan välillä on kyse juuri tästä käsitteestä.

Eri modaliteeteista puhuttaessa ei voi välttyä niiden valintaa koskevalta pohdinnalta. Bernsen [2008] määrittelee modaliteetin valintaan kuuluviksi seuraavat kysymykset:

- Onko modaliteetti M(a) hyödyllinen vai ei tarkoitukseen T?
- Onko modaliteetti M(a) enemmän vai vähemmän hyödyllinen tarkoitukseen T kuin modaliteetti M(b)?
- Onko modaliteetti M(a) *yhdessä* modaliteettien M(c, c+1, …c+n) paras multimodaalinen valinta tarkoitukseen T?

Näihin kysymyksiin monimutkaisuutta lisää kysymys siitä, käytetäänkö jotakin modaliteettia syöte- vai tulostetarkoituksessa. Tämän lisäksi mietittävänä on useita sovellusalueeseen liittyvä seikkoja, kuten käyttäjäryhmä, käyttöympäristö ja käytettävissä olevat laitteet.

Multimodaalisuuden eduista Bernsen toteaa, että ihanteellinen hyöty useiden modaliteettien käytöstä on niiden käyttö toistensa heikkouksien täydentämiseen. Useat modaliteetit myös mahdollistavat sovellusten käytön erilaisille ihmisille, kuten aistirajoitteisille henkilöille.

6. Tiedon esittäminen eri aisteille

Tiedon visualisointi on kohtuullisen laaja tieteenala vuotuisine konferensseineen ja kaupallisine sovelluksineen, mutta muiden aistien hyödyntäminen tiedon esittämiseen on saanut osakseen huomattavasti vähemmän huomiota. Tässä luvussa tarkastellaan kuulo-, tunto-, haju- ja makuaistillista tiedon esittämistä koskevia julkaisuja tarkoituksena muodostaa yleiskuva kunkin alan käsitteellisestä pohjasta ja niiden haasteista ja mahdollisuuksista. Tarkastelu koskee erityisesti tiedon esittämistä, mutta joissain tapauksissa, lähinnä haptiikan osalta, syöte- ja tulostemodaliteettien erottaminen ei ole mielekästä.

6.1. Kuulo

Tiedon äänellisen esittämisen alaan on laskettavissa niin yksinkertaiset huomio- ja hälytysäänet kuin varsinainen hienostuneen tiedon esittäminen, *sonifikointi*. Näistä jälkimmäinen on luonnollisesti tämän tutkielman kannalta kiinnostavampaa. Hälytysäänet antavat kuitenkin perspektiiviä myös sonifikoinnin tarkasteluun. Walker ja Kramer [2004] huomauttavat, että perinteiset huomio- ja hälytysäänet eivät välitä kuin binääristä informaatiota: palohälytysääni ilmoittaa hälytysjärjestelmän lauenneen, mutta ei sitä, millainen tulipalo on ja missä se on. He toteavat myös, että kehittyvä teknologia on avannut tien yksityiskohtaisemmille äänitiedotteille. He esittelevät tästä kaksi lähestymistapaa: *auditiiviset kuvakkeet* ja *earconit*, korvakkeet.

Kuvakkeet ovat graafisissa käyttöliittymissä yleisesti käytettyjä visuaalisia kommunikoinnin välineitä, jotka kuvaavat jotakin toimintoa tms. järjestelmän osaa. Vastaavasti auditiiviset kuvakkeet ovat lyhyitä ääniä, jotka täyttävät samanlaista tehtävää jäljittelemällä jotakin kuvaamaansa kohteeseen yhdistettävää ääntä, kuten tulostimen tapauksessa kirjoituskonetta. Earconit puolestaan ovat viestiääniä, joiden ilmoittamille tapahtumille ei ole luonnollista kuvaavaa ääntä. Niiden merkitys on yleensä käytännössä opittava, eli ne ovat mielivaltaisia, kuten tiedoston poistamista edustava piippaus.

Suoremmaksi äänelliseksi tiedon esittämisen tavaksi Walker ja Kramer mainitsevat *audifikoinnin*, eli aaltomuotoisen datan suoran muuttamisen ääneksi. Usein tämä vaatii datan taajuuden ja esitysnopeuden muuttamista sellaiseksi, että se on ihmisen kuultavissa. Walker ja Kramer ottavat esimerkeiksi seismisen ja avaruusluotaimen tuottaman datan audifikoinnin. Esimerkeissä data oli liian monimutkaista tai se sisälsi liikaa kohinaa, jotta sen visualisoinnista olisi voinut hahmottaa piirteitä luotettavasti, kun taas audifikoinnin myötä jotkin sen piirteistä tulivat havaittaviksi.

Walker ja Kramer jatkavat mainitsemalla äänien käyttämisen dynaamisina *prosessien valvojina*, jotka hyödyntävät ”kuulijan kykyä havaita pieniä muutoksia auditiivisissa tapahtumissa tai käyttäjän tarvetta pitää silmänsä vapaina muita tehtäviä varten”. Heidän esimerkeissään tällaista äänellistämistä on käytetty erilaisilla tuotantolinjoilla ja terveydenhuoltoympäristössä.

Hienostuneimmaksi tiedon äänellistämisen tavaksi Walker ja Kramer määrittelevät ”datan suhteiden muuttamisen akustisen signaalin havaittaviksi suhteiksi kommunikoinnin tai tulkinnan mahdollistamiseksi”. Tätä he kutsuvat *sonifikoinniksi*. Tunnettuna esimerkkinä geigermittari toteuttaa tätä periaatetta, sillä se ilmaisee säteilytiheyttä tikityksen taajuudella.

Walker ja Kramer kuvaavat sonifikointia *auditorisia graafeja* luovaksi menetelmäksi, joka on analoginen visuaalisia graafeja luoville visualisoinneille. Tällöin sarjallisesta datasta muodostetaan esitys, joka vertautuu visuaaliseen graafiin esim. koodaamalla x-akselin data auditorisen graafin aikayksiköiksi ja y-akselin data joksikin äänen ominaisuudeksi, kuten äänenkorkeudeksi. He sanovat sonifikoinnilla olevan käyttöä monimutkaisen ja moniulotteisen datan esittämisessä ja ottavat esimerkiksi tutkimuksen, jossa koehenkilöt valvoivat kahdeksaa yhtäaikaisesti vaihtelevaa muuttujaa anestesiologin työpisteellä. Henkilöt suoriutuivat tehtävästä merkittävästi paremmin sonifikoinnin kuin visualisoinnin avulla.

Walker [2000] esittää muitakin perusteluja sonifikoinnille. Hän väittää sonifikoinnin tarjoavan tehokkaita ja mukaansatempaavia opetusvälineitä tekemällä oppimisympäristöstä rikkaamman. Hän viittaa tutkimuksiin, joiden mukaan

oppilaat olevan innokkaampia käyttämään multimediallisia oppimisjärjestelmiä ja joiden mukaan he arvioivan tilastollisia ominaisuuksia yhtä hyvin auditiivisista kuin vain visuaalisista tai multimodaalisista histogrammeista. Sen lisäksi Walker esittää sonifikoinnin olevan arvokas keino data-analyysin ja tulkinnan mahdollistamisessa näkövammaisille.

Walker ja Kramer tarkastelevat joitakin äänen havaittuja ominaisuuksia ja niiden merkitystä sonifikoinnille. Äänen yleisin tietoa välittävä piirre on se, onko sitä kuultavissa tai ei. Ilmoitus- ja varoitusäänet välittävät tietoa vain olemalla kuultavissa. Tällöin *äänekkyys* on äänen olennaisin piirre: äänen on kuuluttava, mutta silti oltava tarpeeksi hiljainen, jottei se ole kuulijalle vaarallinen tai kivulias. Sonifikaation on erotuttava taustaäänistä, joten sitä suunniteltaessa on otettava huomioon *äänen peittymisen* vaikutukset. *Äänenkorkeudesta* on huomioitava, että vaikka terveiden nuorten ihmisten kuulotaajuus on noin 20–20000 Hz, äänenkorkeuden vaihtelut ovat havaittavimpia 3000 hertsin tuntumassa. Taajuusalue 200–5000 hertsiä onkin tavallinen tyypillisille soittimille. On myös huomioitava, että jotkin äänen havaitut ominaisuudet ovat *vuorovaikutuksessa keskenään*. Esimerkiksi äänekkyys vaikuttaa siihen, kuinka korkeana ääni koetaan. *Tempo ja rytmi* ovat hyvin hyödyllisiä äänen piirteitä datan esittämisessä, sillä ihmisten on havaittu erottavan helposti muutoksia näissä ominaisuuksissa, paljon luotettavammin kuin visuaalisessa datavirrassa. Äänen *sävy*, eli karkeasti ne äänen ominaisuudet, jotka eivät ole äänekkyyttä, korkeutta tai tempoa, on ääneen kvalitatiivinen piirre. Esimerkiksi erilaiset yhtä aikaa soivat soittimet ovat erotettavissa toisistaan niiden sävyn perusteella, vaikka niiden äänen muut ominaisuudet olisivat lähellä toisiaan. Täten äänen sävy on hyödyllinen ominaisuus äänten erottelua vaativissa tehtävissä. Se on myös tärkeä ominaisuus auditiivisen esityksen miellyttävyyden kannalta. Äänestä on havaittavissa myös sen *suunta*. Tämä mahdollistaa erottelun eri äänilähteiden välillä ja, toisin kuin näköärsykkeiden tapauksessa, ääniärsykkeiden havainnointi ei vaadi kuulijalta mitään tiettyä asentoa suhteessa ärsykkeen lähteeseen. Tällöin tämän visuaalinen tarkkaavaisuus jää vapaaksi. Äänilähteistä on syytä huomioida myös se, että samasta kohteesta tulevat yhtäaikaiset äänet ovat erotettavissa eri *virtoihin* kuuluviksi, jos ne erottuvat toisistaan ominaisuuksiltaan. Tämä mahdollistaa *moniulotteiset* auditiiviset esitykset, joissa kuulija pystyy valvomaan lukuisia yhtäaikaisia äänivirtoja.

Walker [2002] on auditiivisia graafeja koskevissa kokeissaan havainnut niiden tulkinnassa graafien suunnitteluun vaikuttavia ongelmia. Hän havaitsi, että vaikka koehenkilöt suoriutuivat tarkasti ääniärsykkeiden ominaisuuksien suoraviivaisesta arvioinnista, esityksistä raportoidut datan käsitteelliset piirteet poikkesivat muodoltaan lähes aina ”pelkästä” äänestä raportoiduista piirteistä,

vaikka niiden muodot olivat tosiasiassa samat. Koehenkilöt eivät myöskään olleet arvioissaan yhdenmukaisia, sillä heidän havaitsemansa piirteet riippuivat siitä, millaista käsitemallia kukin käytti äänen ja datan ominaisuuksien yhteen liittämiseen. Lisäksi koehenkilöt raportoivat eroja identtisissä graafeissa sen mukaan, minkä nimisenä datan ulottuvuus oli heille kerrottu. Walker peräänkuuluttaakin empiirisesti validoitujen heuristiikkojen käyttöä auditiivisten graafien suunnittelussa, sillä suunnittelijan ja käyttäjän käsitemallien ero voi vääristää esitysten tulkintaa.

Luvussa 5 käsiteltyjen Bernsenin [2008] määritelmien kautta tarkasteltuna kuuloaistiin perustuva tiedon esittäminen hyödyntää erityisesti äänen dynaamista luonnetta ilmaisemaan aikaan sidottua tietoa. Huomion sieppaavan vaikutuksensa ansiosta se soveltuu etenkin reaaliaikaisiin valvontatehtäviin. Toisaalta äänen aikasidonnaisuus rajaa abstraktin datan esitystapoja.

Datan ominaisuuksien koodaaminen äänen eri ominaisuuksiin näyttää olevan hankalampaa kuin visuaalinen koodaus, sillä äänen ominaisuuksille ja erilaisille käsitteellisille magnitudeille ei ole yhtä universaaleja käsitemalleja. Suunnittelija ei voi luottaa käyttäjien tulkitsevan esityksiä samoin kuin hän itse.

6.2. Tunto

Siinä missä monimutkaisia näkö- ja kuuloaistimuksia tuottavat laitteet ovat arkipäivää, haptiset tulostelaitteet eivät ole yhtä yleisiä. Eräs haptisissa sovelluksissa käytetty laite on kuvassa 3 nähtävä PHANTOM [Massie and Salisbury, 1994].

Kuva 3. PHANTOM Premium 1.5 (© 2015 Sensable, kuvan lähde http://www.dentsable.com/phantom-premium-1-5.htm)

Roberts ja Panëels [2007] ovat käyneet läpi haptisen visualisoinnin historiaa ja määrittelevät alalle kuusi alakategoriaa sen mukaan, millaista dataa haptinen visualisointi esittää: *kaaviot*, *kartat*, *verkot*, *symbolit*, *diagrammit* ja *kuvat*. Kukin kategoria vaatii omanlaisensa matemaattiset mallit datan tuntoaistillistamiseen. Niille yleistä vaikuttaa kuitenkin olevan se, että niissä datan visuaalinen esitys mallinnetaan virtuaaliseen tilaan, esimerkiksi viivadiagrammin tuntoaistillistaminen onnistuu määrittämällä diagrammi tasoksi kaksiulotteiseen avaruuteen ja viiva kohoumaksi siihen, jolloin käyttäjä pystyy tunnustelemaan sitä esim. PHANTOMilla.

Roberts ja Panëels toteavat katsauksensa pohjalta, että verrattuna koko haptiikan kenttään haptisen visualisoinnin osuus tutkimuksesta on vähäinen. He myös huomauttavat sovellusten seuraavan teknologioita, jotka ovat vasta kehittymässä. He kiinnittävät huomiota vuoden 2006 Eurohaptics-konferenssiin, jossa 34 julkaisua koski laitteita, 30 haptista mallintamista ja 10 haptista visualisointia. Haptisten laitteiden rajoitteista he nostavat esiin taktiilisen palautteen ja voimapalautteen yhdistävän teknologian puutteen.

Oma aineistonhakuni tuotti minulle samanlaisen vaikutelman. Robertsin ja Panëelsin jälkeen julkaistuista datan haptista esittämistä koskevat artikkelit vaikuttavat edelleen kiinnittävän huomiota toteutusteknisiin seikkoihin suunnitteluperiaatteiden käsittelyn jäädessä vähemmälle (mm. [Faeth et al., 2008; Borst and Baiyya, 2009; Evreinova et al., 2012]). Esimerkiksi Evreinova ja muut tarkastelevat haptisen StickGrip-laitteen käyttömahdollisuuksia kaikuluotausdatan perusteella luodun syvyyskartan tarkasteluun osana visuaalista esitystä pelkkään visualisointiin verrattuna. He kiinnittävät huomiota myös laitteen mobiilin luonteen tarjoamiin mahdollisuuksiin verrattuna aiempiin ei-mobiileihin voimapalautelaitteisiin.

Panëels ja Roberts [2010] palasivat artikkelissaan tekemään tarkemman katsauksen haptisten visualisointien suunnitteluperiaatteista. Tällä kertaa he jaottelevat haptiset visualisoinnit *kaavioihin*, *karttoihin*, *merkkeihin*, *verkkoihin*, *diagrammeihin*, *kuviin* ja *taulukoihin*.

Näitä osa-alueita koskevien artikkeleiden perusteella he päätyvät toteamaan, että monet tutkijat ovat keskittyneet suunnittelemaan datan haptisesta esityksestä samankaltaisen sen visuaalisen esityksen kanssa sen sijaan, että pyrkisivät löytämään tehtävään parhaiten sopivan haptisen visualisoinnin. He havaitsivat eniten tutkimusta tehdyn *kaavioiden* parissa sekä *karttojen*, *diagrammien* ja *merkkien* saralla. He nostavat esiin navigoinnin ja yleiskäsityksen muodostamisen vaikeuden. Käyttäjät tarvitsevat käytännössä visuaalista palautetta siitä, missä päin mallinnusta heidän haptinen fokuksensa on, sillä monet nykyiset haptiset laitteet tuottavat palautetta vain yhdestä pisteestä kerrallaan. Rat-

kaisuksi tähän on tarjottu opastettuja läpikäyntejä, joissa käyttäjä viedään ennalta määritetyn haptisen reitin läpi, sekä muita modaliteetteja, kuten puhetta ja sonifikaatiota.

Vaikuttaa siltä, että tiedon tuntoaistillistamisen motivaatiot jakaantuvat karkeasti kahteen ryhmään. Jotkin niistä kuvaavat dataa, joka on joko alun perin fyysistä ja kolmiulotteista, kuten Evreinovan ja muiden analoginen syvyysdatan esitys, tai jäsennetty kolmiulotteiseksi virtuaaliympäristöksi. Toiset taas hyödyntävät tuntoaistia näköaistin korvaavana modaliteettina alun perin visuaalisten esitysten kommunikointiin.

6.3. Haju ja maku

Siinä missä teknologiat hienostuneiden tuntoaistimusten tuottaminen ovat vielä kehittymässä, pätee sama piirre haju- ja makuaistimuksiin vielä voimakkaammin. Kun näkö-, kuulo- ja tuntoaisti kaappaavat fyysisiä ärsykkeitä, olfaktorisen ja gustatorisen aistin, eli haju- ja makuaistin käsittelemät ärsykkeet ovat kemiallisia. Tämä asettaa omat huomattavat haasteensa haju- ja makuärsykkeiden keinotekoiselle tuottamiselle.

Oma ongelmansa on kemiallisten aistimusten ominaisuuksien luokittelu. Kaye [2001] on käynyt läpi hajujen luokittelun historiaa ottaen esiin mm. parfyymien ja viskien hajujen kuvaamiseen käytetyn sanaston ja päätynyt toteamaan, ettei mitään yleistä luokittelujärjestelmää ole syntynyt. Kokemus hajuista on subjektiivista ja kokemukseen perustuvaa; hajuja tunnistetaan ja nimetään sen mukaan, mikä reaalimaailman asia niitä tuottaa, kuten ”ruoho” tai ”ruusu”. Yleensä luonnolliset hajut kuten kahvi ja suklaa eivät myöskään muodostu minkään yksittäisen kemikaalin kautta, mikä hankaloittaa niiden syntetisointia.

Hajujen ja makujen käyttöä tiedon esittämiseen on tutkittu verrattain vähän ja tutkimukset ovat keskittyneet esitysteknologiaan. Narumi ja muut [2011] ovat kehittäneet pseudo-gustatoriseksi laitteeksi kutsumansa järjestelmän, jossa käyttäjä altistetaan erilaisille hajuille hänen syödessään maultaan neutraaleja keksejä. Tällöin käyttäjälle syntyy kokemus mausta. Soveltavaa tutkimusta edustaa Brewsterin ja muiden [2006] tutkimus. Siinä he havaitsivat, että kuvaaineiston merkitseminen itse valituilla hajuärsykkeillä edesauttoi muistamista, joskaan ei yhtä paljon kuin tekstimerkinnät.

Kaye esittää hajuilla olevan käyttömahdollisuuksia *hajuikonien* muodossa. Kuten Walkerin ja Kramerin [2004] kuvaamat *earconit*, hajuja voisi Kayen mukaan käyttää kuvaamaan joitakin diskreettejä käsitteitä. Ruudinhajun erittäminen aseen laukeamisen merkiksi pelissä tai ruuanlaitosta ilmoittaminen laitettuun ruokaan liittyvien hajujen avulla olisivat suoraviivaisia hajuikoneja.

Kaye määrittelee hajujen käytölle etuja ja ongelmia tiedon esittämisessä. Eduiksi hän laskee seuraavat piirteet:

- *Silmien ja korvien vapaaksi jättäminen*
- *Tunteelliset vaikutukset*: Hajut tuottavat lähes sisäsyntyisiä tunnereaktioita
- *Huomiota herättävyys*: Yhtäkkiä ilmestyvä haju vetää huomion puoleensa
- *Taustalla oleminen*: Jatkuva haju hiipuu tietoisesta tarkkaavaisuudesta
- *Ajallisuus*: Tuotettu haju ei katoa heti ärsykkeen tuottamisen jälkeen kuten valo ja ääni tekevät
- *Kumulatiivisuus*: Hajuja voi yhdistellä
- *Hajautuneisuus*: Hajun havaitseminen ei riipu asennostamme suhteessa hajulähteeseen
- *Paikallistaminen*: Kykenemme vertaamaan eri paikoissa havaitsemiamme hajuja.

Ongelmiksi Kaye lukee seuraavat piirteet:

- *Alhainen määrän resoluutio*: Emme kykene erottamaan luotettavasti hajujen voimakkuuksia
- *Absoluuttisten arvojen puute*: Emme kykene määrittämään kemikaalien absoluuttisia konsentraatioita
- *Rajallinen paikallinen tarkkuus*: Emme kykene erottamaan hajun lähdettä yhdellä nuuhkaisulla
- *Rajallinen ajallinen tarkkuus*: Hajut haihtuvat hitaasti, joten ne voivat häiritä niiden jälkeen tulevia hajuja
- *Ristiriidat*: Hajut voivat estää toistensa havaitsemista
- *Koherentin luokittelumallin puute*: Hajuille ei ole yhdenmukaista määrittelyjärjestelmää
- *Häiritsevyys*: Hajut voivat häiritä ja ärsyttää
- *Tunteelliset vaikutukset*: Tunnevaikutukset voivat olla häiritseviä
- *Allergiset vaikutukset*
- *Ylikeskittyminen*: Liiallinen keskittyminen hentoihin hajuihin voi aiheuttaa hyperventilointia
- *Nopea haihtuminen*
- *Leviäminen yksilön ympärille*: Yhtä ihmistä varten eritetyt hajut voivat levitä muiden aistittaviksi
- *Ei historiaa*: Haju ei jätä jälkiä vaan katoaa
- *Käyttäjän rajoitteet*: Kaikki hajut eivät ole ihmisen aistittavissa
- *Hajun nimen muistamisen vaikeus*: Tunnistettujenkin hajujen nimeäminen voi tuottaa hankaluuksia
- *Vaikea syntetisointi*: Mielivaltaisia hajuja ei osata generoida tilanteen mukaan.

Haju- ja makuaisti ovat siis kenties haastavimmat modaliteetit hyödynnettäviksi. Erityisesti abstraktin tiedon välittäminen on hankalaa, koska hajut ja maut yhdistyvät niin voimallisesti niitä aiheuttaviin konkreettisiin asioihin.

7. Yhteenveto

Kirjallisuuskatsaus paljasti eroja eri modaliteettien käyttömahdollisuuksissa ja niitä koskevissa tutkimuskentissä.

Kuuloaistimusten dynaamisuus tekee niistä hyvän keinon aikasidonnaisen tiedon ilmaisemiseen. Tämä piirre yhdistettynä äänien tarkkaavaisuuden sieppaavaan vaikutukseen tekee niistä tehokkaan ja yleisesti käytetyn välineen valvontatehtäviin. Abstraktin datan ja äänen ominaisuuksien yhteyden käsitteellisen monimutkaisuuden vuoksi luotettavien representaatioiden suunnittelu edellyttää tukeutumista heuristiikkoihin.

Tuntoaistiin perustuvien aistillistamistapojen kehittämistä näyttää rajoittavan teknologian ja teoreettisen pohjan kehittymättömyys. Tarjolla olevat laitteet sanelevat ja rajoittavat sovellusten kehittämistä. Datan esitykset ovat lähinnä joko analogisia esityksiä spatiaalisesta datasta, virtuaaliympäristöjä tuntoaistillistavia tai visualisointia korvaavia.

Haju- ja makuaisti ovat toteutusteknisesti kaikkein hankalimmat aistit. Lisäksi niiden kyky hienojakoisen tai abstraktin tiedon esittämiseen on verraten heikko. Niiden hyödyntämisen teoriaa luovaa tutkimusta on kuitenkin olemassa.

Kaiken kaikkiaan tuntoaisti vaikuttaa saaneen osakseen vähiten suunnitteluperiaatteita kartoittavaa tutkimusta. Teknologioiden moninaisuus tekee yhtenäisen pohjan muodostamisesta haastavaa, mutta yhtenäinen käsitteistö ja suunnitteluperiaatteet voisivat edesauttaa alan kehittymistä ja haptiikan käyttömahdollisuuksien ymmärtämistä.

Viiteluettelo

[Ackoff, 1989] Russell L Ackoff, From data to wisdom. *Journal of Applied systems Analysis* **16**, 1 (1989), 3-9.

[Bernsen, 2008] Niels Ole Bernsen, *Multimodal User Interfaces*. Springer Berlin Heidelberg, 2008.

[Borst and Baiyya, 2009] Christoph W. Borst and Vijay B. Baiyya, A 2D haptic glyph method for tactile arrays: Design and evaluation. In: *Proceedings of the World Haptics 2009-Third Joint EuroHaptics conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (2009), IEEE, 599-604.

[Brewster et al., 2006] S. Brewster, D. McGookin and C. Miller, Olfoto: designing a smell-based interaction. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2006), ACM, 653-662.

[Evreinova et al., 2012] Tatiana V. Evreinova, Grigori Evreinov, and Roope Raisamo, Haptic visualization of bathymetric data. In: *Proceedings of the IEEE Haptics Symposium (HAPTICS)* (2012), IEEE, 359-364.

[Faeth et al., 2008] Adam Faeth, Michael Oren and Chris Harding. Combining 3-D geovisualization with force feedback driven user interaction. In: *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* (2008), ACM, 25-29.

[Kaye, 2001] Joseph Nathaniel Kaye. Symbolic olfactory display. Doctoral dissertation, Massachusetts Institute of Technology, 2001.

[Massie and Salisbury, 1994] Thomas H. Massie and J. Kenneth Salisbury. The phantom haptic interface: A device for probing virtual objects. In: *Proceedings of the ASME Winter Annual Meeting, Symposium on Haptic Interfaces for Virtual Environment and Teleoperator systems* **55.1** (1994), 295-300.

[Narumi et al., 2011] T. Narumi, S. Nishizaka, T. Kajinami, T. Tanikawa and M. Hirose, Augmented reality flavors: gustatory display based on edible marker and cross-modal interaction. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2011), ACM, 93-102.

[Panëels and Roberts, 2010] Sabrina Panëels and Jonathan C. Roberts, Review of designs for haptic data visualization. *IEEE Transactions on Haptics* **3** (2010), 119-137.

[Purves et al., 2008] Dale Purves, Elizabeth M. Brannon, Scott A. Huettel, Kevin S. LaBar, Michael L. Platt and Marty G. Woldorff, *Principles of Cognitive Neuroscience*. W. H. Freeman, 2008.

[Rinne et al., 2000] T. Rinne, K. Alho, R.J. Ilmoniemi, J. Virtanen and R. Näätänen, Separate time behaviors of the temporal and frontal mismatch negativity sources, *Neuroimage* **12** (2000), 14-19.

[Roberts and Panëels, 2007] Jonathan C. Roberts and Sabrina Panëels, Where are we with Haptic Visualization?. In: *Proceedings of the Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (2007), IEEE Computer Society, 316-323.

[Rowley, 2007] Jennifer E. Rowley, The wisdom hierarchy: representations of the DIKW hierarchy, *Journal of Information Science* **33** (2007), 163-180.

[Shneiderman, 1996] Ben Shneiderman, The eyes have it: A task by data type taxonomy for information visualizations. In: *Proceedings of IEEE Symposium on Visual Languages* (1996), IEEE, 336-343.

[von der Malsburg and Schneider, 1986] Christoph von der Malsburg and Werner Schneider, A neural cocktail-party processor. *Biological Cybernetics* **54**, 1 (1986), 29-40.

[Walker, 2000] Bruce N. Walker, Magnitude estimation of conceptual data dimensions for use in sonification. Doctoral dissertation, Rice University, 2000.

[Walker, 2002] Bruce N. Walker, Magnitude estimation of conceptual data dimensions for use in sonification. *Journal of Experimental Psychology: Applied* **8**, 4 (2002), 211-221.

[Walker and Kramer, 2004] Bruce N. Walker and Gregory Kramer, *Ecological Psychoacoustics.* Elsevier Academic Press, 2004.

[Ware, 2012] Colin Ware, *Information Visualization: Perception for Design*. Elsevier, 2012.

Naiivi Bayesin algoritmi

Kaisa Tuisku

Tiivistelmä.
Yksinkertainen naiivi Bayes-luokittelija on koneoppimisalgoritmi, joka luokittelee uusia esimerkkejä opetusdatan diskreettien attribuuttien perusteella. Tässä tutkielmassa perehdytään naiivin Bayesin algoritmin toimintaperiaatteisiin ja sovellusalueisiin kirjallisuuskatsauksen menetelmin. Tutkielman kokeellisessa osuudessa esitellään algoritmin toimintaa aineistoon soveltamalla.

Avainsanat ja -sanonnat: tiedonlouhinta, koneoppiminen, naiivi Bayesin algoritmi, riippumattomuusoletus.

1. Johdanto

Naiivi Bayesin algoritmi [Mitchell, 1997; Hand *et al.*, 2001] on käytännöllinen lähestymistapa tiedonlouhinnan luokittelutehtäviin. Ennustavaan luokitteluun yleisesti käytettynä ja tunnettuna koneoppimismenetelmänä naiivia Bayesin algoritmia on sovellettu lukuisilla eri alueilla. Huolimatta algoritmin yksinkertaisesta ja suoraviivaisesta lähestymistavasta luokittelutehtäviin, Mitchell kertoo sen toimivan varsin hyvin monimutkaisillakin sovellusalueilla.

Bayesilaisten algoritmien tavoin naiivi Bayesin algoritmi hyödyntää Bayesin teoreemaa todennäköisyyksien estimoimiseen. Algoritmin naiivius ilmenee sen käytännössä harvoin todenmukaisesta oletuksesta, jossa attribuutit katsotaan toisistaan riippumattomiksi. Kun jokaista attribuutin arvoa voidaan tarkastella itsenäisesti, laskettavien todennäköisyyksien määrä vähenee merkittävästi ja algoritmin soveltaminen käytännön ongelmiin on mahdollista. Huolimatta riippumattomuusoletuksesta, Domingos ja Pazzani [1997] ovat todenneet luokittelijan toimivan yllättävän hyvin myös aineistoilla, jonka attribuuttien välillä on voimakkaita riippuvuussuhteita.

Naiivista Bayesin algoritmista on olemassa lukuisia tutkimuksia, joissa algoritmin luokittelumenestys on ollut samalla tasolla päätöspuiden tai neuroverkkojen tapaisten algoritmien kanssa [Mitchell, 1997; Domingos and Pazzani, 1997]. Algoritmin yksinkertaisin muoto soveltuu kuitenkin ainoastaan aineistoille, jonka attribuutit ovat diskreettejä tai järkevästi diskretoitavissa ilman tärkeän informaation menettämistä. Naiivia Bayesin algoritmia on hyödynnetty paljon esimerkiksi tekstidokumenttien luokitteluun [Mitchell, 1997] ja sairauksien diagnosointiin [Lavrač and Zupan, 2005].

Tässä tutkielmassa esitellään yksinkertaisen naiivin Bayesin algoritmin ominaisuuksia ja käyttösovelluksia lähdekirjallisuuden avulla. Tutkielmassa on myös lyhyt kokeellinen osuus. Luvuissa 2 ja 3 määritellään muutamia aiheeseen liittyviä käsitteitä ja näiden välisiä suhteita. Luvussa 4 selvennetään naiivin Bayesin algoritmin periaatteita ja algoritmin yksinkertaisen muodon käyttöön vaadittavaa esikäsittelyä. Luvussa 5 esitellään muutamia algoritmin sovellusalueita. Luvussa 6 havainnollistetaan naiivin Bayesin mallin käyttöä esimerkkiaineiston avulla. Tutkielman pääasiat on koottu yhteenvetoon luvussa 7.

2. Määritelmiä

2.1. Tiedonlouhinta

Tiedonlouhinnasta puhuttaessa tarkoitetaan useimmiten odottamattomien, yksittäisten attribuuttien välisten tai koko dataa mallintavien suhteiden eli mallien ja hahmojen etsimistä havainnoidusta aineistosta [Hand *et al.*, 2001]. Tarve tiedonlouhintaan on syntynyt, kun tietoyhteiskunnan kehittyminen on tuonut mukanaan valtavia tietovarastoja, joiden analysointiin perinteiset tilastotieteen menetelmät eivät sovellu. Useimmissa tapauksissa tietoa ei ole koottu louhintaprosessia varten, vaan se on kerätty ensisijaisesti muuhun tarkoitukseen käytettäväksi.

Tiedonlouhinnan tavoitteena on etsiä aineistoista sen käyttäjälle uutta, merkityksellistä ja ymmärrettävää tietoa sekä tiivistää ja visualisoida datan tarjoamaa tietoa ymmärrettävään muotoon. Se on osa tietämyksen muodostamisen prosessia, johon kuuluu tiedonlouhinnan lisäksi datan valinta ja sen esikäsittely tiedonlouhinnan mahdollistavaan muotoon ja lopuksi tulosten tulkinta ja arviointi. [Hand *et al.*, 2001]

Tiedonlouhinnassa on paljon sekä itse louhintaan että käsiteltävään dataan liittyviä haasteita. Myös tilastollisesta analyysista tutut ongelmat pätevät tiedonlouhinnan tapauksessa, mutta niiden huomaaminen on vaikeampaa aineiston suuren koon vuoksi. Handin ja muiden [2001] mukaan ongelmiksi nousevat usein datan edustavuus ja sattuman merkitys datasta löydetyille suhteille sekä laajoja aineistoja käsitellessä tiedonlouhintaprosessiin käytettävän ajan määrä. Kuten aiemmin todettiin, tiedonlouhinnassa käsiteltävä data on vain harvoin tiedonlouhintaprosessia varten kerätty, jolloin datan esikäsittely tiedonlouhinnan mahdollistavaan muotoon on usein välttämätöntä. Tässä tutkielmassa käsitellään lyhyesti myös datan esikäsittelyä sen keskeisen aseman vuoksi.

2.2. Koneoppiminen

Kovahi ja Provost [1998] määrittelevät koneoppimisen tieteenalaksi, joka tutkii aineistosta oppivia algoritmeja. Oppimiseen tarkoitettu tietokoneohjelma suorittaa tehtäviä, joissa suoriutumista voidaan mitata. Ohjelman suoriutuessa tehtävässä aikaisempaa pa-

remmin, sen voi katsoa oppineen. Koneoppimisalgoritmit ovat hyödyllisiä erityisesti tiedonlouhinnan ongelmissa, joissa tutkitaan vaikeasti käsitettävän suuria tietomassoja huonosti tunnetuilla alueilla, joissa ihmisen ymmärrys ei välttämättä riitä. Koneoppimista onkin sovellettu monilla tieteenaloilla, kuten tekoälyteknologiassa, informaatiotieteissä, filosofiassa ja psykologiassa.

Hyvin määritelty koneoppimisohjelman koostuu kolmesta komponentista: oppimistehtävästä, oppimisen mittarista ja harjoituskokemuksesta [Mitchell, 1997]. Suorittaakseen oppimistehtäväänsä, ohjelmalla on oltava käytössään opetteluun soveltuvaa materiaalia. Huomioonotettavia seikkoja ohjelman harjoittelukokemuksen suunnittelussa ovat harjoittelusta ohjelman saatavissa oleva palaute, oppijan oman kontrollin aste oppimiseensa ja oppimiskokemuksen edustavuus eli harjoittelutehtävien jakautuminen tasaisesti esimerkkien muodostamalle alalle. Näiden osa-alueiden toteutumisella on suuri vaikutus ohjelman oppimiskykyyn [Mitchell, 1997].

Useimmat koneoppimissovellukset sisältävät haun hypoteesiehdokkaiden joukosta oppimistehtävään parhaiten soveltuvan hypoteesin löytämiseksi. Naiivi Bayesin algoritmi on poikkeus tähän sääntöön.

3. Tiedonlouhinta-algoritmi

Tiedonlouhinta-algoritmit suorittavat tiedonlouhintatehtäviä. Niiden toiminta on teoriassa yksinkertaista. Algoritmi ottaa syötteenä analysoitavan datan ja palauttaa löydetyt hahmot tai mallit. Tiedonlouhintatehtäviä suorittavat tiedonlouhinta-algoritmit koostuvat neljästä osasta [Hand *et al.,* 2001]: mallin rakenne, pisteytysfunktio, hakumenetelmä ja aineiston hallintamenetelmä. Näihin osiin liittyvien valintojen kautta voidaan muodostaa suuri määrä erilaisia tiedonlouhinta-algoritmeja. Tiedonlouhintatehtävän ratkaisemiseksi sopivien algoritmin osien valinta tulisikin suorittaa tehtävälähtöisesti pohtimalla, mitkä rakenteet ja menetelmät sopivat ongelman luonteeseen [Hand *et al.,* 2001]. Esimerkiksi mallin suhteen on usein päätettävä, voidaanko käyttää datan normaalijakautuneeksi olettavaa mallia.

Tiedonlouhintatehtävä on tiedonlouhinta-algoritmin tavoite. Tavoite voi olla olemassa olevan datan kuvailu tai ennustaminen mallien tai hahmojen avulla. Kuvaileva malli tai hahmo kertoo datan olemassa olevista ominaisuuksista ymmärrettävässä ja käytännöllisessä muodossa. Kuvailevaa tiedonlouhintaa on esimerkiksi tiedon visualisointi todennäköisyysjakaumin tai klusterianalyysin menetelmin. Ennustavan mallin tehtävänä on nimensä mukaisesti ennustaa uusien esimerkkien ominaisuuksia tunnetun datan perusteella. Ennustavaa mallintamista on esimerkiksi uuden esimerkin luokittelu tai regressio. Tässä tutkielmassa tarkastellaan naiivia Bayesin algoritmia, jonka tiedonlouhintatehtävä on ennustava luokittelu.

Tiedonlouhinta-algoritmin ensimmäinen osa on mallin tai hahmon rakenne, johon yritetään sovittaa analysoitavaa dataa. Mallirakenteet kuvaavat dataa kokonaisuutena,

kun taas hahmorakenteet keskittyvät datan paikallisiin ominaisuuksiin, jopa kahden attribuutin välisiin suhteisiin. Valittu rakenne määrittelee reunaehdot sille, mitä datasta on mahdollista oppia.

Pisteytysfunktiota käytetään tiedonlouhinta-algoritmin tuloksen, mallin tai hahmon laadun mittaamiseen. Mallin tai hahmon tavoitteena on saada pisteytysfunktiolle paras mahdollinen arvo. Pisteytysfunktio siis asettaa ehdokasmallit ja hahmot järjestykseen niiden paremmuuden mukaan. Yksi yleisesti käytetty pisteytysfunktio on ennustettujen arvojen ja tavoitearvojen neliövirheiden summa.

Hakumenetelmä on tiedonlouhinta-algoritmin oleellinen osa, joka etsii pisteytysfunktion parhaan mahdollisen arvon käytetylle mallille tai hahmolle. Tämä laskennallinen menetelmä siis sovittaa mallin tai hahmon louhittavalle datalle. Hakumenetelmän tehokkuus on erittäin tärkeää suuria datamassoja käsitellessä.

Datan hallinta kattaa säilyttämisen lisäksi järjestämisen eli indeksoinnin ja lopulta tiedon hakemisen muistista. Datanhallintatekniikkana käytetään usein keskusmuistiin tallentamista, mutta hyvin suurien tietomassojen ollessa kyseessä datan hallinnan suunnitteluun on syytä kiinnittää huomiota, sillä toissijaisen muistin tehoton käyttö voi hidastaa algoritmin toimintaa merkittävästi.

Suurin osa aineiston mallien tai hahmojen löytämiseen tai kuvailuun tarkoitetuista tiedonlouhinta-algoritmeista hyödyntää koneoppimisalgoritmeja [Witten and Franke, 2000]. Moni koneoppimisalgoritmi ei kuitenkaan määrittele datanhallintatekniikkaa, joka on tiedonlouhinta-algoritmin tärkeä osa-alue [Hand *et al.*, 2001]. Tiedonlouhinta-algoritmin ja koneoppimisalgoritmin käsitteet ovatkin osittain päällekkäisiä. Tiedonlouhinta-algoritmi kuvaa laajempaa kokonaisuutta, jonka osa koneoppimisalgoritmi voi olla.

4. Toimintaperiaatteita

Naiivista Bayesin algoritmista on olemassa useita erilaisia versioita [Hand *et al.*, 2001], joista tässä tutkielmassa on valittu tarkasteltavaksi ainoastaan diskreettejä attribuutteja käsittelevä yksinkertainen naiivi Bayesin algoritmi. Tässä luvussa esitellään yksinkertaisen naiivin Bayesin algoritmin periaatteita ja sen käyttöön vaadittavaa esikäsittelyä.

4.1. Esikäsittely

Yksinkertainen naiivi Bayesin algoritmi luokittelee ainoastaan diskreettejä attribuutteja, joten mahdolliset jatkuvat attribuutit on diskretoitava ennen naiivin Bayes-luokittelijan opetusta tai käyttöä. Diskretisoinnilla tarkoitetaan jatkuvien attribuuttien saamien arvojen jakamista muutamaan symboleilla esitettävään väliin. Jako voidaan toteuttaa yksinkertaisin tilastotieteellisin menetelmin tai luokkaperustaisesti [Richeldi and Rossotto, 1995].

Diskretisointiin käytettäviä yksinkertaisia tilastollisia menetelmiä kutsutaan myös luokalle sokeiksi menetelmiksi. Muutamia tunnettuja esimerkkejä tällaisista menetel-

mistä ovat jatkuvien attribuuttien arvojen jakaminen samanpituisiin tai saman frekvenssisiin väleihin. Molemmissa menetelmissä välien lukumäärä on määritelty etukäteen. Kun prosessissa ei huomioida opetusdatan luokkia, voidaan menettää luokittelualgoritmin kannalta tärkeää informaatiota eikä luokittelualgoritmi enää pysty erottamaan eri luokkien esimerkkejä saman välin sisällä [Richeldi and Rossotto, 1995]. Ongelma voidaan ratkaista suorittamalla diskretisointi luokkaperustaisesti.

Luokkaperustaisen diskretisoinnin alkutilanteessa jokaisen esimerkin diskretoitavan attribuutin saama arvo muodostaa oman ryhmänsä. Esimerkit järjestetään attribuutin arvojen mukaiseen järjestykseen. Vierekkäiset ryhmät yhdistetään, jos niiden suhteelliset luokkafrekvenssit ovat tarpeeksi lähellä toisiaan. Ryhmien luokkafrekvenssien samanlaisuuden selvittämiseen voidaan käyttää erilaisia mittareita, kuten informaation hyötysuhdetta [Catlett, 1991] tai χ^2-riippumattomuustestiä [Kerber, 1992]. Arvojen väli on ryhmään kuuluvien arvojen minimin ja maksimin rajaama alue.

4.2. Algoritmi

Naiivi Bayesin malli on tiedonlouhinta-algoritmi, jonka tehtävänä on ennustava luokittelu [Mitchell, 1997]. Algoritmin toiminta perustuu opetusdatan arvojen frekvensseistä laskettuihin ehdollisiin todennäköisyyksiin. Todennäköisyyksien yhdistelmistä muodostetaan opittu hypoteesi, jolla uudet esimerkit luokitellaan. Naiivi Bayes-luokittelija ei muodosta mahdollisten hypoteesien joukkoa myöhempää hakua varten kuten suurin osa muista luokittelualgoritmeista. Tästä syystä naiivi Bayesin malli toimiikin tehokkaasti vieden vain vähän muistitilaa. Lisäksi algoritmin toteuttaminen ja sen toiminnan tulkitseminen on yksinkertaista [Hand *et al.*, 2001].

Naiivi Bayesin algoritmi noudattaa bayesilaista lähestymistapaa luokitteluun: uusi esimerkki luokitellaan attribuuttien arvojen perusteella todennäköisimpään luokkaan. Naiivi Bayesin algoritmi olettaa attribuutit toisistaan luokan suhteen ehdollisesti riippumattomiksi. Termi ’naiivi’ algoritmin nimessä kertookin tästä ominaisuudesta, sillä todellisuudessa attribuutit ovat hyvin harvoin riippumattomia. Oletuksen ansiosta vältetään tarve estimoida eksponentiaalinen määrä todennäköisyyksiä. Luokitteluun tarvittavien todennäköisyyksien määrä kasvaa enää lineaarisesti attribuuttien määrän kasvaessa ja uuden esimerkin luokan ennustaminen vaatii vain esimerkin attribuuttiarvojen todennäköisyyksien tulon laskemisen eri luokissa [Mitchell, 1997].

Naiivissa Bayesin algoritmissa kullekin luokan Y arvolle y lasketaan todennäköisyys sillä ehdolla, että attribuuteilla X_i on luokiteltavan esimerkin arvot x_i:

$$P(Y = y \mid X_1 = x_1, X_2 = x_2, \dots, X_n = x_n) = P(Y = y) \prod_{i=1}^{n} P(X_i = x_i | Y = y) \cdot$$

Esimerkin luokaksi valitaan se, jonka todennäköisyys on suurin.

Edellä annettu kaava voidaan johtaa Bayesin teoreemaa ja riippumattomuusoletusta hyödyntämällä. Luokan todennäköisyys on Bayesin teoreeman avulla esitettynä

$$P(Y = y \mid X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)$$
$$= \frac{P(Y = y)P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n \mid Y = y)}{P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)} \cdot$$

Koska teoreeman nimittäjä on kaikilla luokilla sama, voidaan tarkastella vain osoittajaa. Olettamalla attribuutit ehdollisesti riippumattomiksi saadaan

$$P(Y = y)(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n \mid Y = y)$$
$$= P(Y = y)P(X_1 = x_1 \mid Y = y) \ldots P(X_n = x_n \mid Y = y)$$
$$= P(Y = y) \prod_{i=1}^{n} P(X_i = x_i | Y = y) \cdot$$

Tästä päädymme takaisin alussa esitettyyn muotoon

$$P(Y = y \mid X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = P(Y = y) \prod_{i=1}^{n} P(X_i = x_i | Y = y) \cdot$$

Erityisesti pienten aineistojen ongelmana ovat tilanteet, joissa tietty attribuutin arvo ei esiinny luokassa. Yhden attribuutin nolla-arvoinen todennäköisyys tavoiteluokassa nollaa myös kokonaistodennäköisyyden, mikä näin ollen tuottaa epäluotettavan estimaatin. Yksinkertaisin ja yleisesti käytetty ratkaisu ongelmaan on Laplacen menetelmä [Cestnik, 1990], jolla voidaan poistaa mahdollisuus nolla-arvoiseen todennäköisyyteen lisäämällä luokassa esiintyvien attribuuttien arvojen lukumäärään yksi ja luokkaan kuuluvien esimerkkien kokonaislukumäärään yksi jokaista mahdollista attribuutin arvoa kohden.

4.3. Luokitteluesimerkki

Mitchell [1997] havainnollistaa yksinkertaisen naiivin Bayesin algoritmin toimintaa esimerkillä, jossa luokitellaan päiviä niiden tenniksen peluuseen sopivuuden mukaan (taulukko 1).

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Taulukko 1. Opetusdata. [Mitchell, 1997]

Oletetaan, että uuden luokiteltavan esimerkin attribuutit saavat arvot {Sunny, Cool, High, Strong}. Luokkien todennäköisyydet lasketaan aineistosta arvojen frekvenssien perusteella:

$$\mathrm{P(Yes)} = 9/14$$
$$\mathrm{P(No)} = 5/14 \cdot$$

Attribuutin arvon luokkaan kuulumisen todennäköisyys on arvon frekvenssi luokassa:

$$\mathrm{P(Sunny|Yes)} = 2/9$$
$$\mathrm{P(Sunny|No)} = 3/5 \cdot$$

Yksinkertaisella naiivilla Bayesin algoritmilla lasketaan molempiin luokkiin kuulumisten todennäköisyydet kertomalla luokan todennäköisyys kaikkien attribuuttien arvojen luokkaan kuulumisen todennäköisyydellä

$$\mathrm{P(Yes)P(Sunny|Yes)P(Cool|Yes)P(High|Yes)P(Strong|Yes)} = 0{,}0053$$
$$\mathrm{P(No)P(Sunny|No)P(Cool|No)P(High|No)P(Strong|No)} = 0{,}0206 \cdot$$

Tästä huomaamme jälkimmäisen olevan todennäköisempi vaihtoehto, joten PlayTennis-luokka saa arvokseen No.

5. Algoritmin sovelluksia

Tässä luvussa esitellään joitakin naiivin Bayesin algoritmin sovelluksia. Luvun tarkoituksena on selventää naiivin Bayes-luokittelijan soveltuvuutta erilaisille aineistoille esimerkkien avulla.

5.1. Luonnollisen tekstin luokittelu

Yksi paljon käytetty tiedonlouhintatehtävä naiiville Bayesin algoritmille on luonnollisella kielellä esitettyjen tekstidokumenttien luokittelu. Kun tekstidokumentteja luokitellaan tiedonlouhinnan menetelmin, nousee esiin muutamia huomioonotettavia seikkoja, jotka ovat ominaisia erityisesti tälle tiedonlouhintatehtävälle. Mitchell [1997] mainitsee oleellisimpina pohdinnan kohteina kuinka mielivaltainen tekstidokumentti voidaan esittää esimerkkinä attribuutteineen ja kuinka tarvittavat todennäköisyydet tulisi estimoida. Mitchell esittelee yhtenä tekstidokumentin esittämistapana yksinkertaisen ratkaisun,

jossa jokaisen sanan järjestysluku tekstissä on attribuutin nimi ja attribuutti saa arvokseen sanan. Näin attribuuttien lukumäärä vaihtelee dokumentin pituuden mukaan. Todennäköisyyksien laskemiseksi Mitchell esittelee tavan, jossa attribuuteilla eli sanan sijainnilla dokumentissa ei ole merkitystä tekstin relevanttiuden kannalta, jolloin laskettavien todennäköisyyksien määrä laskee huomattavasti.

Naiivin Bayesin algoritmin riippumattomuusoletus Mitchellin kuvaileman tekstidokumentin esittämistavan tilanteessa tarkoittaa, että sanan todennäköisyys tietyllä paikalla ei ole riippuvainen muista sanoista muilla paikoilla. Tämä ei pidä paikkaansa etenkään englannin kielessä, jossa sanaparit ovat yhdyssanoja yleisempiä. Domingos ja Pazzani [1996] huomauttavat kuitenkin algoritmin luokittelutuloksen olevan hyvä huolimatta riippumattomuusoletuksesta. Mitchell [1997] toteaakin naiiviin Bayes-luokittelijaan perustuvan algoritmin olevan luonnollisella kielellä kirjoitettujen tekstidokumenttien luokittelualgoritmeista tehokkaimpien joukossa.

Altheneyan ja Menai [2014] hyödynsivät naiivia Bayesin algoritmia teoksen kirjoittajan tunnistamiseksi teoksen ominaisuuksien perusteella. Kirjat kerättiin arabiankieliseltä Alwaraq-sivustolta. Kirjailijoita valittiin kymmenen, joilta jokaiselta tarkasteltiin kolmea eri kirjaa. Viisivaiheisessa tutkimuksessa dokumenttien keräämisen ja esikäsittelyn jälkeen valitut tekstit esitettiin yli neljänsadan attribuutin muodostamina vektoreina. Attribuuteista puolet oli teksteissä merkittäviä sanoja. Näistä sanoista valittiin naiivin Bayesin algoritmin hyödyntämistä varten ne, joilla oli suurin frekvenssi kunkin kirjoittajan tuotannossa. Saadun attribuuttijoukon perusteella voitiin luokitella testidatan tekstit tietyn kirjoittajan tuotannoksi. Tutkimuksen viimeinen vaihe oli luokittelun evaluointi.

Naiivi Bayesin algoritmi luokitteli Altheneyanin ja Menain tutkimuksessa oikein keskimäärin 82 % esimerkeistä. Algoritmi osasi kuitenkin luokitella viiden kirjoittajan teokset oikein yli 90 % tarkkuudella. Kahden kirjailijan teokset naiivi Bayesin algoritmi luokitteli oikein alle 70 % todennäköisyydellä. Naiivin Bayesin laajennetut muodot suoriutuivat luokittelusta alkuperäistä paremmin. Altheneyan ja Menai mainitsevat myös huomanneensa sanojen muodon normalisoinnin vaikuttavan tuloksiin vain vähän ja luokittelun pelkällä sanan vartalolla jopa heikentävän tuloksia, sillä päätteet kertovat kirjoittajasta enemmän kuin sanan vartalo.

5.2. Sovellus lääketieteessä

Koska nykyaikaisissa sairaaloissa kerätään huomattavasti aikaisempaa enemmän tietoa potilaista, perinteiset analyysimenetelmät ovat muuttuneet tehottomiksi. Aineistoa voidaan kuitenkin hyödyntää lääketieteen kehittämisessä tiedonlouhinnan avulla. Tiedonlouhintatehtävä voi olla aineiston kuvailu tai ennustaminen. Ennustavassa luokittelussa tavoitteena voi olla esimerkiksi oikean diagnoosin, ennusteen tai hoitosuunnitelman valinta.

Erityisesti potilastietojen ja seurantadatan määrä on kasvanut [Lavrač and Zupan, 2005]. Potilastiedoille on ominaista, että datassa on paljon puuttuvia tai virheellisiä arvoja. Seurantadata voi olla peräisin pitkältä yhtenäiseltä ajanjaksolta, kuten onnettomuustapauksissa tai monilta perättäisiltä yksittäisiltä ajanhetkiltä, kuten sairauksien kontrollikäynneiltä.

Lavračin ja Zupanin mukaan naiivi Bayesin algoritmi on todettu lääketieteessä sairauksien diagnosoinnissa erityisen hyödylliseksi sopivien testien valinnassa sairauden todentamiseksi ja oikean hypoteesin varmistamisessa, mutta algoritmi auttaa lähes kaikissa diagnosointiprosessin päätöksentekoa vaativissa vaiheissa.

Zelic ja muut [1997] sovelsivat naiivia Bayesin algoritmia urheiluvammojen diagnosointiin. Opetus- ja testidata muodostettiin Ljubljanan yliopistollisen sairaalan urheilulääketieteen osaston 118 potilaan tietokannasta. Potilasta kuvasi yhteensä 49 eri attribuuttia, joista asiantuntijoiden mukaan tärkeimmät olivat vamman sijainti, pakotetun liikkeen ja Lachmannin testien tulokset. Vammat olivat luokiteltu 30 diagnoosiryhmään. Koska yhdestä luokasta oli saatavilla vain muutama esimerkki, opetusdatan sijaan käytettiin asiantuntijan määrittelemiä diagnosointisääntöjä.

Kun jatkuvien attribuuttien diskretointi tehtiin luokkaperustaisesti, Zelic ja muut pääsivät noin 69 % luokittelutarkkuuteen. Tutkijat olivat yllättyneitä tulokset tarkkuudesta ottaen huomioon datan rajatun määrän. Asiantuntijalääkärin lausuntojen perusteella naiivin Bayesin algoritmin diagnosointitarkkuus on tarpeeksi tarkka, toisin kuin päätöspuiden tuottama luokittelu.

5.3. Muita sovellusalueita

Naiivia Bayesin algoritmia on edellisten lisäksi sovellettu lukuisilla muillakin alueilla. Yhtenä esimerkkinä algoritmin mahdollisista sovellusalueista seuraa työssä suoriutumiseen liittyvä tutkimus. Valle ja muut [2012] tutkivat telemarkkinoijien työssä suoriutumista naiivin Bayes-luokittelijan avulla. Luokittelija jakoi työntekijät tutkimuksessa käytettyjen attribuuttien arvojen perusteella minimituottavuuden arvon ylittävään ja alittavaan luokkaan. Tutkimuksessa kokeiltiin luokittelua sekä työntekijää että työsuoritusta kuvaavien attribuuttien perusteella. Työntekijää kuvaavat attribuutit olivat työntekijän ikäryhmä, sukupuoli, siviilisääty, koulutustaso, asuinpaikan sosioekonominen arvostus numeroasteikolla 1–5 ja myyntikokemus (kyllä/ei). Työsuoritusta kuvaavat attribuutit olivat työntekijän työaseman aktiivisuustunnit yhteensä, kuluttajille puhutut tunnit, kokonaiskontaktien määrä ja loppuun asti vietyjen kontaktien määrä. Attribuutit diskretoitiin ennen naiivin Bayesin algoritmin soveltamista. Tutkimus tehtiin Chilessä ja siinä tarkasteltiin 660 työntekijää, jotka rekrytoitiin vuoden 2009 maalis- ja joulukuun välisenä aikana.

Luokittelun työntekijää kuvaavien attribuuttien perusteella voitiin katsoa epäonnistuneen, sillä 98 % esimerkeistä luokiteltiin virheellisesti minimituottavuuden arvon ylittäneiksi. Työsuoritusta kuvaavat attribuutit ennustivat minimituottavuuden ylittämisen tai alittamisen paremmin, sillä noin 81 % esimerkeistä luokiteltiin oikein. Johtopäätöksinä Valle ja muut päättelivät, että mahdollisesti puhelinmarkkinointityön rutiininomaisesta luonteesta johtuen työntekijän henkilökohtaiset ominaisuudet eivät vaikuta työssä suoriutumiseen. Sen sijaan työtä kuvaavat attribuutit auttoivat ennustamaan tulevien kuukausien työsuorituksia.

6. Testiajo

Tässä luvussa esitellään naiivin Bayesin algoritmin toimintaa pienen testiajon avulla.

6.1. Aineisto

Kokeessa käyttämäni aineisto on Mushroom-aineisto Bachen ja Lichmanin [2013] ylläpitämältä UCI-sivustolta. Aineistossa on 8124 esimerkkiä ja 22 attribuuttia. Esimerkkeinä käytetään *Agaricus*- ja *Lepiota*-sukujen sieniä.

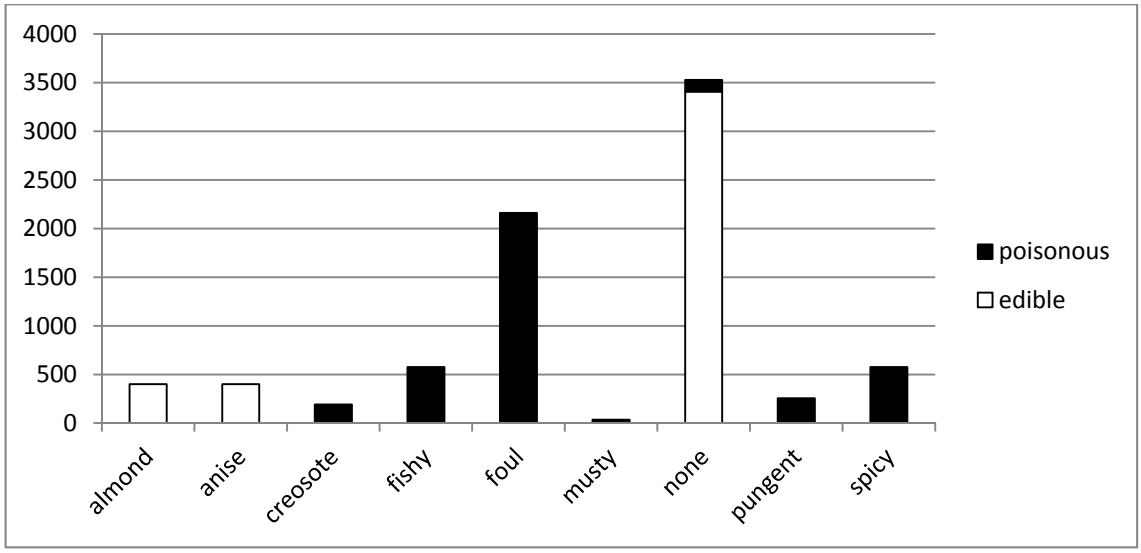
```
Attribute Information: (classes: edible=e, poisonous=p)
     1. cap-shape:                bell=b,conical=c,convex=x,flat=f,
                                  knobbed=k,sunken=s
     2. cap-surface:              fibrous=f,grooves=g,scaly=y,smooth=s
     3. cap-color:                brown=n,buff=b,cinnamon=c,gray=g,green=r,
                                  pink=p,purple=u,red=e,white=w,yellow=y
     4. bruises?:                 bruises=t,no=f
     5. odor:                     almond=a,anise=l,creosote=c,fishy=y,foul=f,
                                  musty=m,none=n,pungent=p,spicy=s
     6. gill-attachment:          attached=a,descending=d,free=f,notched=n
     7. gill-spacing:             close=c,crowded=w,distant=d
     8. gill-size:                broad=b,narrow=n
     9. gill-color:               black=k,brown=n,buff=b,chocolate=h,gray=g,
                                  green=r,orange=o,pink=p,purple=u,red=e,
                                  white=w,yellow=y
    10. stalk-shape:              enlarging=e,tapering=t
    11. stalk-root:               bulbous=b,club=c,cup=u,equal=e,
                                  rhizomorphs=z,rooted=r,missing=?
    12. stalk-surface-above-ring: fibrous=f,scaly=y,silky=k,smooth=s
    13. stalk-surface-below-ring: fibrous=f,scaly=y,silky=k,smooth=s
    14. stalk-color-above-ring:   brown=n,buff=b,cinnamon=c,gray=g,orange=o,
                                  pink=p,red=e,white=w,yellow=y
    15. stalk-color-below-ring:   brown=n,buff=b,cinnamon=c,gray=g,orange=o,
                                  pink=p,red=e,white=w,yellow=y
    16. veil-type:                partial=p,universal=u
    17. veil-color:               brown=n,orange=o,white=w,yellow=y
    18. ring-number:              none=n,one=o,two=t
    19. ring-type:                cobwebby=c,evanescent=e,flaring=f,large=l,
                                  none=n,pendant=p,sheathing=s,zone=z
    20. spore-print-color:        black=k,brown=n,buff=b,chocolate=h,green=r,
                                  orange=o,purple=u,white=w,yellow=y
    21. population:               abundant=a,clustered=c,numerous=n,
                                  scattered=s,several=v,solitary=y
    22. habitat:                  grasses=g,leaves=l,meadows=m,paths=p,
                                  urban=u,waste=w,woods=d
```

Kuva 1. Mushroom-aineiston attribuutit ja niiden arvot. Ote agaricus-lepiota.names-tiedostosta. [Bache and Lichman, 2013]

Attribuutit ovat kategorisia ja ne kuvaavat sienten ominaisuuksia, kuten elinympäristöä, ominaishajua tai populaation levinneisyyttä ja eri osien ulkonäköä tai rakennetta (kuva

1). Tavoitteena on selvittää sienen ominaisuuksien perusteella, kuuluuko se syötävien vai myrkyllisten sienten luokkaan.

Attribuuttikohtaisesta visualisoinnista voidaan huomata osan attribuuteista jakautuvan arvojensa perusteella melko tarkasti syötävien ja myrkyllisten sienten luokkiin. Toisaalta osa attribuuteista ei silmämääräisen tarkastelun perusteella näytä tuovan juurikaan lisäinformaatiota luokitteluun. Haju (odor) on hyvä esimerkki attribuutista, jonka arvot noudattavat luokkajakoa varsin hyvin (kuva 2).



Kuva 2. Haju-attribuutin arvojen jakauma. Syötävät sienet on merkitty valkoisella ja myrkylliset mustalla.

6.2. Tulokset

Luokittelu yksinkertaisella naiivilla Bayesilla algoritmilla toteutettiin Weka Explorer 3.6 -tiedonlouhintatyökalulla [Hall *et al.,* 2009]. Opetusmenetelmänä oli 10-kertainen ristiinvalidointi. Kokeilu onnistui varsin hyvin, sillä noin 96 % esimerkeistä luokiteltiin oikein (taulukko 2).

Todellinen luokka	Ennustettu luokka		Yhteensä
	Myrkyllinen	Syötävä	
Myrkyllinen	3609	307	3916
Syötävä	32	4176	4208
Yhteensä	3641	4483	8124

Taulukko 2. Ristiinluokitusmatriisi

Taulukosta 2 selviää, että myrkylliset sienet luokiteltiin syötäviksi kymmenen kertaa todennäköisemmin kuin syötävät myrkyllisiksi. Myrkyllisistä sienistä syötäviksi luokitelluiden osuudeksi tuli 8 %, kun syötävistä sienistä myrkyllisiksi luokiteltiin vajaa 1 %.

Naiivin Bayesin algoritmin edistyneempien versioiden hyödyntäminen voisi kuitenkin tuottaa jopa parempia tuloksia erityisesti myrkyllisten sienten väärinluokitusluvun pienentämiseksi. Algoritmin toiminnan tehostaminen onnistuisi mahdollisesti myös karsimalla attribuuttien määrää esimerkiksi informaation kasvun suhteen perusteella. Tässä kokeessa käytetty yksinkertaisen naiivi Bayesin algoritmi onnistui luokittelussa kuitenkin jo varsin hyvin.

7. Yhteenveto

Naiivin Bayesin algoritmin riippumattomuusoletus on useimmiten kaukana totuudesta ja algoritmin voisi näin ollen ajatella tuottavan varsin epätäydellisen luokittelun. On kuitenkin huomattu, että naiivi Bayesin algoritmi tuottaa jopa yhtä hyviä luokittelutuloksia kuin monimutkaisemmatkin luokittelualgoritmit [Mitchell, 1997]. Lisäksi riippumattomuusoletuksen vuoksi naiivin Bayesin algoritmin muistin käyttö pysyy hallittavan kokoisena myös suuridimensioisessa datassa. Sopivalla aineiston esikäsittelyllä algoritmin ongelmat ovat usein vältettävissä. Naiivia Bayesin algoritmia voikin näin ollen soveltaa useisiin erilaisiin luokitteluongelmiin.

Naiivia Bayesin algoritmia onkin käytetty lukuisilla eri aloilla, usein menestyksekkäästi. Hyviä tuloksia on saavutettu esimerkiksi luonnollisella kielellä kirjoitettujen tekstidokumenttien luokittelemisessa tietyn kirjoittajan tuotannoksi [Altheneyan and Menai, 2014], urheiluvammojen diagnosoinnissa testitulosten avulla [Zelic *et al.*, 1997] ja telemarkkinointiyrityksen työntekijöiden myyntitulosten ennustamisessa ensimmäisten kuukausien työsuoritusta kuvaavien attribuuttien perusteella [Valle *et al.*, 2012]. Tässä tutkielmassa naiivia Bayesin algoritmia sovellettiin Mushroom-aineistoon [Bache and Lichman, 2013]. Algoritmi tuotti varsin hyvän mallin myrkyllisten ja syötävien sienten erottamiseksi toisistaan.

Viiteluettelo

[Altheneyan and Menai, 2014] Alaa Saleh Altheneyan and Mohamed El Bachir Menai, Naïve Bayes classifiers for authorship attribution of Arabic texts. *Journal of King Saud University - Computer and Information Sciences* **26** (2014), 473–484.

[Bache and Lichman, 2013] Kevin Bache and Moshe Lichman, UCI Machine Learning Repository, 2013. Available as http://archive.ics.uci.edu/ml.

[Catlett, 1991] Jason Catlett, On changing continuous attributes into ordered discrete attributes. In: *Proc. of the EWSL-91*, 164–177.

[Cestnik, 1990] Bojan Cestnik, Estimating probabilities: A Crucial task in machine learning. In: *Proc. of the ECAI-90*, 147–150.

[Domingos and Pazzani, 1996] Pedro Domingos and Michael Pazzani, Beyond independence: Conditions for the optimality of the simple Bayesian classifier. In: *Proc. of the 13th International Conference on Machine Learning,* 105–112.

[Domingos and Pazzani, 1997] Pedro Domingos and Michael Pazzani, On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning* ***29*** (1997), 103–130.

[Hall *et al.*, 2009] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann and Ian H. Witten, The WEKA Data Mining Software: An Update. *SIGKDD Explorations* **11**, 1 (2009), 10–18.

[Hand *et al.*, 2001] David Hand, Heikki Mannila and Padhraic Smyth, *Principles of Data Mining.* The MIT Press, 2001.

[Kerber, 1992] Randy Kerber, ChiMerge: Discretization of Numeric Attributes. In: *Proc. of the AAAI-92,* 123–128.

[Kovahi and Provost, 1998] Ron Kovahi and Foster Provost, Glossary of terms. *Machine Learning* **30** (1998), 271–274.

[Lavrač and Zupan, 2005] Nada Lavrač and Blaz Zupan, Data mining in medicine. In: Oded Maimon and Lior Rokach (eds.), *Data Mining and Knowledge Discovery Handbook.* Springer, 2005, 1111–1136.

[Mitchell, 1997] Tom M. Mitchell, *Machine Learning*. McGraw-Hill, 1997.

[Richeldi and Rossotto, 1995] Marco Richeldi and Mauro Rossotto, Class-driven statistical discretization of continuous attributes. In: Nada Lavrač and Stefan Wrobel (eds.), *Machine Learning: ECML-95.* Springer, 1995, 335–338.

[Valle *et al.*, 2012] Mauricio A. Valle, Samuel Varas and Gonzalo A. Ruz, Job performance prediction in a call center using a naïve Bayes classifier. *Expert Systems with Applications* ***39***, 2012, 9939–9945.

[Witten and Franke, 2000] Ian H. Witten and Eibe Franke, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann Publishers, 2000.

[Zelic *et al.*, 1997] Igor Zelic, Igor Kononenko, Nada Lavrač and Vanja Vuga, Induction of decision trees and Bayesian classification applied to diagnosis of sport injuries. *Journal of Medical Systems* ***21***, 1997, 429–444.

Bipartite graph crossing minimization with self-organizing maps

Ville-Veikko Valtonen

Abstract.
This study examines the viability of Kohonen self-organizing map for solving the bipartite graph crossing minimization problem. A solution for presenting bipartite graph data to a self-organizing map and its high-level implementation is suggested. The performance of the implementation is assessed with generated graph data based on parameters from real life examples. Statistical results are presented to compare the proposed solution to the widely used barycenter algorithm.

Keywords: Kohonen self-organizing map, bipartite graphs, crossing minimization.

1. Introduction

Kohonen self-organizing map is an artificial neural network developed by Teuvo Kohonen (Kohonen, 2001). The map can be used to visualize or solve problems by mapping high-dimensional data to a low, often two dimensional grid. The specific problems solved with a self-organizing map usually involve a relationship derived from the different properties of the data, that is otherwise difficult or very time consuming to determine. The self-organizing map solves these problems by using a set of high-dimensional learning data to create a low-dimensional map. Vectors from the learning data are placed on the map by finding their best matching vector on the map. Additional data can be examined by finding their best matching vector and comparing it to the surrounding vectors of data on the map.

A graph is a set of nodes with edges connecting various pairs of nodes together (Harary, 1994). A graph is bipartite if the nodes can be divided into two groups where no two nodes within a group share an edge. If these two groups are then aligned onto parallel lines, we can count the number of crossings between edges of the two node groups. Minimizing the number of edge crossings of bipartite graphs leads to, for example, efficient layouts for circuits. The bipartite graph crossing minimization problem is a sorting problem that aims to minimize this overlap between edges of nodes. (Stallman et al., 2001)

This work focuses on examining whether or to what extent the bipartite graph crossing minimization problem can be solved with the Kohonen self-organizing map. The core idea is to present each node as a vector, and each edge as a component within that vector. Ideally, the map would be able to ar-

range two groups of these vectors on a two-dimensional map, where the vectors within each group are ordered so that the number of crossings become minimal.

2. Kohonen self-organizing map

Using self-organizing maps today is one of the most common methods within the domain of unsupervised learning in neural networks. The Kohonen self-organizing map is often referred to as KSOM whilst a general self-organizing map is simply a SOM. All self-organizing maps produce a low-dimensional structure from a high-dimensional set of data, which is then either computationally or visually examined to extract information about the different topological relationships between the vectors on the map. Often the goal of using a self-organizing map is to find relationships that derive from a large set of features otherwise not easily seen or represented. These can then be translated to more practical uses such as speech recognition. (Kohonen, 2001)

Self-organizing maps vary in their properties and uses, but all of them consist of the same basic components used in different stages of the creation and training of the map. The basis of a self-organizing map is a topological structure which forms both the starting point and the end result of the SOM process. The basic Kohonen map is a square grid, but there are also maps that use hexagonal grids, toroidal grids and different shapes containing these features internally. (Mount, 2011)

Perhaps the most important step of creating a successful self-organizing map for a particular purpose is the selection and representation of features from the data to be examined. (Laine, 2002) Whilst a single vector in a set of data may contain a vast amount of features, only certain features need to be selected in order to derive the relationship specific to this combination of features. A small amount of unrelated features or a lack of some features may not have a large impact on the final map. However, the more specific the set of features is to the relationship the self-organizing map is to represent, the better the positioning of the vectors on the final map (Kohonen, 2001). Visual examination of the results of different combinations of features often help in determining the correct set from the data.

Once the shape and internal structure of the map has been decided and a set of vectors with the interesting features is available, the map may be trained to produce the final result. The learning stage is often modified to better suit the finding of the desired results (Kohonen, 2001). The main process of the training algorithm consists of choosing a random vector from the training data, finding the position of the map closest to the chosen vector, determining a neighbour-

hood around the position and making the points within the neighbourhood more similar to the chosen vector. The size of the neighbourhood and the amount of change to points within it depend on the number of iterations the map has already been trained with. The shape of the neighbourhood and the function of a vector's similarity towards a single point on the map are fixed.

The first function required in the training process is the best matching unit. In Kohonen's original implementation this function is euclidean distance between two vectors, the current learning vector and a vector on the map. (Kohonen, 2001). This type of comparison requires the training set to be normalized with respect to different weights to be given to each feature. Other BMU functions include Manhattan distance and functions where the lack of matching values is given less weight than matching values, for example, whether two vectors have the same value representing gender. A particular BMU function has a large impact on the resulting map and it's incorrect use can skew the result.

The neighbourhood or distance function for training of a self-organizing map determines the radius of change for a single iteration of training. In most cases euclidean distance is used for the neighbourhood function, with its distance spanning the entire map on the first iteration. The radius of the neighbourhood function decreases with iteration count, commonly using gaussian or linear function for rate of decrease. The purpose of the neighbourhood function is for its radius to initially reach far enough for the map to form a rough order within it, and decrease the radius during further iterations to produce statistical accuracy (Kohonen, 2001). The neighbourhood function may also span separate maps or regions in a map that have different training sets or features (Pampalk, 2003).

The learning function of a self-organizing map determines the extent to which a single vector in the chosen neighbourhood is made similar to the vector chosen. In the beginning of the training each vector is modified only little, whilst towards the end of the training vectors may be made nearly identical to the chosen vector (Kohonen, 2001). The amount of training usually decreases using the gaussian function or a linear function, and its impact on all features are calculated similarly. However, in some cases specific features may be trained differently, such as binary features. The neighbourhood function and learning function together form the rate and radius of a single iteration of training. The training process of self-organizing map always progresses towards faster learning and smaller neighbourhood as opposed to a similar method called simulated annealing, where the learning process may be again quickened if a

desired level is not reached in the resulting map (Kohonen, 2001; Srivastava and Sharma, 2008)

Once a self-organizing map has gone through a sufficient number of training iterations with a set of vectors, all of the vectors are placed on the map using the same BMU function as in the training. The map may be colored and lines drawn between known features of vectors to enhance visual examination of the map. Labels are also often drawn for each vector placed on the map. Computational interpretation of the map may be finding lines or curves that span a set of vectors, calculating average distances or clusters, or choosing new sets of data for further examination (Kohonen, 2001). Results for visual use are usually two dimensional whilst computational maps may be in any number of dimensions not easily visualized.

3. Bipartite graph crossing minimization problem

This chapter defines bipartite graphs and the concept of crossings as well as discusses two well-known algorithms for solving the bipartite graph crossing minimization problem; Barycenter and median.

A graph may be defined as a tuple G = (V,E), where V is the set of vertices in the graph and E the set of edges connecting pairs of the vertices together. Vertices are sometimes referred to as nodes or points and they may have other information connected to them such as position in space. Edges are the connections between vertices, often drawn as lines between points in visual representations of graphs.

Graphs are either directed or undirected and their edges are weighted or unweighted. In a directed graph each edge is given a direction in which it can be travelled. An example of such a graph would be a road network represented as a graph, where roads have a direction in which they can be driven. Undirected graphs allow each edge to be travelled in both directions. Weighted graphs have a relative value, a weight for each of their edges. The weight of an edge may have many conceptual meanings, for example, importance or length, and they may have an impact in some graph drawing problems. In this study all graphs are undirected and unweighted for the purposes of simplicity, even though some algorithms such as the barycenter method can be modified to process weighted graphs. (M. Forster, 2002)

Bipartite graphs, also known as bigraphs, are such graphs G = (V,E), that V may be divided into two distinct sets L and R such that no edge in E connects two vertices belonging to the same set L or R. Based on this definition we may refer to bipartite graphs as a triple G = (L,R,E). In other words, in a bipartite we may form two groups of nodes where the nodes have no connections with each

other in the same group. Arranging these groups parallel to each other enables the counting of crossings between edges that span the two groups. It is obvious that the number of edge crossings depends on the order of the nodes on the parallel lines. Hence, minimizing the number of edge crossings is solved by finding the correct order of the nodes on the lines.

A drawing of a bipartite graph is an ordered pair of functions (h,r) where each function gives the order of a given vertex in the drawing when the vertices are divided and drawn onto two parallel lines. A crossing is a set of two edges {(u,y),(v,x)}, where u,v belong to L and x,y belong to R such that h(u)<h(v) and r(y)>r(x) or h(u)>h(v) and r(y)<r(x). (Figure 1) A drawing is said to be optimal when there exists no other drawing with fewer crossings (Martí and Laguna, 2004).

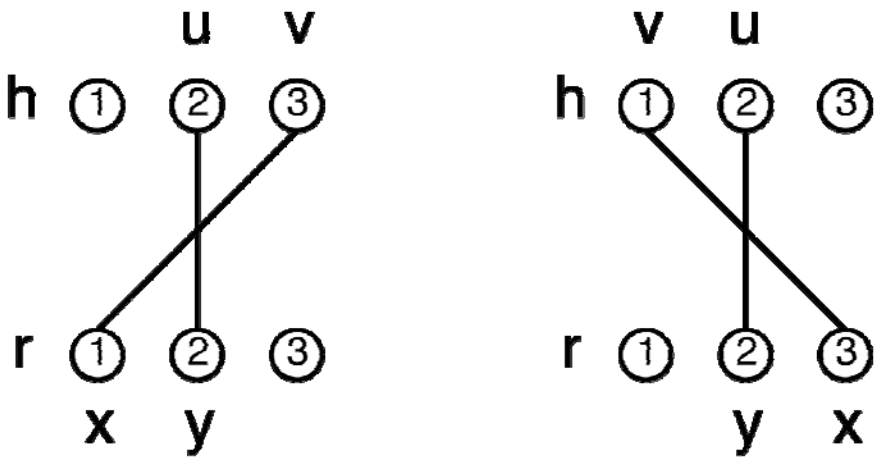


Figure 1. The number of edge crossings depends on the order of nodes on the parallel lines.

The bipartite graph crossing minimization problem is known to be NP-complete (Johnson, 1982). As such, there currently exists no solution to the problem in polynomial time and all current methods are heuristics often finding locally optimal solutions. A simple solution that iterates every possible ordering may be created. However, the computational time increases very fast when the number of nodes in the graph increases.

The most popular heuristics for solving the bipartite graph crossing minimization problem all employ a similar approach. In essence the two layers are processed with a simple set of rules that reorganize the points in one layer at a time. The process continues until two successive runs, one for each layer, is complete with no further changes to the orderings of points in a layer. The most popular heuristics are the barycenter and median heuristics. (Martí and Laguna, 2004).

The barycenter heuristic processes each node in a layer one at a time, and calculates the arithmetic mean from the positions of its adjacent nodes. Once all nodes in a layer have been processed, they are ordered according to these arithmetic means. The process repeats on the two layers until both of the layers remain unchanged after two successive iterations. The median heuristic is very similar. However, it calculates the mean of a particular set of adjacent nodes instead of the arithmetic mean. There exist variations of the barycenter and median heuristics for example the barycenter 2 with tie-breaking mechanism (Gansner, 1988) and semimedian heuristic (Mäkinen, 1990).

4. Testing environment

Theoretical proof or mathematical results identifying the benefits of the self-organizing map for a specific problem are not trivial to produce. While bipartite graphs are relatively simple structures to represent for theoretical hypotheses, practical examination of different approaches for the use of self-organizing maps for them was seen most beneficial at the current state of research. There exist many variations of self-organizing maps and their use for specific problems. Therefore, practical tests for their performance not only yield results on the general workability of the problem with SOM's, but also the type of parameters and functions to be used within the SOM itself.

The first problem to be solved before the construction and design of the testing application was the representation of the problem to the map itself. It is clear that a SOM organizes the vectors it is represented with onto the map, and since the bipartite graph crossing minimization problem is a problem organizing either the nodes or the edges between them, these two were the first candidates to be considered as vectors for the map. Examining the properties of edge and node as vectors, the nodes were chosen for this particular set of examinations. The reason for this decision was the ability of a set of nodes to represent both the relative span of a single edge and position the endpoints of an edge separately on the map. Edges as vectors would have caused problems that require further sorting and testing of separate cases, for example, where two edges are near each other they may either be parallel or cross each other.

In order to represent nodes as vectors for the self-organizing map, the information contained in each node had to be presented as a numerical value within the vector. Observing the properties of a node on a bipartite graph, we enumerate three properties: line or group in which the node belongs to, its order within the group and the connections it has with other nodes.

In Figure 2 there are four nodes. Node A belongs to group 1, has order 2 and a connection to B and D. Node B belongs to group 2, has order 2 and a con-

nection to A and E. Handling both nodes separately we could represent them both with vectors containing 3 values. However, each vector must be comparable to another vector, or a current state of a randomly generated or modified vector on the map. Therefore the value for absence of a particular connection to a node is also required. In this case A would require information for the absence of an edge to F and B the absence of an edge to C. Furthermore, as each vector has a coordinate in the map and it is the variable that the SOM modifies, it will be used to represent the order of a vector. Thus the final representation of a single node for these tests is: a value for the group of a node and a value for every edge in the graph, 1 if the node belongs to this edge and 0 if it does not.

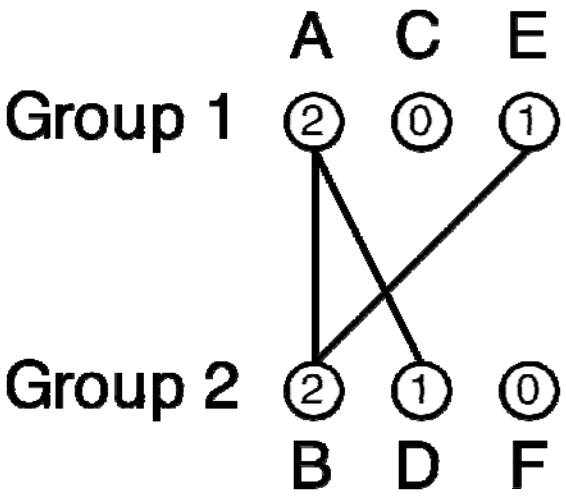


Figure 2. Illustration for node and edge representation.

With the vector representation chosen the basic structure and behaviour of a SOM was implemented in Java with gaussian functions for learning and neighbourhood size and euclidean distance for the BMU. A visualizer was attached to the SOM training process to examine the initial results of the SOM, including some known problems that a SOM is viable for solving. While the basic problems known to be solvable with a SOM all produced good results, the results with bipartite graph data were mixed. Clearly the map was able to cluster nodes with a lot of common edges together, but the results varied with the randomness of the map. In addition with larger graphs the results were more and more inconsistent.

For the second iteration of the Java application the different functions of a SOM were separated. Neighbourhood function, learning function and the BMU function were all made separate interfaces so that different kinds and varying values for the functions could be tested. Still, though with better results using

more moderate learning functions the map produced varying results. Further examining these results drawn as graphs they revealed to reduce the number of crossings in many cases. Most cases that did not reduce or even increased crossings were possible to be corrected by moving the left- or right-most pairs of nodes to the other side of the parallel line. Clearly; the ordering of the nodes was human determined at this point. To gather statistical evidence for the viability of the SOM to produce good results required a computational method of arranging the nodes from the resulting two-dimensional map.

A good method for ordering the output nodes produced by the SOM was not found, however. Different methods were tested both by attempting to use similar logic as in the ordering done by hand and by using some algorithms from internet sources. However, all of these produced either somewhat random orderings or orderings that discarded too much information. This was due to the fact that it was almost always impossible to draw a line on the 2D plane that would distinctly divide the two groups. One of the last tests attempted was another round of SOM which would reduce the 2D plane to a 1D plane. This produced better results than before but was much more resource intensive than the other methods. The final form of the SOM for this experiment was derived from this result; taking into account both features of the 2D map and the ability of a 1D map to produce a fit ordering.

Using just a 1-dimensional map as the first iteration of the SOM was too restrictive for nodes that shared very similar conditions and attempted to fill the same space. This problem was less apparent in the 2-dimensional map as each node had more space to settle in. To combine these benefits the 2-dimensional map was divided in half by modifying the learning function and the BMU function. Firstly, the learning function was modified not to alter the value representing the layer of a specific node. All the other values were altered as before. Second, the BMU function was altered to always return 0 for the relative similarity, if the value representing the layer of a node was not the same. This combined with an initialization function that set one half of the map's layer values 1 and the other 0 effectively divided the map in half.

These modifications were enough to run some tests. However, it quickly became apparent that a neighbourhood function that decreases radius on both axes at the same time was not adequate. Nodes became quickly isolated on their own sides and ceased to have an effect on the nodes of the other side, cutting off their influence. The neighbourhood function was then modified so that it's reach would always span the entire axis on which the plane was divided, and only shrink on the other axis. At this point it was clear that some of the positi-

onal gain from a 2D versus 1D map was lost. The neighbourhood function modification produced the last kind of map to be tested for this experiment.

5. Results

This chapter discusses various results produced by the self-organizing map implementation outlined in Chapter 4. Initial examination focuses on visual results, followed by simple statistical measures from a generated set of data and finally examination of a single outlier case in the data. Visual examination was conducted on small manually created graphs. Generated data was produced with a Java based random bipartite graph generation algorithm.

The random generation algorithm for bipartite graphs produced graphs with equal amount of nodes on each of the bipartite layers. The basic functionality of the algorithm was to randomly choose a node from one layer, alternating between the two layers, then choose another node that was already connected to the graph and form an edge between them. This was repeated until the graph was fully connected. Finally, a random number of edges was added between random pairs of nodes. Various sets of these random graphs were produced. However, only the 10-node and 100-node sets were used in the statistical tests. Various internet sources provided single bipartite graphs that were used as templates for the parameters of generated graphs, such as the Stanford Large Network Dataset Collection (Leskovec and Krevl, 2014)

Visual results were produced both during the development of the Java implementation and after it was finished to verify the correct functionality of the application as well as provide guidelines for parameters. Figure 3 presents the basic view for a single run of the SOM algorithm, which when run shows the change of the map live. The red lines and squares represent edges and nodes with their information presented in the upper left corner. The background color represents the normalized value of each vector on the map. As seen between images A and B, larger graphs become quickly more complicated and human capability to judge the quality of the end result decreases. However, the background proved very useful in verifying that the different functions used in the SOM process produced stable maps and observing how they evolved.

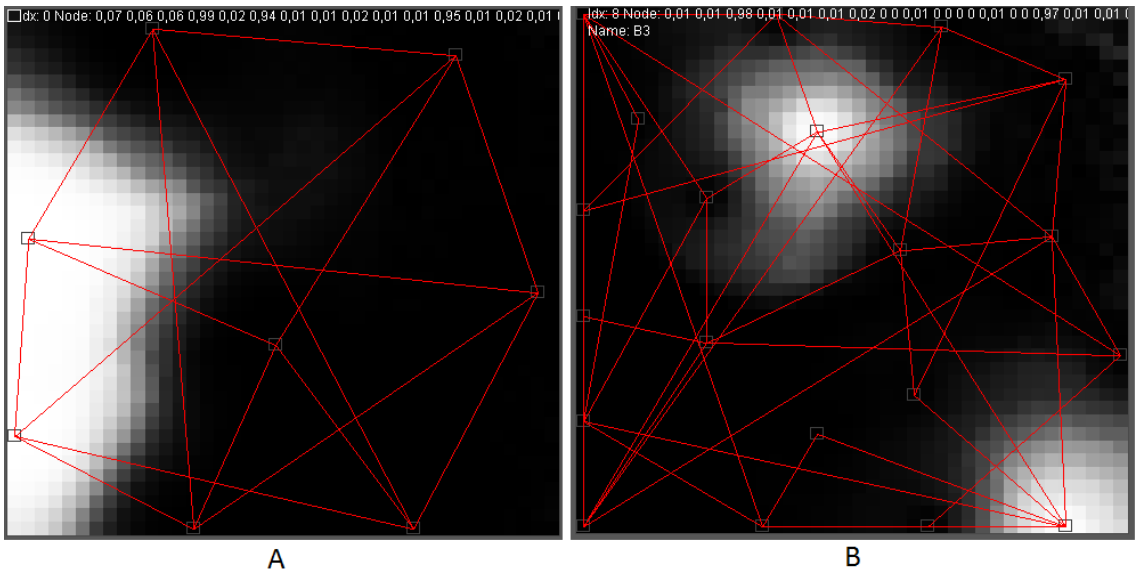


Figure 3. Live visualization of bipartite graphs being iterated through the SOM algorithm.

One of the first results observed visually was, that it makes no difference if a map is divided into two equally shaped parts where both the initialization of random vector values on the map and the neighbourhood function are the same for both sides. In this case, euclidean distance did not produce the same results but the vertical approach to neighbourhood function as proposed in Chapter 4 produced exactly the same results on both sides. This was observed by first testing a map of 40 by 40 vectors divided equally half and then testing a map of 20 by 40, the size of previous half, side by side with the same random seeds. The result was exactly the same where visually the 20 by 40 map was just the two previous maps overlaid on each other. This find led to a final modification of the algorithm to not split the map which slightly reduced the complexity of the algorithm.

The second visual result as seen in Figure 3 is that the current implementation of the SOM does attempt and succeed to certain lengths at minimizing the number of crossings between nodes. However, the minimization is performed on two axes while a bipartite graph crossing minimization is performed on a single axis. Most of the crossings that do not exist in the produced map become a problem when the nodes on the map must be projected onto their respective layers.

Statistical tests were planned on two generated sets of bipartite graphs, each with 1000 test cases: the first with 10 nodes on each layer and the second with 100 nodes on each layer. For the smaller graphs the amount of edges varied between 19 and 90 while the larger graphs had nodes between 199 and 5500. Parameters for the initial runs of the algorithm were three different neighbourhood functions: Euclidean, square and vertical. Learning rate values of 0.1 and

0.05 were also planned in combination with each of these neighbourhood functions. During first practical runs it was discovered that a single test case of the 100 node graphs took between 10 and 40 minutes, which was deemed too long for any practical use and the tests were focused on the 10 node graphs instead.

The few 100 node graphs that were ran through the tests all used Euclidean distance with 0.1 learning rate. The graphs varied between 500 000 and 3 000 000 crossings initially with an average improvement of 5,35% less crossings with SOM and 24,35% less crossings with barycenter.

All planned tests for the 10 node graphs were carried out successfully. Each test case took between 1 and 2 seconds with slower learning rates producing slightly longer runs despite having the same number of iterations in the algorithm. The averages between all six parameter configurations were 9.1% less crossings with SOM and 28.6% less crossings with barycenter. The barycenter algorithm does not include any random factors so the figure stands for all future tests with the 10 node graph set.

Studying the tests further the worst results were achieved with the square and Euclidean neighbourhood functions with 8% and 8.5% reductions respectively. Vertical neighbourhood function as hypothesized in Chapter 4 implementation produced the best results with 10.9% reduction in crossings. Between 0.1 and 0.05 as learning rate parameters, the 0.1 reduced the number of crossings 1.6% more. Taking the vertical neighbourhood function an additional test set with learning rates between 0.1 and 0.25 were tested. These produced a further improvement with the vertical neighbourhood function with 0.15 learning rate producing an average of 13.9% reduction in crossings.

Individual results showed some cases where the barycenter algorithm performed worse than the SOM. Observing the same graphs in different settings, the 0.15 learning rate with vertical neighbourhood function produced the most of such cases. However, only one case appeared consistently better with the SOM than barycenter. This case number 535 of the 10 node set was ran through an additional 100 runs of the SOM to verify if it was an outlier in the data caused by the random factor in the SOM. Using 0.1 learning rate and vertical neighbourhood function the SOM produced better results than barycenter in 83% of the cases with as few as 13 crossings whilst barycenter produced 40 from the original of 77.

The algorithm used to generate the data also produced a visual representation for every graph. Figure 4 presents the image for case 535. Observing the case superficially shows that four nodes with only one on the upper layer have just one edge. Most of the nodes have two edges and five nodes have more than that. Knowing that the graph is fully connected it could be suggested that there

exists a single straight path through most of the graph with a few additional edges. No further study for this case or this type of graph regarding the SOM or barycenter was carried out.

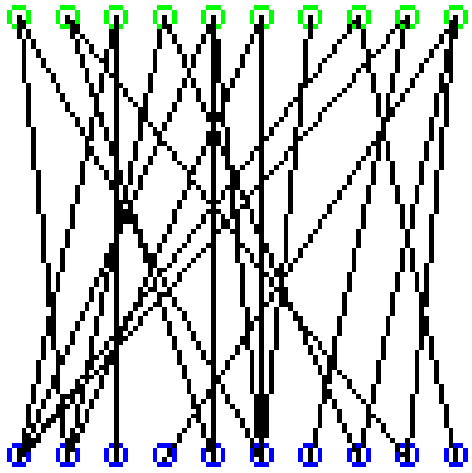


Figure 4. Graph generator output of 10-node case 535.

6. Summary

The results of this study do not generally encourage the self-organizing map as a viable method for solving the bipartite graph crossing minimization problem. Practical tests show that it takes considerable time to solve the problem using SOM with most notably the number of edges having the largest impact on the algorithm whilst barycenter and others scale with the amount of nodes. The amount of possible edges grows in factorial scale when the amount of nodes grows linearly. The Stanford Large Network Dataset Collection contains a single bipartite graph for the Wikipedia.com election data that could serve as an example of an actual graph that might need this type of solving. However, this graph of 7000 nodes is so large that no SOM run for it was completed even in several days' time (Leskovec and Krevl, 2014).

Furthermore, the barycenter performs considerably better in almost all cases tested. While the results do show that SOM can reduce the number of crossings in a bipartite graph with the given implementation, it reduces the amount far less than barycenter.

This study clarifies possibilities for further examination of the SOM with bipartite graphs and the barycenter algorithm. The case 535 could provide a good starting point for either examining whether it is a special case for the barycenter algorithm or for the SOM. Also the arrangement of nodes from a two dimensional map to two single dimensional parallel lines would be of great interest with regards to the performance of the current SOM implementation. Other avenues of research would be different ways of presenting bipartite graphs to the

SOM where either the BMU function would favor similarity more than dissimilarity or the map would scale with the amount of nodes instead of the amount of edges.

References

[Forster, 2002] Michael Forster, Applying crossing reduction strategies to layered compound graphs, In: *Proceedings of Graph Drawing* (2002), 276–284.

[Gansner et al, 1988] Gansner E.R., S.C. North and K.P. Vo, DAG - A program that draws directed graphs, *Software Practice and Experience* **18** (1988) 1047–1062.

[Harary, 1994] Frank Harary, *Graph Theory*. Perseus Books, 1994.

[Johnson, 1982] David S. Johnson, The NP-completeness column: an ongoing guide. *Journal of Algorithms* **3** (1982), 288–300.

[Kohonen, 2001] Teuvo Kohonen, *Self-Organizing Maps. 3rd Edition*. Springer, 2001.

[Laine, 2002] S. Laine, Selecting the variables that train a self-organizing map (SOM) which best separates predefined clusters. *Neural Information Processing* **4** (2002), 1961–1965.

[Leskovec and Krevl, 2014] Jure Leskovec and Andrej Krevl, Stanford Large Network Dataset Collection, http://snap.stanford.edu/data, 2014

[Mäkinen, 1990] Erkki Mäkinen, Experiments on drawing 2-level hierarchical graphs, *International Journal of Computer Mathematics* **36** (1990), 175–181.

[Martí and Laguna, 2004] Rafael Martí and Manuel Laguna, Heuristics and meta-heuristics for 2-layer straight line crossing minimization. *Disc. Appl. Math*. **100**, 3 (2003), 665–678.

[Mount, 2011] N. J. Mount, Self-organizing maps and boundary effects: quantifying the benefits of torus wrapping for mapping SOM trajectories, *Pattern Analysis & Applications* **14** (2011), 139–148.

[Srivastava and Sharma, 2008] Kamal Srivastava and Reeti Sharma, A hybrid simulated annealing algorithm for the bipartite crossing number minimization problem. In: *CEC* 2008, 2948–2954.

[Stallman et al., 2001] Matthias Stallmann, Franc Brglez and Debabrata Ghosh, Heuristics, experimental subjects, and treatment evaluation in bigraph crossing minimization. *JEA* **6** (2001), Article No. 8.