Kohosen itseorganisoituva kartta virtualisoitujen laskentaresurssien eksploratiivisessa data-analyysissa

Jaakko Routamaa

Tampereen yliopisto
Informaatiotieteiden yksikkö
Tietojenkäsittelyoppi
Pro gradu -tutkielma
Ohjaaja: Martti Juhola
Heinäkuu 2015

Tampereen yliopisto
Informaatiotieteiden yksikkö
Tietojenkäsittelyoppi
Jaakko Routamaa
Pro gradu -tutkielma, 49 sivua
Heinäkuu 2015

Tiivistelmä

Laskentaresurssien virtualisointi on helpottanut resurssien käyttöä kuvaavan datan keräämistä. Organisaatioiden privaattipilvien kustannukset ovat korkeat, mutta käytön tarkoituksenmukaisuuden arvioiminen on usein hankalaa. Tässä tutkielmassa pohditaan Nokia Networksin privaattipilven käyttöasteen seuraamiseen liittyviä ongelmia. Tarkastelussa analysoidaan käyttöä kuvaavaa dataa Kohosen itseoraginsoituvan kartan avulla. Kohteen tarkat piirteet ovat epäselviä, joten tämän tutkielman lähestymistapa on eksploratiivinen data-analyysi, johon itseorganisoituvan kartan havaitaan soveltuvan hyvin.

Avainsanat ja -sanonnat: eksploratiivinen data-analyysi, Kohosen itseorganisoituva kartta, neurolaskenta, ohjaamaton oppiminen, pilvipalvelut, tiedonlouhinta, tietämyksen muodostaminen, virtualisointi.

Sisällys

1. Johdanto

Nokia Networks omistaa paljon laskentaresursseja, joita hyödynnetään ohjelmistokehityksessä. Laskentaresursseja varataan testausympäristöjen eli laboratorioiden tarpeisiin. Laskentaresurssit maksavat paljon, joten laboratorioiden käyttötavan halutaan olevan tarkoituksenmukainen. Jos esimerkiksi laboratorion varaamia resursseja ei vapauteta tarpeen päätyttyä, niin resursseja haaskataan.

Laboratorioiden seuraamiseksi on tilattu analysointityökaluja, mutta niitä ei ole pystytty toteuttamaan. Merkittävä ongelma on se, että laboratorioille on monenlaisia käyttötapoja, jotka oletettavasti eroavat toisistaan merkittävästi.

Pilvilaskenta yhdistelee erilaisia laskentaresursseihin liittyviä tekniikoita ennennäkemättömän laajalla tavalla. Pilvipalveluiden ansiosta palveluntarjoajat ovat saavuttaneet suuria säästöjä, sillä esimerkiksi riippuvuus kalliista ihmistyövoimasta on pienentynyt merkittävästi. Usein suuri osa rutiininomaisista toimenpiteistä on automatisoitu, minkä seurauksena pienen henkilöstön on mahdollista ylläpitää laajaa pilveä.

Nokian laskentaresurssit sijaitsevat privaattipilvessä, mikä tarkoittaa vain Nokian tuotekehityksen tarpeisiin varattua pilveä. Pilvilaskennan tarjoamia etuja on hyödynnetty paljon Nokian sisällä, mutta virtualisoitujen laboratorioiden käyttöastetta ei vielä pystytä seuraamaan. Kolmannen osapuolen tuotteiden ansiosta Nokian privaattipilven toiminnasta mitataan erilaisia käyttöä kuvaavia tuloksia, joita ei kuitenkaan ole vielä hyödynnetty.

Tässä tutkielmassa perehdytään saatavilla olevaan suorituskykydataan. Suorituskykydatan käyttämiseen liittyy paljon epäselvyyksiä, joten ongelmaa lähestytään eksploratiivisen data-analyysin avulla.

Kohosen itseorganisoituva kartta on aivokuoren toimintaan perustuva neurolaskentamenetelmä. Itseorganisoituvaa karttaa on sovellettu jo yli 10 000 tieteellisessä tutkimuksessa. Itseorganisoituvan kartan on havaittu sopivan monien eri tieteenalojen ongelmiin. Kohosen [2013] mukaan itseorganisoituva kartta sopii erityisen hyvin sellaisiin ongelmakohteisiin, joista on mahdollista kerätä lähes rajattomasti dataa.

Itseorganisoituvan kartan uskotaan sopivan myös tämän tutkielman tarpeisiin monesta syystä. Esimerkiksi kohteesta muodostetut muuttujat eivät tyypillisesti noudata mitään tunnettua jakaumaa, mutta itseorganisoituva kartta ei aseta tiukkoja vaatimuksia datan jakaumien suhteen.

Tutkielman toisessa luvussa käsitellään neurolaskentaa yleisesti. Neurolaskennan havaitaan yhdistelevän tietokoneiden ja aivojen toimintamalleja tietokoneille epätyypillisten ongelmien ratkaisemiseksi.

Kolmannessa luvussa tarkastellaan päämenetelmää. Luvun perusteella selviää itseorganisoituvan kartan toimintaperiaate ja algoritmin yksityiskohdat. Luvussa käsitellään myös yleisiä hyviä periaatteita laadukkaan analyysin toteuttamiseksi.

Neljännessä luvussa käsitellään sovelluskohdetta. Virtualisointi ei ole uusi tekniikka, mutta se on tällä hetkellä erittäin tärkeä käsite, sillä monet yritykset ja yhteisöt ovat tehostaneet toimintotapojaan siirtymällä käyttämään pilvipalvelumalleja. Virtualisointi on yksi tärkeimmistä pilvipalvelujen taustalla olevista tekniikoista. Tutkielman kannalta on erityisen mielenkiintoista, kuinka paljon virtualisointi on helpottanut käyttöä kuvaavan datan keräämistä.

Viides luku käsittelee datan keräämistä ja esiprossointia. Datan piirteisiin tutustutaan alustavasti. Jo esiprosessoinnin perusteella tehdään mielenkiintoisia havaintoja sovelluskohteesta.

Kuudennessa luvussa toteutetaan Kohosen itseorganisoituvaan karttaan pohjautuva eksploratiivinen data-analyysi. Analyysissa käytetään esiprosessoitua suorituskykydataa. Luvun alussa etsitään sopivat itseorganisoituvan kartan parametrit, minkä jälkeen esitellään menetelmä laboratorioiden vastaavuuden havaitsemiseksi, jota myös sovelletaan kohteeseen.

2. Yleisesti neurolaskennasta

Tämän luvun alussa käsitellään aivotoiminnan mallintamisen vaikutusta neurolaskennan syntymiseen, minkä jälkeen esitetään tärkeitä neurolaskentaan liittyviä periaatteita. Ohjattua oppimista käsittelevässä kohdassa esitetään tunnettu *monikerroksinen Perceptron-neuroverkko* (engl. *multilayer Perceptron*, *MLP*) lyhyesti. Luvun lopussa käsitellään ohjaamatonta oppimista, minkä yhteydessä esitetään alustava esimerkki *Kohosen itseorganisoituvasta kartasta* (engl. *Self-organizing map, SOM*).

2.1. Aivojen perusrakenteista

Useiden neurolaskentamenetelmien kehittämisen yhteydessä aivojen toiminnan mallintamisella on ollut suuri merkitys. Näin ollen neurolaskennan syntymiseen vaikuttaneiden aivojen perusrakenteiden ja -toimintojen tarkastelu on perusteltua.

Aivosolut eli *neuronit* ovat aivojen keskeisiä yksiköitä, jotka muodostavat aivojen hermoverkkojen rakenteessa solmukohtia. *Synapsien* tehtävänä on yhdistää hermoverkkojen rakenteita muuttamalla neuronien välisiä sähköisiä signaaleja kemialliseen muotoon ja välittää ne *reseptoreille*, jotka välittävät signaalit jälleen sähköisinä eteenpäin. Jo aikaisen näkemyksen mukaan synapsien tehtävänä uskottiin olevan signaalien aktivoiminen tai estäminen [Haykin, 1994].

Vierekkäisiä neuroneja ei ole välttämättä yhdistetty suoraan toisiinsa, vaikka viereisten neuronien muodostamilla naapurustoilla todetaankin myöhemmin olevan erityinen merkitys. *Aksonit* välittävät signaaleja ulos neuroneista ja *dendriitit* ottavat niitä vastaan, mikäli vastassa on toinen neuroni. Aksoni voi ohjautua myös aivoista kehon lihaksistoon. Hermosoluilla on useita dendriittejä, mutta hermosolusta lähtee kuitenkin vain yksi aksoni.

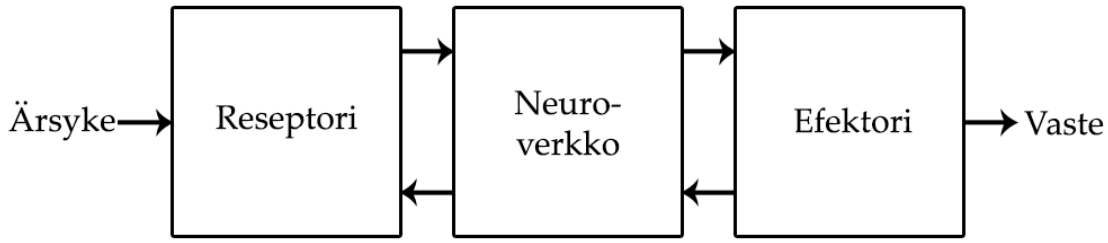
Suuri haaste aivojen toiminnan ymmärtämisessä on eri alitoimintojen yhteisvaikutus. Monista alitoiminnoista on toteutettu karkeita laskennallisia menetelmiä, mutta aivojen toimintaa kokonaisuutena ei ole vielä pystytty jäljittelemään kovinkaan hyvin.

Perusrakenteiden mallinnuksella ja yhteistoimintaa jäljittelemällä on kuitenkin saavutettu mielenkiintoisia keinotekoisia systeemejä, joita voidaan hyödyntää esimerkiksi tiedonlouhintaongelmien ratkaisemiseksi.

2.2. Aivojen toiminnan mallintaminen

Haykinin [1994] mukaan Arbib [1987] mallinsi aivot karkeasti kuvassa 1 esitetyllä tavalla kolmivaiheiseksi systeemiksi. Reseptorit tulkitsevat kehon saamia ärsykkeitä sähköisiksi signaaleiksi ja lähettävät niistä viestin aivojen neuroverkon käsiteltäväksi. Efektorit muuttavat neuroverkoilta saatuja signaaleja kehon

vasteiksi, kuten esimerkiksi käden heilautukseksi. Systeemin vaiheista välittyy palaute edelliseen vaiheeseen.



Kuva 1. Arbibin [1987] malli aivojen kolmivaiheisesta ärsykkeenkäsittelystä.

Havainnot aivojen ja tietokoneen toimintaperiaatteiden erilaisuudesta herättivät kiinnostuksen aivojen toiminnan keinotekoiseen mallintamiseen [Haykin, 1994]. *Keinotekoisilla neuroverkoilla* (engl. *artificial neural network*) tai lyhyesti *neuroverkoilla* tarkoitetaan vähintään etäisesti aivojen hermoverkkojen toimintaan perustuvia matemaattisia malleja. Tässä tutkielmassa keskitytään käsittelemään neuroverkkoja *neurolaskennan* näkökulmasta, joka soveltaa keinotekoisia neuroverkkoja ja tutkii niiden hyödyntämistä tiedonlouhinnan ja koneoppimisen ongelmissa. Jatkossa tässä tutkielmassa neuroverkolla tarkoitetaan neurolaskennassa hyödynnettävissä olevaa menetelmää, ellei erikseen toisin mainita.

Neurolaskennan on havaittu sopivan hyvin moniin tunnistustehtäviin, erityisesti niihin, joissa on tarpeen hyödyntää jotakin aivoille ominaiselta vaikuttavaa ratkaisumallia. Biologisesta näkökulmasta neurolaskentamenetelmien ratkaisuvoima on toistaiseksi rajoittuneempi kuin aivojen ratkaisuvoima; aivojen toimintaa kokonaisuutena ei ole vielä pystytty kovinkaan hyvin mallintamaan [Haykin, 1994]. Neurolaskentamenetelmiä voidaan kuitenkin hyödyntää biologisin perustein monissa ongelmissa: ongelma saattaa olla vaikeasti ratkaistavissa algoritmisesti, mutta sama ongelma voi vaikuttaa ominaiselta aivojen tehtävältä. Esimerkiksi kasvojen tunnistaminen vaikuttaa ihmisaivoille helpolta tehtävältä.

Neuroverkkosovellusten ei kuitenkaan tarvitse rajoittua suorittamaan aivojen alitehtäviä, sillä neurolaskennan avulla voidaan hyödyntää tietokoneiden laskentakykyä ja ylittää biologisten aivojen suorituskyky jossakin tarkasti määritellyssä ongelmassa.

Osa neurolaskentamenetelmistä pyrkii jäljittelemään melko tarkasti oikeiden hermoverkkojen toimintaa, kuten esimerkiksi tässä tutkielmassa tarkasteltu päämenetelmä. On kuitenkin syytä korostaa, että monet neurolaskentamenetelmistä on optimoitu tiedonlouhinnan ja koneoppimisen ongelmiin, minkä

seurauksena kaikille neurolaskentamenetelmille ei välttämättä löydy suoria biologisia perusteita.

Yhteenvetona voidaan todeta, että keinotekoisten neuroverkkojen avulla mallinnetaan aivojen monimutkaista, epälineaarista ja rinnakkaiseen prosessointiin perustuvaa toimintatapaa [Haykin, 1994].

2.3. Keinotekoisten neuroverkkomenetelmien yleispiirteet

Keinotekoinen neuroverkko on *laskentayksiköistä* eli neuroneista koostuva kokonaisuus. Neuroverkon tarkoituksena on muodostaa sovelluskohteen piirteitä jäljittelevä sisäinen esitys laskentayksiköiden yhteistoimintaa säätämällä. Laskentayksiköt voivat olla toiminnaltaan hyvin erilaisia eri neuroverkkomallien välillä.

Neuroverkkomallista ja käyttötavasta riippuen, sisäistä esitystä saatetaan hyödyntää eri tavoin. Esimerkiksi luokittelutehtävissä laskentayksiköiden toimintaa säädetään siten, että sisääntuloarvoja yhdistetään todennäköiseen ulostuloarvoon. Toisaalta ohjaamattoman klusteroivan neuroverkkomenetelmän avulla opittu neuroverkon sisäinen esitys voidaan visualisoida analyysin tuloksena.

Plastisuus tarkoittaa aivojen kykyä oppia. Plastisuuden käsitettä käytetään myös keinotekoisten neuroverkkojen yhteydessä. Neuroverkon oppimisvaihe on välttämätön ennen kuin sisäistä esitystä voidaan tulkita mielekkäästi. Toisin sanoen neuroverkot muodostuvat aivojen tavoin esivalmiista rakenteista, joita hienosäädetään ympäristön suhteen sopiviksi.

Hebbin oppiminen (engl. *Hebbian learning*) on hyvin tunnettu neuroverkkoihin liittyvä oppimisperiaate, jonka mukaan neuronien A ja B välistä yhteyttä vahvistetaan, jos A:n aksoni on lähellä B:tä ja A osallistuu toistuvasti B:n laukaisemiseen.

Keinotekoinen neuroverkko on aivojen hermoverkkojen tavoin hyvin vikasietoinen kokonaisuus. Aivot säilyttävät toimintakykynsä, vaikka neuraalinen rakenne osittain rappeutuisikin. Neuroverkosta ei varsinaisesti tuhoudu laskentayksiköitä, mutta muiden laskentayksiköiden paikkaava vaikutus on havaittavissa, sillä erilaisilla neuroverkkotopologiaan liittyvillä valinnoilla voidaan saavuttaa hyvin samanlainen tunnistustarkkuus.

Tyypillisesti neuroverkon tunnistuskyky kasvaa, kun neuronien määrä lisääntyy. Suuren neuronimäärän seurauksena neuroverkko saattaa kuitenkin oppia liian hyvin, minkä seurauksena se korostaa liikaa oppimiaan piirteitä ja menettää yleistämiskykyään. Tätä ilmiötä kutsutaan *ylioppimiseksi*. Ylioppimista saattaa aiheuttaa myös epäsopiva syötejoukko.

Aivojen tavoin myös keinotekoiset neuroverkot ovat hajautettuja rakenteita, minkä seurauksena neuroverkko sisäistää hahmoavaruuden epälineaarisia piirteitä. Monet monimutkaiset ja tärkeät systeemit, kuten esimerkiksi puheen tuottamisen taustalla oleva fyysinen mekanismi, ovat luonteeltaan epälineaarisia [Haykin, 1994].

Neuroverkon oppimistavan avulla voidaan luokitella toiminnaltaan ja käyttötarkoitukseltaan melko erilaisia menetelmiä. Oppimistapa on yleinen luokittelukriteeri neurolaskentamenetelmille. Ohjattua ja ohjaamatonta oppimista tarkastellaan tarkemmin seuraavissa alaluvuissa.

2.4. Ohjattu oppiminen

Ohjattua oppimista voidaan käyttää jos historiadataan ja asiantuntemukseen liittyvät esivaatimukset täyttyvät. Ohjatussa oppimisessa menetelmälle esitetään esimerkkitapauksia joiden perusteella asianmukainen algoritmi kouluttaa sisäistä esitystä.

Esimerkiksi lääketieteessä ohjattua oppimista on sovellettu diagnosoinnin yhteydessä. Tiettyä sairautta sairastavien ja terveiden verrokkien keskuudesta kerättyä lääketieteellistä aineistoa voidaan opettaa ohjatulle menetelmälle, minkä jälkeen menetelmää voidaan hyödyntää uusien sairastapausten diagnosoinnin yhteydessä. Kyse on siis sen yleistämisestä, miten sisääntuloarvot tulee yhdistää mahdollisiin ulostuloarvoihin. Lääketieteellisen testin tapauksessa ulostuloarvo saattaa esimerkiksi kertoa, että onko henkilöllä kohonnut riski sairastua tiettyyn tautiin. Jos sovelluskohteeseen on valittu sopiva ohjatusti opetettava menetelmä ja data on esitetty hyvin, niin menetelmä oppii yleistämään opetussyötteistä niitä vastaavien tulosten epälineaariset piirteet.

Ohjatun oppimisen käyttäminen on mahdollista, kun saatavilla on paljon laadukasta dataa ja tieto siitä, miten menetelmän tulee luokitella datasta koostetut syötteet. Käytännössä ohjatun opetuksen toteuttaminen saattaa olla kuitenkin liian vaikeaa, kallista tai aikaa vievää. Kuten tiedonlouhinnassa yleensä, ohjattuun oppimiseen perustuvien neuroverkkojen soveltamiseksi ei riitä pelkkä teoreettinen ymmärrys menetelmän toiminnasta, sillä yleensä vaaditaan lisäksi paljon sovelluskohteeseen liittyvää asiantuntemusta.

Tarkastellaan seuraavaksi lyhyesti yksittäistä *Perceptron-laskentayksikköä* ja rinnakkaisten Perceptron-laskentayksiköiden muodostamaa *yksikerroksista Perceptron-neuroverkkoa.* Alakohdan lopuksi tarkastellaan rinnakkaisten ja peräkkäisten aktivaatiofunktiolla varustettujen Perceptron-laskentayksiköiden muodostamaa MLP-neuroverkkoa.

Puhuttaessa neuroverkoista yleisesti tarkoitetaan usein nimenomaan MLP-menetelmää.

2.4.1. Perceptron-laskentayksiköstä ja yksikerroksisesta Perceptronista

Rosenblattin [1962] kehittämä Perceptron-laskentayksikkö ottaa vastaan syötevektorin $\boldsymbol{x}$, joka on painotettu painoarvovektorilla $\boldsymbol{w}$. Perceptron-neuronille lasketaan

$$f(x) = \begin{cases} 1, kun\ \boldsymbol{w} \cdot \boldsymbol{x} + b > 0 \\ 0, kun\ \boldsymbol{w} \cdot \boldsymbol{x} + b \leq 0 \end{cases}.$$

Neuroni aktivoituu, jos painovektorin ja syötevektorin pistetulo ylittää kynnysarvon b. Painovektorin $\boldsymbol{w}$ arvo opetetaan neuroverkon opetusvaiheessa. Rosenblattin [1962] ideana oli yhdistää useita Perceptron-laskentayksiköitä rinnakkain, minkä seurauksena muodostuu yksikerroksinen Perceptron-neuroverkko. Tosin tällaisella yksikerroksisella Perceptron-neuroverkolla voidaan suorittaa mielekästä luokittelua vain silloin, kun luokat ovat lineaarisesti erottuvia. XOR-ongelma on kuuluisa esimerkki yksinkertaisesta tilanteesta, jota ei voida ratkaista käyttämällä yksikerroksista Perceptron-neuroverkkoa. XOR-ongelman esittelee esimerkiksi Haykin [1994].

2.4.2. Monikerroksisesta Perceptronista

Edellisessä alakohdassa tarkasteltiin yksittäistä Perceptron-neuronia ja rinnakkaisten Perceptron-neuronien muodostamaa neuroverkkoa. Hyödyntämällä sopivalla tavalla yhdistettyjä Perceptron-neuroneita muodostuu hilamainen MLP-rakenne. MLP-neuronit on yhdistetty kuvassa 2 esitetyllä tavalla, mikä mahdollistaa syötesolmuista eteenpäin syöttämisen.

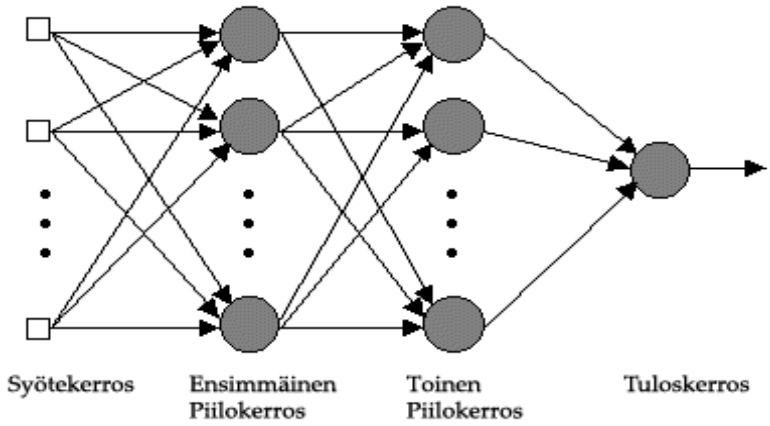
MLP:n opetusvaiheessa neuroverkkoa korjataan painoarvoja säätämällä, kunnes se on oppinut luokittelemaan syötteet niitä vastaaviin luokkiin. Painovektorien säätämiseen käytetään usein Rumelhartin ja muiden [1986] kehittämää *takaisinkytkentää* (engl. *backpropagation*), jonka kehittämistä voidaan pitää tärkeänä edistysaskeleena MLP-tutkimuksessa [Haykin, 1994].

Takaisinkytkennän periaatteena on korjata opetusvaiheessa ilmennyttä luokitteluvirhettä peruuttamalla neuroverkossa ja korjaamalla painoarvoja siten, että jatkossa kyseinen syöte luokiteltaisiin todennäköisemmin oikein. Takaisinkytkennän aikana pitää säilyttää aikaisempien opetussyötteiden perusteella tehdyt muutokset mahdollisimman hyvin. Takaisinkytkennän avulla pyritään ratkaisemaan Minskyn [1961] määrittelemä *credit-assignment–ongelma*. Kun takaisinkytkennän periaatteet toteutuvat, neuroverkko oppii yleistämään opetussyötteiden edustaman hahmoavaruuden piirteet.

Kuvassa 2 esitetään neuroneista muodostunut MLP-neuroverkko, josta voidaan havaita, että neuronien syöte muodostuu ensimmäisessä piilokerroksessa suoraan syötevektorin arvoista. Syötekerroksen jälkeisissä kerroksissa neuronit saavat syötteen edeltävän kerroksen neuroneilta. Viimeisen kerroksen (tulos-

kerroksen) toiminnan perusteella on mahdollista päätellä neuroverkon vaste syötteelle, mikä on myös analyysin luokittelutulos. Kuvan 2 tapauksen MLP-neuroverkko käsittää yhden ulostulosolmun, minkä seurauksena neuroverkko arvioi syötteiden kuulumista yhteen luokkaan.

MLP ei ole tämän tutkielman päämenetelmä, joten sitä ei käsitellä yksityiskohtaisesti. MLP:tä käsittelee tarkasti esimerkiksi Haykin [1994].



Kuva 2. Kahdella piilokerroksella varustettu MLP.

2.4.3. MLP:tä vastaavat uudemmat menetelmät

Vaikka termin ”oppiminen” analogia aivotoimintaan on osuva, on syytä korostaa käsitteen tärkeyttä neurolaskennan ulkopuolella tiedonlouhinnan ja koneoppimisen yhteydessä. Esimerkiksi Cortesin ja Vapnikin [1995] kehittämää ohjattuun oppimiseen perustuvaa *tukivektorikonetta* (engl. *support vector machine*, *SVM*) voidaan käyttää monissa kohteissa vaihtoehtoisena menetelmänä MLP:lle. SVM suoriutuu monista perinteisistä luokittelutehtävistä usein tarkemmin ja nopeammin kuin MLP, minkä seurauksena SVM on osittain syrjäyttänyt MLP:n. SVM:n ja MLP:n yhteyttä tarkastelee esimerkiksi Collobert ja Bengio [2004].

Viime vuosina MLP:tä ja muita neuroverkkomenetelmiä on sovellettu koneoppimisen alalla niin sanotussa *syväoppimisessa* (engl. *deep learning*). Syväoppimisessa on kyse hyvän yleistämiskyvyn saavuttamisesta yhdistelemällä ohjaamattomaan ja ohjattuun oppimiseen liittyviä periaatteita. Syväoppimiselle on mielenkiintoa erityisesti tekoälytutkimuksen näkökulmasta, sillä datan esiprosessointivaihetta on pystytty keventämään, eikä ulkoisen opettajan rooli ole enää niin tärkeä [Bengio, 2009].

2.5. Ohjaamaton oppiminen

Ohjaamattomaan oppimiseen perustuvaa menetelmää voidaan soveltaa tiedonlouhintaongelmiin, joissa eri luokkia yhdistävät ja erottavat piirteet ovat epäselviä, mutta luokkien piirteiden uskotaan kuitenkin sisältyvän dataan. Jos data edustaa eri luokkia ja on esiprosessoitu hyvin, niin on mahdollista kehittää malli, joka kuvaa hahmoavaruuden tärkeimpiä piirteitä hävittämällä vähemmän tärkeitä piirteitä.

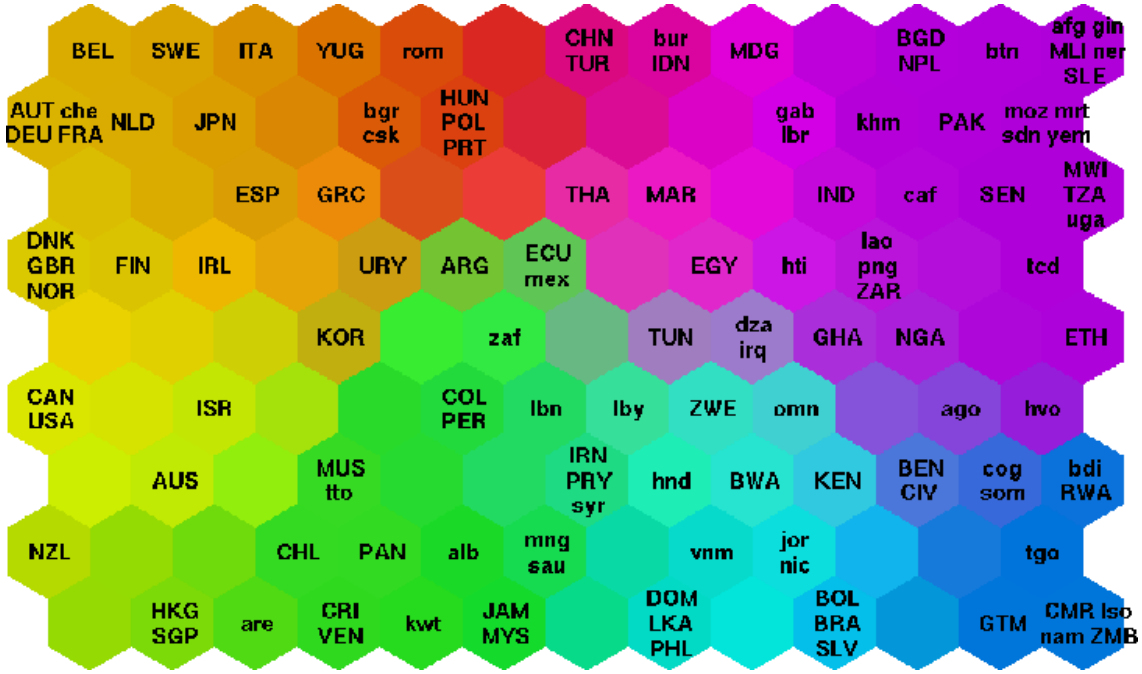
Ohjaamaton oppiminen asettaa analysoitavalle datalle vähemmän esivaatimuksia kuin ohjattu oppiminen, sillä luokiteltua ohjausdataa ei tarvitse olla etukäteen saatavilla, eikä asiantuntemuksen tarvitse olla yhtä hyvää kuin ohjatun oppimisen yhteydessä – ohjaamattoman oppimisen avulla pyritään usein nimenomaan lisäämään asiantuntemusta. Ihmisillä on hyvä kyky havaita erilaisia säännönmukaisuuksia visuaalisista esityksistä, joten usein data-analyysin tarkoituksena on saavuttaa tulos, joka on visualisoitavissa tietämyksen muodostamiseksi.

Toisaalta ohjaamattomaan oppimiseen liittyvät ongelmat vaikuttavat ohjattua oppimista haastavammilta, sillä ohjaamattoman menetelmän pitäisi pystyä rakentamaan dataan sisältyvien suhteiden perusteella edustava esitys ilman minkäänlaista ulkopuolista ohjausta. Lisäksi ohjatussa oppimisessa osa opetusjoukosta voidaan määrätä testijoukoksi, minkä avulla analyysin tulosta pystytään arvioimaan. Ohjaamattomassa oppimisessa tulosten arviointi on haastavampaa.

2.6. Alustavasti Kohosen itseorganisoituvasta kartasta

Kohosen itseorganisoituva kartta on paljon tutkittu ohjaamattomaan oppimiseen perustuva neurolaskentamenetelmä, jonka tuloksena on hahmoavaruutta jäljittelevä sisäinen esitys. Usein valmis sisäinen esitys visualisoidaan. Kartan lopullinen tila muodostuu, kun hahmoavaruudesta rakennettuja syötteitä on esitetty menetelmälle satunnaisessa järjestyksessä useiden iteraatioiden eli *epookkien* ajan.

Tarkastellaan seuraavaksi Kasken ja Kohosen [1996] luomaa itseorganisoitunutta karttaa. Kuvassa 3 on esitetty melko tyypillinen itseorganisoitunut kartta, jossa on visualisoitu eri valtioiden välinen suhde hyvinvoinnin tunnusmerkkien perusteella. Kartta on opetettu vastaamaan hahmoavaruutta, minkä jälkeen kartalle on lisätty eri maakoodeilla varustettuja leimoja (engl. *label*) mahdollisimman sopiviin kohtiin. Leimaamattomat tyhjät neuronit eivät edusta mitään valtiota.



Kuva 3. Eri valtioiden välinen hyvinvointi SOM:in avulla esitettynä [Kaski ja Kohonen, 1996].

Pelkästään kuvan 3 esityksen perusteella ei voida sanoa mitkä maat menestyvät paremmin kuin toiset, sillä vain maiden keskinäiset suhteet on visualisoitu. Muiden tarkastelujen avulla on mahdollista selvittää, että esimerkiksi vasemmassa yläkulmassa sijaitseva Belgia edustaa menestyneintä kärkijoukkoa, mutta oikeassa alakulmassa sijaitsevat Kamerun, Lesotho, Namibia ja Zambia täyttävät heikosti hyvinvoinnin tunnusmerkit.

SOM-neuroni on hyvin erilainen kuin aiemmin esitelty Perceptron-neuroni. SOM käyttäytyy lateraalisesti, minkä seurauksena vierekkäisillä neuroneilla on erityinen suhde.

SOM-neuroni oppii edustamaan hahmoavaruudesta esitettyjen tapausten perusteella tietynlaista mallia. Kuvan 3 esityksessä värien sävyllä ja kylläisyydellä on erityinen merkitys. Vierekkäisten neuronien eriävä väritys perustuu neuronin mallia edustavan vektorin suuntaan. Värikoodatussa esityksessä värisävyn liukuva vaihtelu vastaa alueiden profiilin vaihtelua siirryttäessä tarkasteltavasta neuronista naapuroivaan neuroniin. Tasasävyiset alueet ovat samanlaisten esiintymien muodostamia klustereita.

Kuvan 3 värivalinnoilla on erityinen merkitys. Kellertävällä taustavärillä väritetty klusteri sisältää pääasiassa OECD-maita, punainen alue koostuu pääasiassa Itä-Euroopan maista, vihreä alue koostuu pääasiassa Keski- ja Etelä-Amerikan maista, purppura alue koostuu pääasiassa Aasian ja Afrikan maista ja turkoosi ja sininen alue koostuvat pääasiassa Afrikan maista.

Signaalinkäsittelyssä on jo ennen SOM:in kehittämistä käytetty klassista vektorikvantisointia (engl. *vector quantization*, VQ), jota SOM muistuttaa voimakkaasti. SOM:in generoimalla kartalla mallien spatiaalinen järjestys kuitenkin ilmaisee mallien erilaisuutta. Toisin sanoen SOM yhdistää vektorikvantisoinnin ja vektoriprojektion.

3. Kohosen itseorganisoituva kartta

3.1. Motivaatio

Aivot käsittelevät muiden tehtävien ohessa aistiärsykkeitä, minkä mallintamista on tutkittu paljon. Itseorganisoituva kartta perustuu useista neuroneista muodostuvien keskittymien järjestelmiin, joita kutsutaan aivokartoiksi tai lyhyesti kartoiksi (engl. *brain map*).

Kohonen [2013] kertoo Mountcastlen [1957] ja Hubelin ja Wieselin [1962] tutkimusten selittäneen erityisesti aivokuoren toimintaa aistiärsykkeiden käsittelyssä. Aivokartan neuronien todetaan muodostavan ryhmittymiä siten, että samoja piirteitä sisältävät aistiärsykkeet aiheuttavat lähekkäisten neuronien aktiivisuutta. Tietty keskittymä aivokuorella edustaa jonkin ominaisuuden arvoaluetta. Nämä havainnot eivät kuitenkaan vielä selittäneet sitä, miten kartat voivat organisoitua muodostaakseen paikallisia keskittymiä ja pitämään tiettyjä ärsykkeitä samanlaisina. Oletuksena oli, että neuronien ryhmittyminen kartalle on täysin geneettisesti määritelty. Vasta myöhemmin on selvinnyt, että aistiärsykkeet vaikuttavat niille herkkien neuronien ryhmittymiseen. Vähintään aivokuoren neuronien tarkka järjestys ja signaalien skaalaus säätyvät aistihavaintojen perusteella. [Kohonen, 2013]

Osa teoreettisista biologeista kiinnostui mahdollisuudesta toteuttaa ärsykkeiden perusteella rakentuvia keinotekoisia neuraalisia järjestelmiä. Kohonen [2013] haluaa mainita Malsburgin [1973] ja Amarin [1980] työt merkittävinä teoreettisina todistuksina keinotekoisen syöteohjatun itseorganisoitumisen puolesta. Näissä tutkimuksissa esiteltyjen karttojen rakenne mukautuu *kilpailevan oppimisen* perusteella. Kilpailevan oppimisen mukaan neuroneita säädetään syötteiden esittämisen yhteydessä siten, että syötteille laskettiin niitä parhaiten edustavat voittajaneuronit, joita muokattiin lähemmäksi syötteen piirteitä. Näin ollen voittajaneuronit herkistyivät esitetyn ärsykkeen kaltaisten ärsykkeiden ominaisuuksille.

Ensimmäiset keinotekoiset itseorganisoituvat menetelmät mallinsivat karttoja, joilla voitiin todeta olevan aivoille ominaisia piirteitä, mutta ne eivät kuitenkaan vielä sopineet käytännössä sovellettaviksi data-analyysissa. Näiden aivokuoren toimintaa mallintavien menetelmien avulla saatiin muodostettua syötteiden ominaisuuksien perusteella mukautuvia karttoja. Kartat koostuivat kuitenkin paikallisista kertymistä, joiden välille ei saavutettu spatiaalista järjestystä. Toisin sanoen samankaltaisille syötteille herkät neuronit saattoivat sijaita lähekkäin samoissa keskittymissä, mutta samankaltaiset keskittymät sijaitsivat kartalla hajautetusti satunnaisessa järjestyksessä. [Kohonen, 2013]

3.2. Perinteinen algoritmi

Alkuperäisessä SOM-algoritmissa mallit on laskettu approksimointiprosessin avulla, jossa syötevektorit on esitetty yksi kerrallaan satunnaisessa järjestyksessä toistamalla, kunnes riittävän tasapainoinen tila on saavutettu.

Kartta koostuu sopivasta määrästä neuroneita. Jokainen neuroni oppii algoritmin toistamisen (epookkien) seurauksena edustamaan jotakin mallia. Jos datajoukon esiintymien välillä on olemassa spatiaalinen järjestys, niin parhaimmillaan SOM pystyy löytämään sen.

Algoritmissa $t = 1,2,...$ on syötteen esittämisen ajanhetki. Ennen ensimmäistä ajanhetkeä neuronien malleiksi on määritelty satunnaiset vektorit.

Kaikille syötteille $\boldsymbol{x}(t)$ määritellään aluksi voittaja $\boldsymbol{m}_c(t)$, joka on parhaiten täsmäävä neuroni (engl. *best matching unit*, *BMU*) suorittamalla vertailu

$$\forall i, ||\boldsymbol{x}(t) - \boldsymbol{m}_c(t)|| \leq ||\boldsymbol{x}(t) - \boldsymbol{m}_i(t)||.$$

Toisessa vaiheessa karttaa opetetaan voittajaneuronin perusteella. Voittajaneuronin läheisyydessä olevien neuronien halutaan myös oppivan, joten päivitetään voittajaneuronia $c = c(\boldsymbol{x})$ ympäröivä naapurusto laskemalla

$$\boldsymbol{m}_i(t+1) = \boldsymbol{m}_i(t) + h_{c(x),i}(t)(\boldsymbol{x}(t) - \boldsymbol{m}_i(t)),$$

missä $h_{c(x),i}(t)$ on naapurustofunktio ajanhetkellä t. Naapurustofunktio on opetuskerroin, jonka voimakkuuteen vaikuttavat ajanhetki ja opetuksen kohteena olevan neuronin etäisyys voittajaneuronista. Naapurustofunktiota käsitellään tarkemmin seuraavassa kohdassa.

3.3. Algoritmin toimintaan vaikuttavat yksityiskohdat

SOM:iin liittyy monia huomioitavia yksityiskohtia, mutta laadukas esiprosessointi on tärkeämpää kuin SOM:in yksityiskohtiin liittyvät valinnat. Koska kilpaileva oppiminen toimii haastavissakin olosuhteissa, SOM on hyvin joustava lähtöarvojen ja muiden laskentaan vaikuttavien yksityiskohtien suhteen. Yksityiskohtien vaikutus pitää kuitenkin ymmärtää, sillä huonoilla valinnoilla on mahdollista pilata koko analyysi. Yksityiskohtien valinnassa kannattaa toisinaan huomioida sovelluskohteen oletetut piirteet ja analyysin odotetut tulokset, mutta joskus liiallisten oletusten tekeminen saattaa haitata odottamattomien mallien paljastumista.

Naapurustofunktio on mahdollista toteuttaa monella eri tavalla, mutta kirjallisuudessa esiintyvät kaksi tapaa erityisen usein.

Yksinkertaisen naapurustofunktion $h_{c(x),i}(t)$ perusteella indeksi i koskee ajanhetkien edetessä yhä harvempaa neuronia siten, että aina vain lähimmät neuronit lasketaan naapurustoon kuuluviksi. Tätä naapurustofunktiota kutsutaan toisinaan kuplafunktioksi. Jos voittajaneuronin naapurusto koostuu indek-

seistä N_t, niin $h_{c(x),i}(t) = \alpha(t)$, jos $i \in N_c$ ja $h_{c(x),i}(t) = 0$, jos $i \notin N_c$. Oppimisnopeutta $0 < \alpha(t) < 1$ tulee vähentää opetuksen edetessä. Usein $\alpha(t)$ on monotonisesti (esimerkiksi hyperbolisesti, eksponentiaalisesti tai paloittain määritellysti) vähenevä funktio [Kohonen, 2013].

Toinen yleisesti käytetty naapurustofunktio on Gaussin funktio

$$h_{c(x),i}(t) = \alpha(t) \exp\left(-\frac{sqdist(c,i)}{2\sigma^2(t)}\right),$$

missä $sqdist(c,i)$ on voittajaneuronin c ja neuronin i välisen geometrisen etäisyyden (esim. kortteli- tai euklidinen metriikka) neliö [Kohonen, 2013]. Funktio $\sigma(t)$ on toinen monotonisesti laskeva funktio. Määrittelemällä opetuksen vaikutussäteen määräävä $\sigma(t)$ monotonisesti vähenevästi, vaikuttaa opetus t:n ollessa pieni suureen osaan kartan neuroneista, mutta kun t kasvaa, algoritmi hienosäätää pienen naapuruston samankaltaisia malleja. Kohonen [2013] toteaa, että funktion $\sigma(t)$ valinnalla ei ole suurta merkitystä; $\sigma(t)$ voi aluksi olla esimerkiksi puolet kartan halkaisijasta.

Lähimmät naapurit voidaan määritellä eri tavoilla. Kuusikulmio on erityisen suosittu valinta neuronin muodoksi, sillä kuusikulmioista rakentuvan kartan avulla saavutetaan vaikutelma, jossa naapuruston solut ovat yhtä lähellä.

Kohonen [2013] erottelee kartan opetuksesta kaksi vaihetta: karkea ja hienosäätö. Karkeassa vaiheessa kartan tila suppenee karkeaan approksimaatioon hahmoavaruudesta. Hienosäätövaiheessa hahmoavaruuden malleja hienosäädetään vielä monen iteraation ajan, kunnes lopetusehto on saavutettu. Lopetusehtona voi olla lähes muuttumaton kartan tila tai se, että iteraatioita on suoritettu ennalta määrätty lukumäärä. Kohosen [2013] mukaan tyypillisessä opetuksessa karkea vaihe saattaa vaatia 1000 iteraatiota ja hienosäätövaihe jopa 10000 iteraatiota.

Kohosen [2013] mukaan algoritmin suppeneminen spatiaalisen järjestyksen saavuttamiseksi on todistettu tietyissä tapauksissa. Lisäksi tulosten laadun arvioimiseksi on käytettävissä matemaattisia työkaluja. SOM:iin liittyviä teoreettisia tarkasteluja tekivät esimerkiksi Cottrell ja muut [1997].

Neuronien lukumäärän valitseminen on yleisimpiä SOM:iin liittyviä kysymyksiä. Usein apuna käytetään tietoa datan pääkomponenteista. Kuitenkin yleensä päätös sopivasta neuronien lukumäärästä tehdään erikokoisilla kartoilla suoritettujen testien jälkeen. [Kohonen, 2013]

Jotkin SOM-variaatiot pyrkivät ratkaisemaan solmujen lukumäärän kiinnittämiseen liittyvän ongelman. Yksi idea on kasvattaa kartan kokoa tarpeen mukaan. Esimerkiksi Friztke [1994] on tutkinut tätä strategiaa.

Jos kartan data on kehämäistä, niin saattaa olla havainnollista käyttää esityksiä, joissa kartan päät tai sivut on yhdistetty toisiinsa. Tällöin kaksiulotteinen kartta voidaan esittää sivuista rajoittamattomana, sylinterin tai toroidin muotoisena rakenteena. Yleensä kaksiulotteinen kartta esitetään kuitenkin yksinkertaisesti tasona, sillä sen esittäminen on helppoa.

Kohonen [2013] suosittelee käyttämään jotakin SOM:in valmista toteutusta, kuten SOM_PAK- tai SOM Toolbox -toteutuksia, sillä käytännön toteutukseen liittyy monia huomioon otettavia seikkoja. [Kohonen, 2013]

3.4. Syötteen esittäminen

Vaikka tiedonlouhintamenetelmän algoritminen osa olisi toteutettu oikein, niin datan huono esittäminen voi tehdä data-analyysistä järjettömän. Esimerkiksi syötteiden huonon esitystavan seurauksena mitättömät erot syötteissä saattavat korostua analyysin pilaavalla tavalla.

Aiemmin on havaittu, että samankaltaisia instansseja (tapauksia) yhdistävät SOM:in mallit sijaitsevat toistensa läheisyydessä. Kohonen [2013] toteaa, että samankaltaisuus esimerkiksi ihmisten tai aikakausien välillä ei ole helposti esitettävissä etäisyyksinä, sillä samanlaisuuden pitää perustua matemaattisiin tunnusmerkkeihin.

Kun instansseja kuvaavat muuttujat on valittu, kannattaa ne skaalata siten, että eri komponenttien arvot ovat keskenään vertailtavia. Komponenttien alkuperäiset arvot jakautuvat mahdollisesti hyvin eri tavalla eri lukualueille. Sopiva menetelmä tämän ongelman ratkaisemiseksi on suorittaa varianssien normalisointi tai samojen suurimpien ja pienimpien arvojen käyttäminen eri komponenttien välillä.

Sopivan skaalauksen jälkeen vektorien vertailu on mahdollista jo euklidisten etäisyyksien perusteella, mikä Kohosen [2013] mukaan riittää usein käytännön tilanteissa. Kohonen [2013] mainitsee, että usein vielä parempi tapa vertailun tekemiseksi on käyttää mallien normalisoitujen vektorien välistä pistetuloa. Pistetulo antaa kosinimitan arvon. Sillä verrataan vektorien välisten kulmien kosiniarvoja eli samalla itse kulmia.

Algoritmin yhteydessä määrättiin alustamaan mallit aluksi satunnaisilla vektoreilla, mikä ei kuitenkaan yleensä ole järkevää, sillä SOM ei välttämättä järjesty mielekkäästi tai se järjestyy hyvin hitaasti [Kohonen, 2013]. Käytännössä SOM-neuronit kannattaa alustaa datajoukkoa kuvailevien tunnusmerkkien perusteella.

Esimerkiksi kuva-analyysi on SOM:in yleinen sovelluskohde, jonka yhteydessä data kuitenkin esitetään toisinaan huonosti. Erityisesti pikselien suora esittäminen on ongelmallista, sillä se korostaa vaihtelua yleensä väärin. Syöt-

teiden tulisi olla sellaisia, että SOM pystyisi tunnistamaan ongelmitta samanlaiset tapaukset esimerkiksi kierrosta tai valaistusolosuhteista riippumatta. Pikselien suoran esittämisen sijaan tulisi käyttää muita tapoja, kuten värispektogrammia tai pääkomponenttiesitystä. Lisäksi jos sovelluskohteessa on mahdollista käyttää esitystapoja, jotka eivät ole herkkiä epäkiinnostavan vaihtelun suhteen, niin syötteiden ulotteisuus yleensä pienenee ja laskenta helpottuu. Kierron tai jonkin muun operaation vaikutus voidaan poistaa esimerkiksi käyttämällä *ASSOM*:ia (engl. *adaptive-subspace SOM*) [Kohonen, 1995], jonka ideana on oppia tunnistamaan syötteiden perusteella suodatettava operaatio. [Kohonen, 2013]

3.5. Tehokkaampi eräajoalgoritmi

Aiemmin esitetty perinteinen algoritmi on yleisin versio itseorganisoituvan kartan algoritmista. Yleinen versio on yksinkertainen ja sopiva pääidean selvittämiseen, mutta Kohosen [2013] mukaan käytännön sovelluksissa kannattaa usein käyttää eräajoon perustuvaa SOM-algoritmia (eräajoalgoritmia). Eräajoalgoritmi on tehokkaampi, vähemmän parametreja vaativa ja joskus jopa perinteistä algoritmia tarkempi. Syötteiden esittäminen erissä vaatii muutamasta opetusiteraatiosta muutamaan tuhanteen opetusiteraatiota, joten eräajoalgoritmi kouluttaa kartan suuruusluokkaa tehokkaammin kuin perinteinen algoritmi [Kohonen, 2013].

Käytännössä eräajoalgoritmi on tärkeä tämän tutkielman kannalta, sillä Matlab-ympäristössä perinteisen algoritmin tehottomuus korostuu.

Eräalgoritmin toiminta perustuu siihen, että kaikki syötteet käydään kerran läpi ja lasketaan syötteiden yhteisvaikutus, jota käytetään verkon opettamiseen. Yhteisvaikutus painotetaan naapurustofunktion avulla. Perinteisen algoritmin koulutusvaihe muutetaan muotoon

$$\boldsymbol{m}_i(t+1) = \frac{\sum_{j=1}^{n} h_{c(\boldsymbol{x}),i}(t)\boldsymbol{x}_j}{\sum_{j=1}^{n} h_{c(\boldsymbol{x}),i}(t)},$$

jossa n on syötevektorien lukumäärä ja $h_{c(\boldsymbol{x}),i}(t)$ naapurusto ajanhetkellä t. Malleja $\boldsymbol{m}_i(t)$ käytetään enää vain voittajan $c(\boldsymbol{x})$:n laskemiseksi.

3.6. Kartan laadun arviointi

Kvantisointivirhe (*kv*) on kvantisointimenetelmien yhteydessä usein käytetty tuloksen laadun arviointimenetelmä. Menetelmän periaatteena on laskea kaikkien opetusvektorien ja niiden BMU:iden välisten etäisyyksien keskiarvo. Kvantisointivirhe ei kuitenkaan kerro mitään spatiaalisesta järjestyksestä. Kvantisointivirhettä on mahdollista pienentää käyttämällä suuria karttoja, mut-

ta silloin kartan yleistämiskyky heikkenee, mikä saattaa olla sovelluskohteen kannalta epätoivottua.

Jos testattavan opetusvektorin paras ja toiseksi paras BMU eivät ole vierekkäin, on kyseessä *topografinen virhe* (*tv*). Koko kartan topografinen virhe saadaan laskemalla virheellisten topografisten testien osuus kaikista topografisista testeistä. Topografinen virheluku kuuluu välille [0,1]. Jos virheluku on nolla, niin kartta on testin näkökulmasta täydellinen.

3.7. SOM ja eksploratiivinen data-analyysi

Yleensä data-analyysin yhteydessä on tärkeää tuntea tarkkailtava kohde hyvin, jotta data-analyysin eri vaiheita pystyttäisiin toteuttamaan järkevällä tavalla. Kohteeseen liittyvää asiantuntemusta vaaditaan valmistelun, esiprosessoinnin, analyysin ja jälkikäsittelyn vaiheissa, eli kaikissa data-analyysin vaiheissa.

Eksploratiivisen data-analyysin (engl. *Exploratory Data Analysis, EDA*) tapauksessa asiantuntemukseen liittyvät vaatimukset ovat vähäisempiä, kuin data-analyysissa yleensä. EDA:n tärkein tavoite on nimenomaan asiantuntemuksen lisääminen. EDA voi sisältyä varsinaisen data-analyysin valmisteluvaiheeseen tai se voi tukea asiantuntijoita analysointitehtävissä. Kohosen [2013] mukaan EDA on usein erityisen sopiva SOM:in soveltamistapa.

Kuten jo aiemmin on todettu, itseorganisoituvalla kartalla solujen vektorit muodostavat järjestyksen, jossa vierekkäisten solujen vektorit edustavat hahmoavaruuden samankaltaisia esiintymiä.

Erilaisilla menetelmillä pystytään korostamaan kartan eri ominaisuuksia. Osa käytetyistä visualisointikeinoista on päällekkäisiä ja sopivan menetelmän valitseminen riippuu esityksen laatijan mieltymyksistä.

Mallivektorien vaihtelun esittämiseksi käytetään usein U-matriisia (engl. *unified distance matrix*), jonka avulla korostetaan vierekkäisten solujen erilaisuutta. Käytännössä U-matriisia käytetään kun klusterirajojen tunnistaminen on tärkeää. Tyypillisissä esityksissä vierekkäisten vektorien erilaisuutta korostetaan tummalla solun sävyllä ja samanlaisten solujen läheisyyttä puolestaan esitetään vaalealla sävyllä. Näin ollen U-matriisin avulla on mahdollista havaita myös klusterien erilaisuus. Solun erilaisuuden suuntaa voidaan korostaa käyttämällä solun värityksen sijaan solun särmän vaihtelevaa väritystä.

Komponenttiesitys on usein käytetty SOM:in visualisointitapa. Komponenttiesityksen avulla on mahdollista havaita eri komponenttien välinen suhde. Tietysti moniulotteisilla kartoilla kaikkien komponenttien samanaikainen esitys saattaa olla hyvin epäkäytännöllistä.

Komponenttiesitys on usein erinomainen menetelmä korrelaation etsimiseen, sillä sama solmu on vastaavassa kohtaa jokaisen komponentin esitykses-

sä. Lisäksi jos kartta sisältää paljon erilaisia malleja, saattaa kahden komponentin välinen korrelaatio olla tilastollisesti merkityksetöntä, mutta SOM-analyysin perusteella voidaan havaita tiettyjä klustereita, joiden välillä on korrelaatiota.

4. Virtualisoitujen palvelinresurssien käytön seuranta

4.1. Tietokonejärjestelmän mallintaminen SOM:in avulla

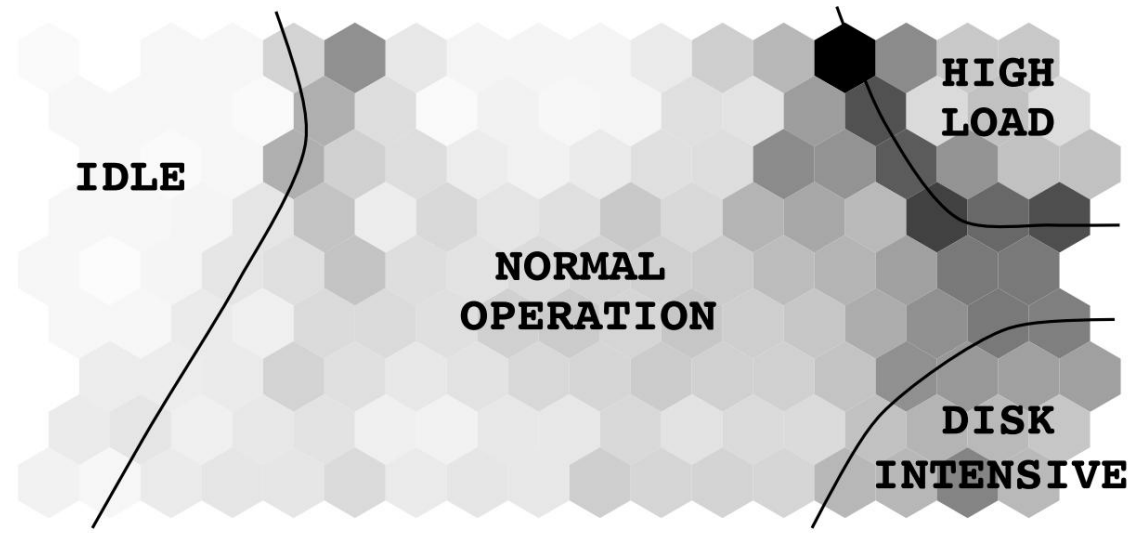
Simula ja muut [1996] käyttivät SOM:ia tietokonejärjestelmän prosessien analyysissa. Järjestelmästä seurattiin käyttöastetta ja verkkoliikenteen määrää. Havainnoista muodostettiin syötevektoreita, jotka koostuivat seuraavista yhdeksästä muuttujasta:

1. Jaetulta kiintolevyltä luetut lohkot
2. Jaetulle kiintolevylle kirjoitetut lohkot
3. Käyttäjän aiheuttama suoritinkäyttö prosentteina
4. Järjestelmän aiheuttama suoritinkäyttö prosentteina
5. Suorittimen keskeytyssignaalien määrä prosentteina
6. Siirrännän odottamiseen käytetty aika prosentteina
7. Suorittimen lepotilojen osuus prosentteina
8. Verkosta saapuvien pakettien määrä
9. Verkkoon lähtevien pakettien määrä.

Näihin muuttujiin liittyviä arvoja mitattiin järjestelmästä, joka oli normaalikäytössä. Tulosten perusteella luotu kartta on esitetty kuvassa 4. Kartan vasemmassa reunassa sijaitsee klusteri, jossa järjestelmä on lepotilassa. Normaalia käyttöä edustava klusteri on suurin, ja se sijaitsee kartan keskellä. Oikeassa yläreunassa sijaitsevat järjestelmää paljon kuormittavien tilojen klusteri ja oikeassa alareunassa on suurta levynkäyttöä vastaava klusteri. SOM löysi spatiaalisen järjestyksen hyvin, sillä järjestelmän käyttöaste suurenee edettäessä vasemmalta oikealle.

Simula ja muut [1996] arvioivat mielenkiintoiseksi lähestymistavaksi piirtää peräkkäisten BMU:iden välille kaaria, joiden avulla voi esittää muutosta SOM-kartalla. Koska SOM järjestää mallit spatiaaliseen järjestykseen, kaarien pituus kuvaa tilojen vaihtelun voimakkuutta. Tosin siirtymä klusterista toiseen saattaa todellisuudessa olla paljon suurempi kuin siirtymää vastaavan kaaren pituuden perusteella voisi arvioida.

Lisäksi Simula ja muut [1996] laajentavat edellä esiteltyä menetelmää siten, että jokaista neuronia kohti lisätään uusi SOM-kartta, jota kutsutaan dynaamiseksi kartaksi. Tällöin alkuperäistä karttaa voidaan kutsua päätason kartaksi. Dynaamista karttaa opetetaan tietyllä tavalla siten, että se sisältää kaikki siirtymät, jotka liittyvät vastaavaan päätason neuroniin. Menetelmää voidaan käyttää reaaliaikaisessa data-analyysissa, sillä dynaamisen kartan siirtymiä tulkitsemalla voidaan ennustaa seuraavia päätason siirtymiä.



Kuva 4. Järjestelmän tilojen sijainti SOM-kartalla.

4.2. Virtualisointi

Virtualisointi vähentää laitteiston ja ohjelmiston riippuvuutta toisistaan. Käytännössä virtualisoinnin avulla voidaan esimerkiksi asentaa monta (virtualisoitua) käyttöjärjestelmää laitteistoon, johon perinteisesti on ollut mahdollista asentaa vain yksi käyttöjärjestelmä. Tällaista virtualisoitua käyttöjärjestelmää kutsutaan *virtuaalikoneeksi*.

Virtualisointi ei ole uusi tekniikka, mutta nykypäivänä se on erittäin tärkeä käsite *pilvilaskennassa*. Virtualisointi on mahdollisesti pilvilaskennan sisältämistä tekniikoista kaikkein tärkein.

Pilvilaskenta on syntynyt kun laskentaresursseihin liittyviä kuluja on haluttu karsia. Tekniikan riippuvuus hitaista ja kalliista ihmisistä on vähentynyt, sillä pilvilaskenta yhdistelee ennennäkemättömän laajalla tavalla erilaisia tekniikoita. Tiedonlouhinnan näkökulmasta tällainen järjestelmien välinen toiminta on kiinnostavaa, sillä dataa on lähes rajattomasti saatavilla.

Kalliiden laskentaresurssien käyttöasteen toivotaan olevan niin korkea kuin mahdollista. Virtualisoinnin ansiosta on mahdollista selvittää tarkasti virtuaalikoneiden varaamien resurssien määrä. Ilman virtualisointia tällainen selvitys olisi hyvin haastavaa, sillä kaikilta erilaisilta käyttöjärjestelmiltä pitäisi kerätä suorituskykydataa. Lisäksi eri käyttöjärjestelmät mittaisivat mahdollisesti eri kohteita, esittäisivät tulokset eri tarkkuudella ja käyttäisivät erilaisia mittausintervalleja.

4.3. Yksityinen pilvipalvelu Nokia Networksin ohjelmistokehityksessä

Nokia Networks (Nokia) on noin 50 000 työntekijää työllistävä tietoliikenneohjelmistoja ja -laitteita valmistava yritys. Nokia toimittaa esimerkiksi neljännen sukupolven LTE-ratkaisuja useille asiakkaille eri puolille maailmaa.

NetAct on Nokian valmistama verkonhallintajärjestelmä (engl. *Operations Support Systems*, *OSS*), joka sisältää tietoliikenneverkon ylläpitoon vaadittavia työkaluja. Toimiva tietoliikenneverkko sisältää erilaisia teknologisia ratkaisuja, joita pitää pystyä seuraamaan, mittaamaan, konfiguroimaan ja optimoimaan.

Ohjelmistokehityksen tarpeisiin pitää pystyä tuottamaan sopivia kehitysympäristöjä järkevillä kustannuksilla. NetActin kehitysympäristöjä kutsutaan NetAct-laboratorioiksi. Laboratorioiden kustannukset ovat suuret, sillä laboratorioita on paljon, ne varaavat paljon laskentaresursseja ja niiden hyvä suorituskyky on haluttu varmistaa laadukkaalla palvelinlaitteistolla.

Laboratoriopalvelut on päätetty tuottaa sisäisen henkilöstön toimesta pilvipalveluna, tarkemmin privaattipilvenä. Privaattipilvi on ollut mahdollista toteuttaa, sillä osaavinta henkilöstöä ja sopivaa laitteistoa löytyy Nokian sisältä.

Nokia on hyödyntänyt useita virtualisoinnin tarjoamista mahdollisuuksista, joista erityisen tärkeänä pidetään laboratorioiden varaustoimintojen automatisointia.

Laboratoriokonfiguraatio määrittelee sen, kuinka monesta ja minkälaisista virtuaalikoneista luokan laboratoriot koostuvat. Eri laboratoriotyypin laboratorioiden vertaileminen on vaikeaa, sillä niiden virtuaalikoneiden rooli on erilainen ja virtuaalikoneita on eri lukumäärä.

Konfiguraatioluokkien kokoonpanoja myös muutetaan aina tarvittaessa, mikä vaikuttaa myös olemassa olevien laboratorioiden kokoonpanoihin. Konfiguraatiopäivitysten mahdollisuus saattaa hankaloittaa reaaliaikaisten analysointityökalujen kehittämistä ja ylläpitoa.

Suurimpaan konfiguraatioluokkaan kuuluu noin 400 laboratoriota. Tarkkailu rajoitetaan vain näihin suurimman luokan laboratorioihin, joten jatkossa laboratoriolla tarkoitetaan vain suurimpaan konfiguraatioluokkaan kuuluvia laboratorioita.

Laboratoriot koostuvat aina vähintään kymmenestä perusnoodista. Huomioidaan vain kymmenen noodia, sillä harvinaisia lisänoodeja ei huomioida analyysissa.

Jokaiseen laboratorioon liittyy tieto siitä, mikä ryhmä omistaa sen. Lisäksi laboratorioille on ilmoitettu jokin vaihe. Myöhemmin tulosten analyysin yhteydessä hyödynnetään tietoa ryhmästä ja vaiheesta.

4.4. Esiprojekti

4.4.1. Motivaatio - käyttöasteen mittaaminen

Koska Nokia on varannut paljon resursseja privaattipilven toteuttamiseen, halutaan resurssien käytön olevan tarkoituksenmukaista. Julkisen pilven tapauksessa käyttöaste pysyy yleensä hyvänä, sillä veloitettava maksu perustuu usein käyttömäärän tarkkaan mittaamiseen. Heinon [2010] mukaan yksityisen pilven kohdalla käytön seuraaminen on kuitenkin hankalampaa, sillä useimmat yritykset eivät halua lisätä byrokratiaa veloittamalla omilta osastoiltaan yksityisen pilven käytöstä.

Nokian laboratoriopilven toimintaa halutaan tehostaa, mutta säästöhakuisesta laboratoriokiintiöiden pienentämisestä on helposti paljon haittaa ohjelmistokehitykselle. Ohjelmistokehityksen laboratoriotarve on jatkuva ja ohjelmistokehitysyksikköjen varaustarpeista käydään lähes taukoamatonta keskustelua.

Laboratorioiden käyttöasteen selvittämiseksi on tilattu sisäisesti analysointityökaluja, mutta niitä ei ole yleensä pystytty toteuttamaan. Joitakin työkaluja on kehitetty, mutta ne on tehty tarkasti johonkin tiettyyn kohteeseen, eikä niitä voi käyttää yleisen käyttöasteen mittaamiseksi.

Yleisten käyttöasteen mittaamiseen tarkoitettujen työkalujen kehittäminen on haastavaa, sillä vaikka laboratorioihin olisikin asennettu sama tuote (NetAct-kehitysversio), laboratorioita käytetään eri tarkoituksiin. Käyttötapojen ero korostuu mahdollisesti eri ryhmien välillä, mitä selvitetään myöhemmin.

Ohjelmistokehitys saattaa pysähtyä tai häiriintyä, jos laboratorioita ei ole riittävästi saatavilla. Jotta käyttöaste olisi parempi, Nokian tulee pyrkiä lyhentämään yksittäisten laboratorioiden varausjaksoa käyttötarvetta vastaavaksi.

4.4.2. Lähestymistapojen arviointi ja karsinta

Yksi idea laboratorioiden arvioimiseksi liittyy erityisen näkymän toteuttamiseen. Näkymän avulla esitettäisiin laboratorioiden luokittelu käytettyihin ja käyttämättömiin. Näkymä esittäisi lähihistoriaan liittyviä tuloksia. Yksinkertainen analysointityökalu tähän ongelmaan voisi olla sellainen, joka luokittelee laboratorion käyttämättömäksi vain silloin, kun kaikki virtuaalikoneet on sammutettu. Tällaisten laboratorioiden osuutta testidatassa käsitellään myöhemmin.

Alun perin tämän tutkielman aiheeksi pohdittiin ohjattuun oppimiseen perustuvan työkalun toteuttamista, mutta jatkokaavailut paljastivat ilmeisiä ongelmia. Erityisesti selvä luokittelukriteerien puuttuminen ja ennalta tuntematon data vaikuttivat riskitekijöiltä. Lisäongelmaksi arvioitiin se, että samanlaisilla laboratorioilla on paljon erilaisia käyttötarkoituksia.

Koska ohjattuun oppimiseen perustuvan työkalun toteutus ei vaikuttanut järkevältä, arvioitiin eksploratiivista data-analyysia tukevia lähestymistapoja. Kohosen itseorganisoituvalla kartalla pohdittiin olevan erityisen mieluisia ominaisuuksia, jotka mahdollistaisivat monenlaisten tulkintojen tekemisen yhden analyysin pohjalta.

SOM:iin perustuva lähestymistapa todettiin turvallisemmaksi ja mahdollisesti hedelmällisemmäksi, sillä sen oletettiin antavan ainakin toisarvoisia tuloksia, joiden avulla kohdetta ymmärrettäisiin jatkossa paremmin.

4.5. VMware-suorituskykydata

VMwaren toimittama virtualisointialusta kerää Nokian yksityisen pilven toiminnasta runsaasti suorituskykyä kuvaavia arvoja, joita ei kuitenkaan Nokialla ole aikaisemmin hyödynnetty. Erilaisia virtuaalikoneiden toimintaa kuvaavia muuttujia löytyi tietokannasta yhteensä 19.

VMware myy työkaluja suorituskyvyn seurannan avuksi, mutta niiden on arvioitu olevan epäsopivia, erityisesti liian yksinkertaisia Nokian tarpeisiin.

Tämän tutkielman kannalta keskeistä on VMware-ohjelmiston keräämä data, jota voidaan lukea erityisen ohjelmointirajapinnan (API) avulla. API:n avulla on mahdollista lukea kaikki kerätty data, mutta suorituskykyongelmien välttämiseksi dataa voidaan lukea vain rajoitetusti. Suorituskykyongelmat osoittautuivat niin haastaviksi, että API:n käyttö ei ollut mahdollista. Käytettävissä on kuitenkin muita keinoja datan keräämiseksi.

Virtualisointialustan MS SQL –relaatiotietokannasta luodaan varmuuskopio päivittäin. Tietokanta voidaan ottaa käyttöön myös VMware-tuotteiden ulkopuolella, joten melko ajankohtaista dataa voidaan kerätä vapaasti ja analysoida häiritsemättä järjestelmän suorituskykyä. Tietokannasta on mahdollista hakea sama data, joka on haettavissa API:n avulla.

VMware on esimääritellyt suorituskykydatan keräämiselle neljä tasoa. Valittu taso sisältää myös itseään pienempiin tasoihin kuuluvat muuttujat. Taso yksi on oletustaso, joka on käytössä Nokian pilvessä. Tämän luvun jälkeen tarkastellaan vain tason yksi muuttujia.

Tason yksi suorituskykytarkkailulla kerätään vain niitä mittaustuloksia, joiden on oletettu olevan tärkeimpiä suorituskyvyn seuraamiseksi. Taso kaksi tuo lisää joitakin muuttujia ja myös se on tason yksi tapaan sopiva suorituskyvyn tarkkailuun. Tasot kolme ja neljä on tarkoitettu kehitystyön tueksi, eikä niistä ole lisähyötyä suorituskyvyn arvioinnissa [VMware]. Käytännössä tasoja kolme ja neljä voi käyttää vain lyhyen ajan, jotta tietokanta ei täyttyisi.

Suorituskykydataa tuotetaan paljon, joten sitä pitää pakata. Pakkaustasoja on neljä: viiden minuutin tarkkuus, puolen tunnin tarkkuus, kahden tunnin

tarkkuus ja yhden päivän tarkkuus. Pakkaustasojen säilytysaikoja on vastaavasti neljä: viimeisin vuorokausi, viimeisin viikko, viimeisin kuukausi ja viimeisin vuosi. Jokaiselle tarkkailutasolle on oma tietokantataulunsa, joihin on talletettu dataa virtuaalikoneiden, isäntien, resurssivarantojen ja tietokoneresurssien mittaustuloksista, joista tässä tutkielmassa tarkastellaan vain virtuaalikoneita.

4.6. Muuttujien esikarsinta

Kuten aiemmin mainittiin, laboratorioiden virtuaalikoneista kerätään yhteensä 19 muuttujaa. Osa muuttujista vaikuttaa erityisen hyödyllisiltä, kuten esimerkiksi käyttöä kuvaavat muuttujat suorittimelle, muistille, kiintolevylle ja verkolle. Analyysin kannalta on oletettavasti järkevää pyrkiä valitsemaan muuttujia ainakin jokaista mittausluokkaa kohden.

Palvelinresursseista ja niiden virtualisoinnista vastaavan henkilöstön mukaan muistiin liittyviä mittaustuloksia ei kuitenkaan kannata käyttää, sillä virtuaalikoneiden tiedetään varaavan muistia valmiiksi ilman varsinaista tarvetta. Tämän kehotuksen perusteella muistin käyttöä ei huomioida analyysissa. On mahdollista, että muistin varaamiseen liittyvien mittausten perusteella voitaisiin tehdä joitakin päätelmiä, mutta suorittimen, muistin ja levyn käyttöön liittyvien muuttujien uskotaan riittävän tämän tutkielman tarpeisiin.

VMware-dokumentaation [VMware] perusteella osa muuttujista kuvaa tiedonlouhinnan kannalta samaa ilmiötä. Esimerkiksi erilaiset suoritinkäyttöä kuvaavat muuttujat kuvaavat todennäköisesti samaa ilmiötä.

Joidenkin muuttujien määrittelytapa on melko monimutkainen. Esimerkiksi virtuaalikoneen mittaustuloksiin saattaa vaikuttaa isäntäkoneen toiminta. Tässä tutkielmassa käytettyjen muuttujien halutaan olevan ympäristön toiminnasta riippumattomia.

Suorittimen usagemhz-muuttuja perustuu isäntäkoneen laskemaan virtuaalikoneen suoritinkäyttöön. Isäntäkone tunnistaa, kuinka paljon virtuaalikone käyttää suoritinkapasiteetista mittausjaksolla ja ilmoittaa sitä vastaavan määrän megahertseinä.

Verkon käyttöön liittyvä usage-muuttuja mittaa verkkosovittimen ja virtuaalikoneen välisen verkonkäytön keskiarvon. Lähtevän ja saapuvan liikenteen arvot on yhdistetty. Muuttujan yksikkö on kilotavua sekunnissa.

Levyn käyttöön liittyvä usage-muuttuja ilmoittaa keskiarvon mittausjaksolta. Muuttujan yksikkö on kilotavua sekunnissa.

5. Datan kerääminen ja esiprosessointi

Tietokannasta kerätään edellisessä luvussa esiteltyjä suorittimen, levyn ja verkon käyttöä mittaavia arvoja. Laboratorioissa on kymmenen noodia, joista jokaiselta kerätään valittuihin mittareihin liittyviä arvoja. Vaikka eri noodien mittaustulokset kuuluvat samoihin mittausluokkiin, pidetään ne analyysia varten erillään. Näin ollen pyritään muodostamaan syötevektoreita, joilla on 30 elementtiä.

Tämän tutkielman pääpaino on SOM:issa, mutta datan keräämiseen ja esiprosessointiin panostetaan, sillä dataa sovelletaan ensimmäistä kertaa. Erilaisten muuttujien lukumäärä on melko pieni, jotta tiedonlouhintaa edeltävät vaiheet voitaisiin suorittaa riittävän hyvin. Alustavien testien perusteella valittujen muuttujien uskotaan riittävän tämän tutkielman tarpeisiin.

5.1. Tiedon valmistelu

Tiedon kerääminen aloitettiin tunnistamalla suurimpaan konfiguraatioluokkaan kuuluvat laboratoriot investointitarpeiden yhteydessä kerätystä investointidatasta. Investointidata oli kuitenkin osittain puutteellista ja epäjohdonmukaista. Ennen investointidatan hyödyntämistä, asiantuntija korjasi ja yhdenmukaisti tiedot mahdollisimman hyvin. Investointidata sisältää oleellista tietoa tulosten analysointivaihetta varten, sillä investointidatan avulla on mahdollista tietää ryhmä, johon laboratorio kuuluu.

Mittausdatan kerääminen oli välillä melko haastavaa, sillä dataa oli niin paljon. Tietokantakyselyt piti suunnitella riittävän huolella, jotta niiden ajaminen ei kestäisi liian kauan. VMware on toteuttanut SQL-kyselyjen helpottamiseksi tietokantatauluja yhdisteleviä näkymiä. Näkymien käyttämiseksi on saatavilla dokumentaatio, joka tosin on pintapuolinen. Erityisen hankalaa oli selvittää näkymien väliset viittaukset, sillä esimerkiksi ER-kaavion tapaista dokumentaatiota ei löytynyt. Varsinaisiin tauluihin ei löytynyt minkäänlaista julkista dokumentaatiota. VMware ei vastannut viesteihin, joiden tarkoituksena oli tarkemman dokumentaation saaminen.

Tulostaulusta havaittiin, että laboratorioista karsiutui suuri osa, sillä lopullisessa datajoukossa niitä oli vain 317. Jotkin laboratoriot karsiutuivat, koska niille ei ollut lainkaan mittaustuloksia. Ainakin osassa tapauksista tulos puuttuu siksi, että yksikään laboratorion virtuaalikoneista ei ole päällä.

SQL-tulostaulujen tallennus tiedostoihin ja siirtäminen järjestelmien välillä loi erityisiä haasteita. Esimerkiksi noin 2 Gt:n csv-tiedostoa oli hankala käsitellä. Mikään testattu taulukkoeditori ei pystynyt lukemaan näin suurta tiedostoa muistiin, kun käytössä oli paras saatavilla oleva tietokone (8Gt RAM-muistia,

Intel i7-2600K 3,40GHz -suoritin, Windows 7 Home Premium 64 bittinen -käyttöjärjestelmä). Tulostaulun sisältävän tiedoston käsittelyyn piti toteuttaa oma työkalu.

Työkalu luki csv-tiedostosta rivin kerrallaan ja lisäsi oleellisen datan muistiin tietorakenteeseen. Lopuksi tietorakenteen sisällön perusteella luotiin tulostaulu, joka tallennettiin uuteen tiedostoon, minkä jälkeen dataa oli mahdollista käsitellä Matlab-ympäristössä.

5.2. Mittausintervallin valitseminen

Mittausintervalliksi valitaan kaksi tuntia, minkä seurauksena dataa uskotaan olevan saatavilla sopivasti (yksi kuukausi).

Jos halutaan etsiä vuorokausittaisen työajan määrittelemää syklistä laboratorioiden käyttöä, ei analyysia voida toteuttaa yhden vuorokauden mittausintervallilla.

Jos käyttäytymistä halutaan seurata muulta jaksolta kuin kuukausi taaksepäin tietokantakopion luonnista, pitää dataa yhdistellä useammasta kannasta tai järjestelmään pitää luoda omia sääntöjä datan keräämiselle.

5.3. Puuttuvat arvot

Puuttuvia arvoja on 4,7 prosenttia. Puuttuvat arvot liittyvät lähes pelkästään noodeihin 7, 8 ja 9. VMware-tietokanta ei sisällä pelkästään suorituskyvyn mittaamiseen tarkoitettua dataa, joten tietokannan tietoja voidaan hyödyntää myös puuttuvien arvojen syyn selvittämisessä. Tietokannasta on mahdollista hakea esimerkiksi erilaisia järjestelmän luomia viestejä ja komponenttien tilatietoja.

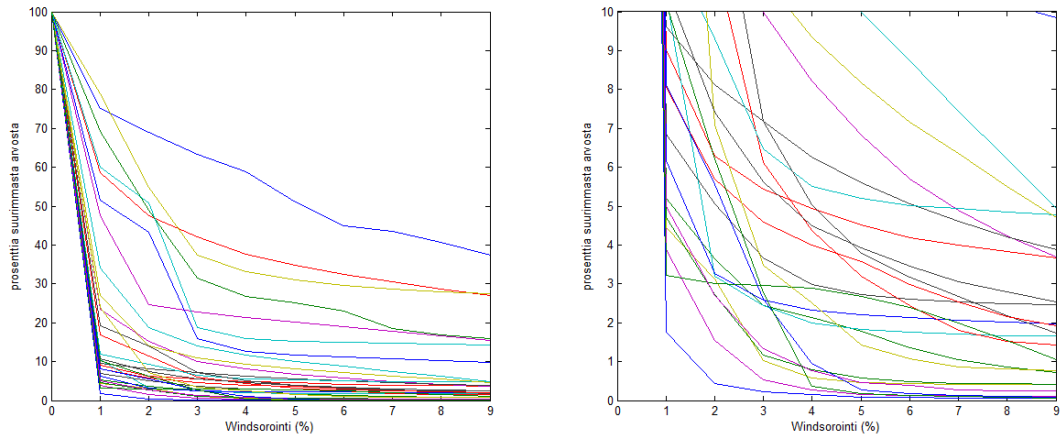
Virtuaalikoneesta ei kerätä mittaustuloksia, jos se on sammutettu. Tilatietojen perusteella virtuaalikoneita on sammutettu kuitenkin paljon enemmän kuin puuttuvien arvojen perusteella voi olettaa. Koska tässä tutkielmassa hyödynnetty data on kerätty huoltokatkon aikana tehdystä tietokannan varmuuskopiosta, päätettiin tutkia oletusta, jonka mukaan suuri osa sammutetuista virtuaalikoneista on sammutettu vain hetkellisesti.

Kun huoltokatkon lähihistoriassa sammutettujen virtuaalikoneiden oletettiin olevan käytössä suorituskykydatan keräämisen aikana, mittaustuloksen puuttumisen syyksi osoittautuu se, että virtuaalikone on sammutettu.

Näin ollen puuttuvat arvot voidaan korvata nollalla. Tosin ensiksi tarkistettiin, että ei luoda uutta klusteria pienimpien mahdollisten arvojen alapuolelle. Tarkastelujen perusteella selvisi, että nolla on varsin yleinen mittaustulos noodeilla 7, 8 ja 9.

5.4. Alustavat jakaumat

Useimmilla muuttujilla on niin suuria ääriarvoja, että esimerkiksi histogrammiesityksien resoluutio ei riitä jakaumien visuaaliseen analysointiin. Suuret ääriarvot voidaan käsitellä poistamalla tietty prosentti suurimmista arvoista ja korvata poistetut arvot suurimmalla jäljelle jääneellä arvolla. Tätä menetelmää kutsutaan toisinaan (suurien arvojen) *windsoroinniksi*. Tässä tutkielmassa Windosoroinnilla viitataan pelkkien suurimpien arvojen käsittelyyn.



Kuva 5. Windsoroinnin vaikutus jakaumien suurimpiin arvoihin.

Eri normalisointimenetelmät ovat eri tavalla herkkiä ääriarvoille. Esimerkiksi SOM:in yhteydessä usein käytetty arvojen normalisointi välille [0,1] on selvästi erittäin herkkä ääriarvoille.

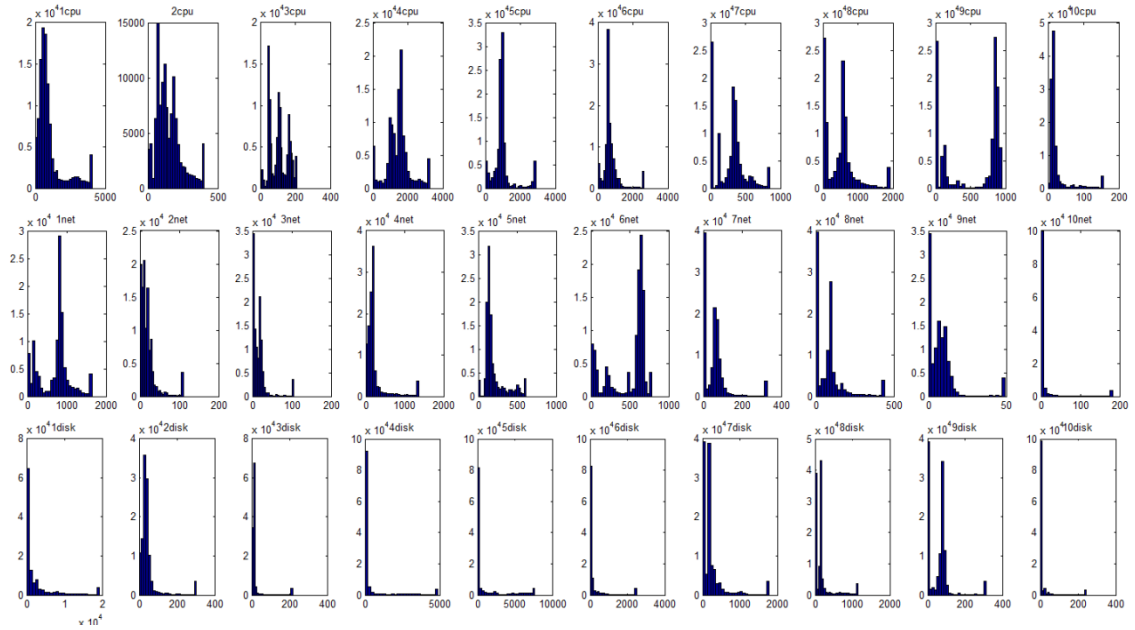
Windsorointi-prosentin merkitystä normalisoinnissa välille [0,1] voidaan havainnollistaa tarkastelemalla kuvaa 5. Lähes kaikilla muuttujilla on suuria poikkeusarvoja. Ääritapauksissa poikkeavat arvot ovat erittäin suuria, sillä esimerkiksi neljän prosentin windsoroinnilla jo seitsemän muuttujan suurin arvo on pienentynyt alle sadasosaan alkuperäisestä.

Erilaisia windsorointi-prosentteja tarkasteltaessa havaitaan, että osalla muuttujista on erityisen suuria suhteellisia ääriarvoja. Vertailemalla samaan muuttujaluokkaan kuuluvia muuttujia, voidaan arvioida tiettyyn noodiin liittyvien mittaustulosten merkitystä. Esimerkiksi kolmannen noodin levynkäyttö (3disk) on kuvan 6 perusteella suurimmillaan alle 80, vaikka levynkäyttö noodilla yksi (1disk) voi olla jopa 10 000.

On mahdollista, että kaikki muuttujat eivät ole tärkeitä minkään kiinnostavan ilmiön mallintamiseksi. Eri noodien käyttäytymismalli on kuitenkin todennäköisesti erilainen, joten muuttujan suhteellisen pieni vaihteluväli ei välttämättä tarkoita sitä, että sen mallintama ilmiö olisi vähäpätöisempi kuin jonkin

toisen noodin vastaavan muuttujan mallintama ilmiö. Näin ollen muuttujia ei painoteta tässä tutkielmassa.

Kuvasta 6 havaitaan, että osa jakaumista on approksimoitavissa eksponenttijakaumana. Joillakin jakaumilla on puolestaan useita huippuja. Yleisesti jakaumat painottuvat pieniin arvoihin. Suorittimen käyttöä ja verkon käyttöä kuvaavien muuttujien jakaumat ovat tasaisempia kuin levyn käyttöä kuvaavien muuttujien jakaumat.



Kuva 6. Jakaumat kolmen prosentin windsoroinnin jälkeen.

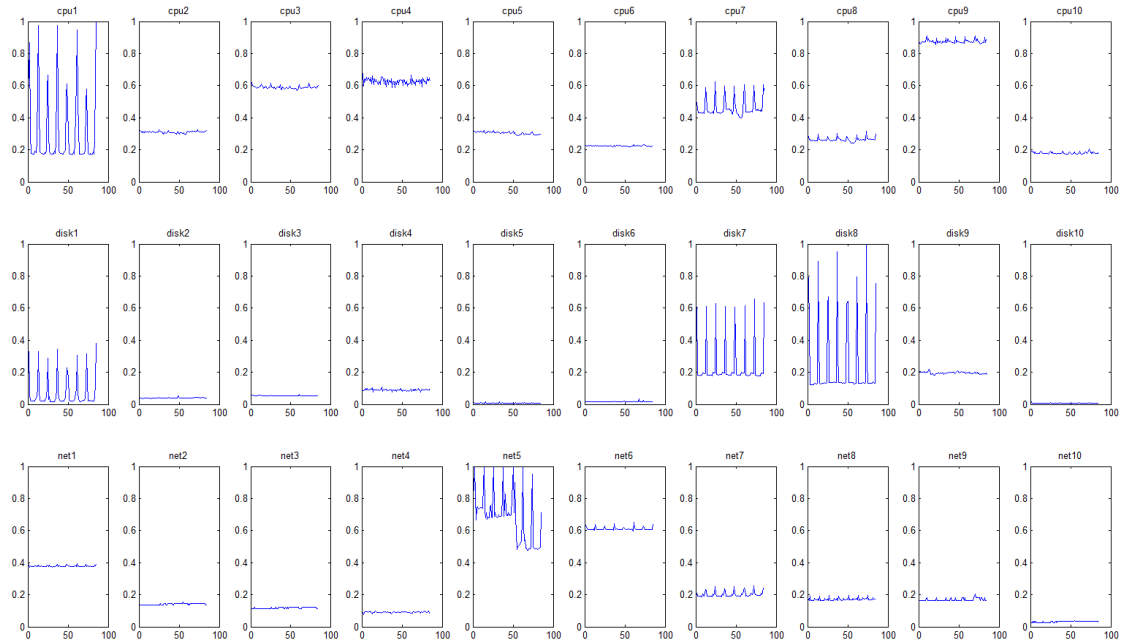
Kuvissa 7, 8, 9 ja 10 on esitetty eri laboratorioiden käyttäytymistä aikasarjoina viikon jaksolta maanantaista alkaen. Aikasarjojen perusteella useisiin laboratorioihin liittyy hetkellisen korkean käytön muodostamia piikkejä. Esimerkiksi kuvasta 7 voidaan havaita, että mittausarvot voivat esiintyä hyvin säännönmukaisesti. Lisäksi jakauman piikkien esiintymistiheys näyttäisi olevan vakio. Laajemman tarkastelun perusteella havaitaan, että jos piikkejä on havaittavissa, niin lähes poikkeuksetta ne esiintyvät kerran vuorokaudessa samaan kellonaikaan. Säännölliset piikit johtuvat todennäköisesti työajan ulkopuolella suoritettavista automaattisista prosesseista.

Automaattisten ajastettujen prosessien esiintyminen ei kuvaa käytön tarkoituksenmukaisuutta, mutta sen sijaan automaattiset prosessit saattavat häiritä analyysin tekemistä. Pelkkä automaattisten prosessien muodostama käyttö ei paranna käyttöastetta.

Jos oletetaan, että laboratorioiden käytön voimakkuus korreloi pelkästään työrytmin mukaan, pitäisi suurimman osan mittausarvoista olla jakauman pienessä päässä. Lisäksi käyttö vähenisi viikonloppuisin. Tämän lisäksi vähän käy-

tetyt laboratoriot lisäisivät pienten arvojen painotusta. Toisaalta ajastetut automaattiset prosessit lisäävät aktiivisuutta.

Laboratoriokohtaisten aikasarjojen perusteella on kuitenkin tavallisesti vaikea havaita korrelaatiota esimerkiksi viikonlopun ja mittaustulosten pienenemisen välillä. Kuvassa 8 on kuitenkin esitetty tilanne, jossa viikonloppuna mitatut tulokset vaikuttavat pienemmiltä. Erityisesti noodien 1, 4 ja 10 arvot vaikuttavat olevan pieniä. Lisäksi net2 ja net6 näyttävät olevan hieman normaalia pienempiä viikonloppuisin. Noodit 7, 8 ja 9 eivät ole lainkaan aktiivisia.

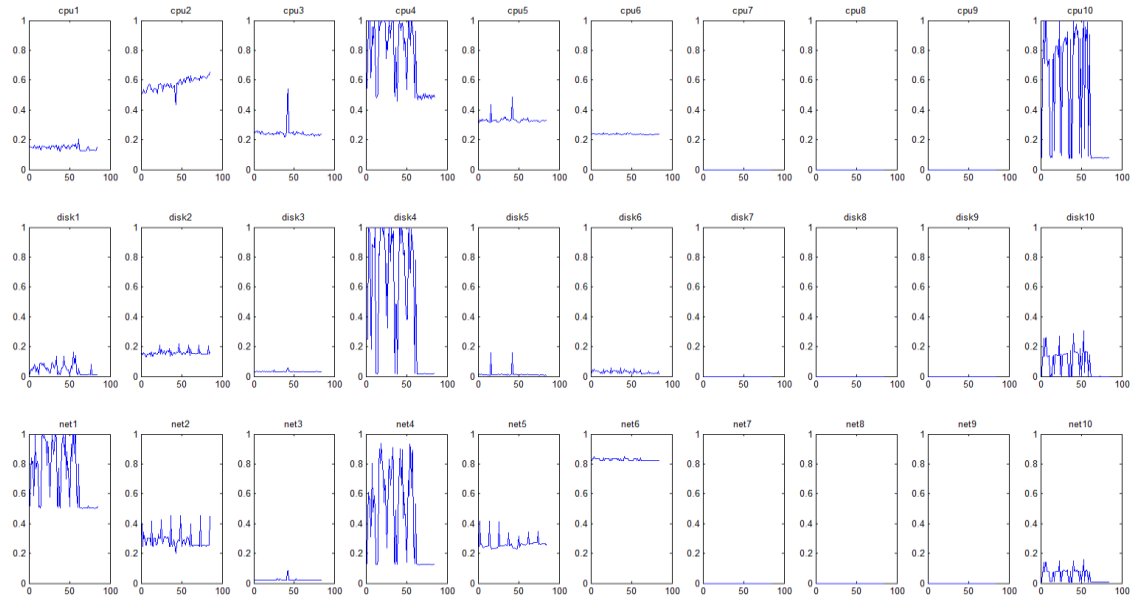


Kuva 7. Esimerkki laboratoriosta, jonka käyttäytyminen vaihtelee vähän. Säännöllisesti esiintyvät piikit kuitenkin korostuvat muuttujien cpu1, cpu7, disk1, disk7, disk8 ja net5 avulla.

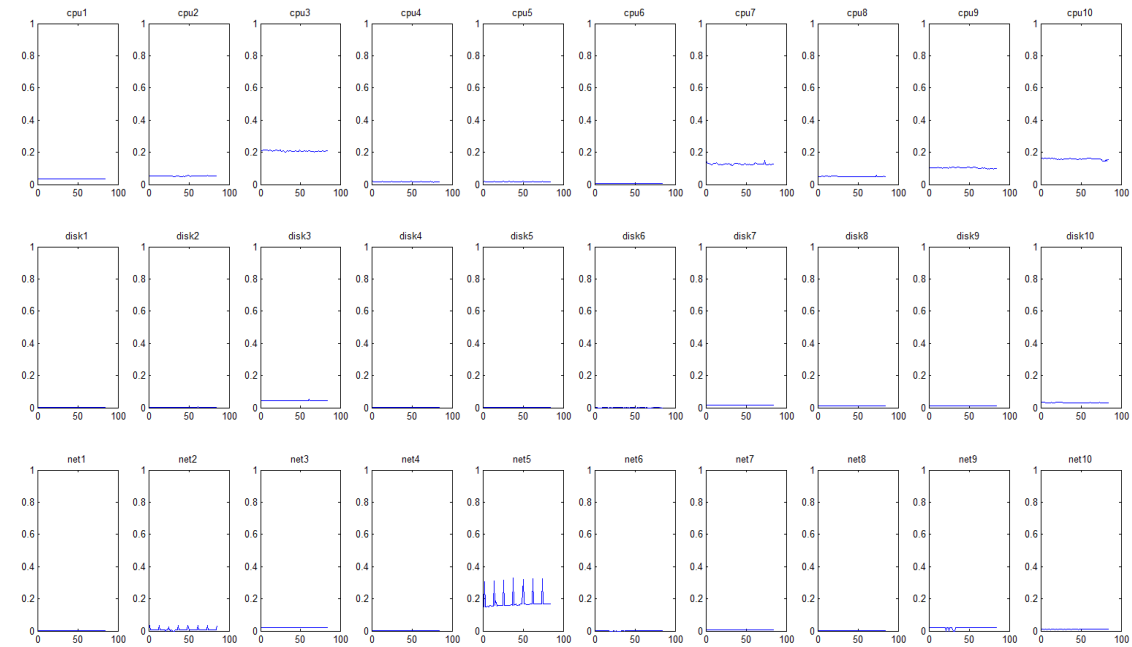
Monissa tapauksissa vaihtelu on vähäistä ja arvot ovat kaikkien muuttujien perusteella suhteellisen pieniä. Luultavasti kuvan 9 laboratorio ei ole tarkoituksenmukaisessa käytössä.

Kuvassa 10 on esitetty kaksi laboratoriota, joista mitataan keskenään lähes yhtä suuria tuloksia. Alustavasti erilaisia laboratorioiden toimintamalleja vaikuttaa olevan paljon, mutta usein kuitenkin samaan laboratorioon liittyvien eri ajankohtien mittaustulokset ovat keskenään melko samanlaisia.

Hyvien ja huonojen tilojen erottelua vaikeuttaa se, että mitään todella selvää perustilaa ei vaikuta löytyvän. Useilla laboratorioilla on paljon peräkkäisiä mittauskertoja, jotka antavat lähes samanlaisia tuloksia. Lisäksi toistaiseksi tehdyt tarkastelut eivät paljasta selvää korrelaatiota eri noodien välillä, paitsi silloin kun laboratoriolle mitataan säännöllisiä piikkejä.



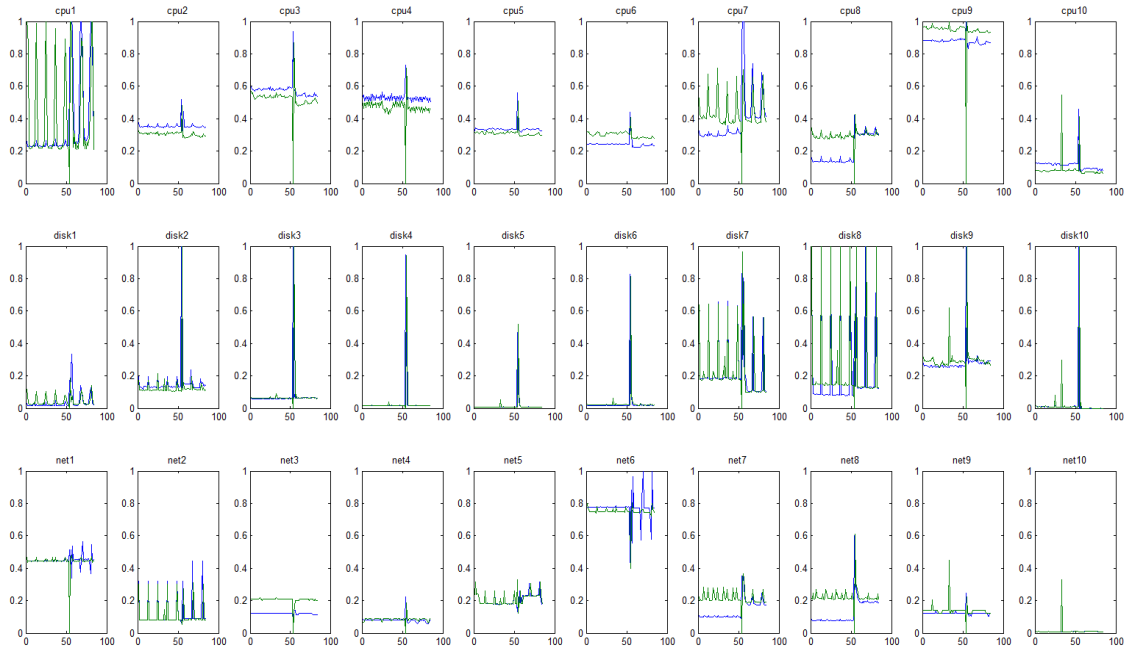
Kuva 8. Esimerkki tapauksesta, jossa lauantaina ja sunnuntaina mitatut arvot vaikuttavat pienemmiltä.



Kuva 9. Esimerkki laboratoriosta, jonka mittaustulokset ovat hyvin pieniä.

Aikasarjojen tarkastelun perusteella vaikuttaa siltä, että käyttäjien yksittäisiä toimenpiteitä on vaikea havaita datan perusteella. Valittu kahden tunnin mittausintervalli on pitkä yksittäisten käyttötapausten tunnistamiseksi. Lisäksi käyttäjästä riippumattomat taustaprosessit saattavat peittää käyttäjän aktiivisuuden jättämän jäljen. Joka tapauksessa laboratorioiden käyttäytymismallien selvittämiseen liittyy useita mielenkiintoisia kysymyksiä, joita voidaan pyrkiä selvittämään SOM:in avulla.

Kuten jo aiemmin todettiin, tilan vaihtelu vaikuttaa olevan usein melko vähäistä saman laboratorion sisällä. Korkean käyttöasteen laboratoriot saattavat vaihtaa tilaa usein, joten erilaisten tilojen lukumäärää voidaan pyrkiä selvittämään. Käytettävissä on myös tieto laboratoriota käyttävästä ryhmästä ja vaiheesta, joten on mahdollista testata, että selittääkö tietty luokka havaitun käyttäytymismallin. Jos oletetaan, että luokkiin liittyy tietty käyttäytymismalli, niin laboratorion käyttöaste saattaa olla heikko, jos se käyttäytyy alle oman luokan käyttäytymismallin.



Kuva 10. Esimerkki kahdesta laboratoriosta, joiden käyttäytymismallit vaikuttavat korreloivan positiivisesti.

6. Sopivien kartan parametrien etsiminen ja kartan soveltaminen

Edellisessä luvussa käsiteltiin datan ominaisuuksia ja analysoitiin dataa alustavasti. Monen laboratorion kohdalla huomattiin aikasarjoissa selkeitä piikkejä. Varsinaisia SOM-ajoja varten pohdittiin useita erilaisia lähestymistapoja.

Tämän luvun alussa esitellään strategia analyysin tekemiseksi, minkä jälkeen etsitään hyvät SOM-parametrit. Lopuksi karttaa sovelletaan laboratorioiden välisen korrelaation etsimiseen.

6.1. Testaus ja visualisointi

Yksi visualisointi-ideoista pohjautuu siihen, että aikasarjoja voidaan käsitellä laskemalla laboratoriolle BMU ajankohdan funktiona. Yksittäisten laboratorioiden muutosta voidaan selvittää piirtämällä *vaihtelu* (engl. *trajectory*) kartan pinnalle. Tällöin SOM:ia käytetään spatiotemporaalisena pintana. Tässä yhteydessä vaihtelulla tarkoitetaan kaarta aikasarjassa peräkkäisten syötteiden BMU:iden välillä. Lähestymistapa on vastaava kuin Simula *et al*. [1996] esittelemä lähestymistapa.

On mahdollista tutkia oletusta, jonka mukaan vähän käytettyjen laboratorioiden BMU vaihtelee vain pienen naapuruston sisällä. Vastaavasti paljon käytettyjen laboratorioiden BMU:n vaihtelu on suurempaa ja epäsäännöllisempää.

Esimerkiksi hyödyntämällä U-matriisia ja SOM:in kykyä löytää spatiaalinen järjestys, pystytään tekemään havaintoja BMU:n vaihtelun perusteella eri klusterien välillä.

Lähestymistavan tulokset ovat melko informatiivisia, sillä tuloksista näkyy laboratorion tilojen vaihtelu aikasarjana, minkä lisäksi samaa karttaa olisi mahdollista käyttää myös tavalliseen tapaan datan ulottuvuuksien vähentämiseen ja piirteiden havaitsemiseen.

6.2. SOM Matlab-ympäristössä

Matlab tarjoaa joustavan ympäristön matemaattisten ongelmien käsittelyyn. Tässä tutkielmassa on hyödynnetty Matlabia erityisesti ohjelmointi- ja visualisointityökaluna.

SOM Toolbox [SOM Toolbox] on ilmainen Matlab-kirjasto, joka tarjoaa paljon erilaisia työkaluja SOM-analyysien tekemiseksi. Suuri osa toteutuksista liittyy peruskartan käsittelyyn, mutta esimerkiksi joitakin SOM-variaatioita sisältyy kirjastoon. SOM Toolbox käyttää oletusarvoisesti eräajoon perustuvaa algoritmia. Yksi Matlabin heikkouksista on laskentaympäristön hitaus, joten eräajon merkitys korostuu Matlab-ympäristössä. Eräajo antaa alustavien testien perusteella vähintäänkin melko hyviä tuloksia.

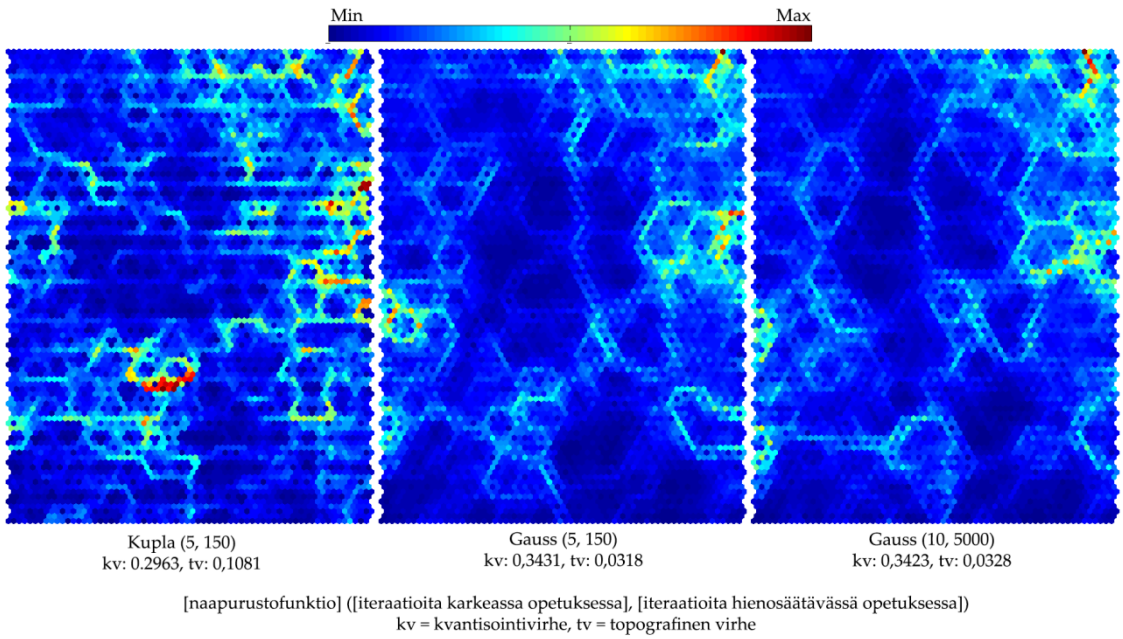
SOM-työkalut on mahdollista ottaa helposti käyttöön, sillä SOM Toolbox käyttää oletusarvoisesti tiettyjä lähestymistapoja algoritmin parametrien valitsemiseksi. SOM-parametreihin liittyviä valintoja on kuitenkin syytä pohtia ja testata tarkemmin.

6.3. Alustavat testit

Alustavat testiparametrit valitaan karkeasti arvioitujen optimaalisten arvojen läheisyydestä. Kuten aiemmin todettiin, parametrien valitsemiseksi on olemassa korkeintaan suuntaa antavia menetelmiä. Lopulliset tulokset riippuvat voimakkaasti käytettävästä datasta, joten järkevä lähestymistapa hyvien SOM-parametrien löytämiseksi on suorittaa alustavia testejä.

Joitakin parametreja voidaan kuitenkin kiinnittää heti. Kartan solmujen muodoksi valitaan kuusikulmio, sillä sen avulla naapurustojen käsitteleminen on luonnollisimmillaan. Karttapintana käytetään kaksiulotteista tasoa sylinterin tai toroidin pinnan sijaan, sillä data ei ole syklistä. Kaksiulotteinen taso on ihanteellinen visualisoinnin kannalta.

Naapurustosäteenä käytetään SOM Toolboxin oletusasetusta. Aloitussäteeksi asetetaan pisimmän sivun neljännes, mistä se vähenee karkean opetuksen aikana neljäsosaan, ellei neljäsosa ole alle yhden. Hienosäätövaihe jatkaa siitä mihin karkea opetus jäi. Myös hienosäätövaiheen aikana säde on aina vähintään yksi.



Kuva 11. Eri naapurustofunktioiden ja opetuspituuksien vaikutus klusteroitumiseen.

Kun käytettiin eräajoalgoritmia, yksikään käytetyistä matemaattisista tai visuaalisista validointitekniikoista ei kannustanut käyttämään pitkää opetusta. Ajoihin käytetyllä tietokoneella (8Gt RAM-muistia, Intel i7-2600K 3,40GHz -suoritin, Windows 7 Home Premium 64 bittinen -käyttöjärjestelmä) pisimmät opetusajot kestivät noin 15 tuntia. Lyhyt opetus kesti tyypillisesti vain hieman yli minuutin. Hienosäätävä opetus oli aina vähintään kymmenen kertaa suurempi kuin karkea opetus.

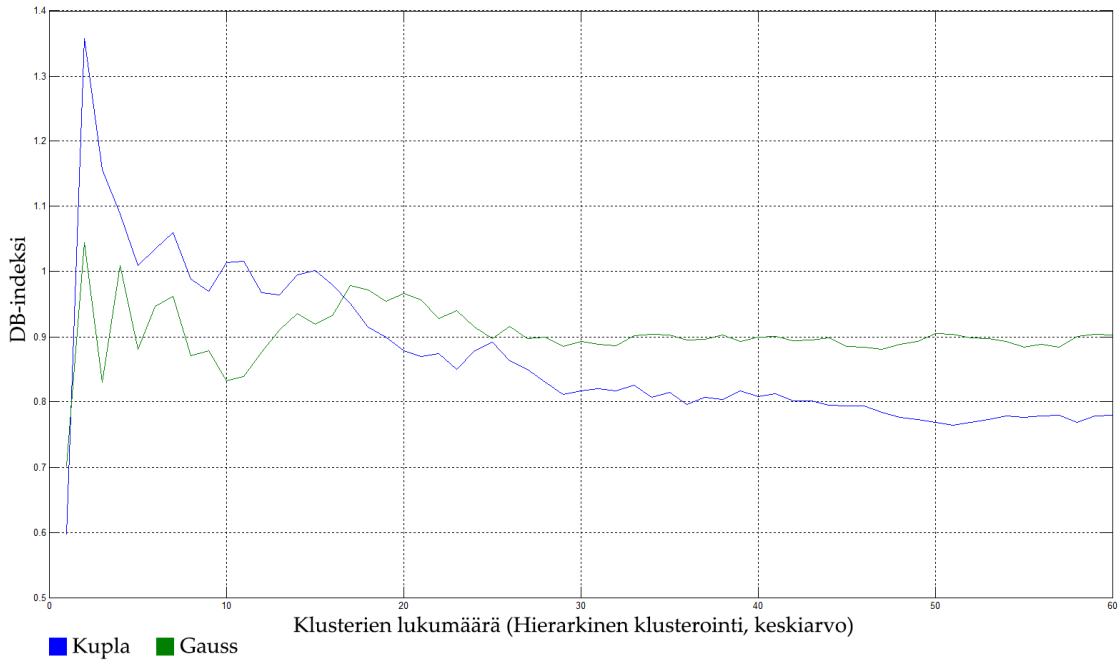
Solmujen lukumäärän valitsemiseen ei ole tarkkoja sääntöjä, mutta Kohosen [2013] mukaan usein kannattaa luoda niin suuri kartta kuin laskennallisesti on mahdollista. Solmujen lukumäärä kannattaa kuitenkin perustella testiajojen tulosten perusteella. Käytetään solmujen lukumäärän lähtötason valitsemiseksi usein käytettyä kaavaa $5\sqrt{n}$, missä n on opetussyötteiden lukumäärä solmujen valitsemiseksi.

Kvantisointivirhe oli tyypillisesti vain hieman pienempi pitkille ajoille. Esimerkiksi kuvan 11 tapauksessa oikeanpuoleisen kartan kvantisointivirhe on vain 0,2 prosenttia pienempi kuin keskimmäisen kartan kvantisointivirhe, vaikka hienosäätövaiheita suoritettiin yli 30 kertaa enemmän. Naapurustofunktion valinnalla havaittiin suurempi vaikutus kvantisoinnin laatuun, sillä esimerkiksi vasemmanpuoleisen kartan kvantisointivirhe on jo 13,4 prosenttia pienempi kuin oikeanpuoleisen kartan kvantisointivirhe.

Opetuksen pituuteen liittyvät parametrit voidaan kiinnittää, sillä niiden kasvattamisesta ei vaikuta olevan suurta hyötyä enää valittuja parametriarvoja suuremmilla arvoilla. Jatkossa karkeita vaiheita ajetaan kymmenen ja hienosäätövaiheita ajetaan 150.

Naapurustofunktioista testataan kuplafunktiota ja Gaussin funktiota, sillä ne ovat ääritapauksia. Näillä valinnoilla korostetaan naapurustofunktion merkitystä.

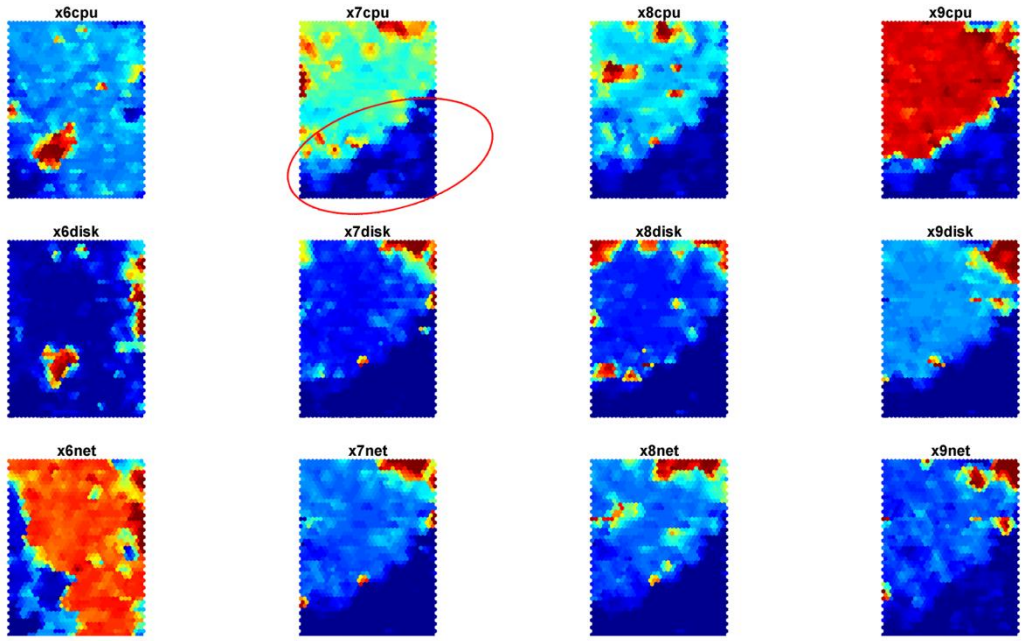
Kuplanaapuruston vaikutus näkyy selvästi kuvan 11 vasemman kartan U-matriisissa, jossa klusterien reunat näyttävät olevan hyvin voimakkaita. Topografinen virhe on kymmenen prosenttia, joten se on yli kolme kertaa suurempi kuin muilla kartoilla. Koulutus on pakottanut kartan vastaamaan syöteavaruutta melko hyvin, mutta samalla kartan yleistämiskyky on heikentynyt. Vasemman kartan voidaan todeta ylioppineen syöteavaruuden piirteet.



Kuva 12. DB-indeksi laskettuna hierarkisen klusteroinnin perusteella.

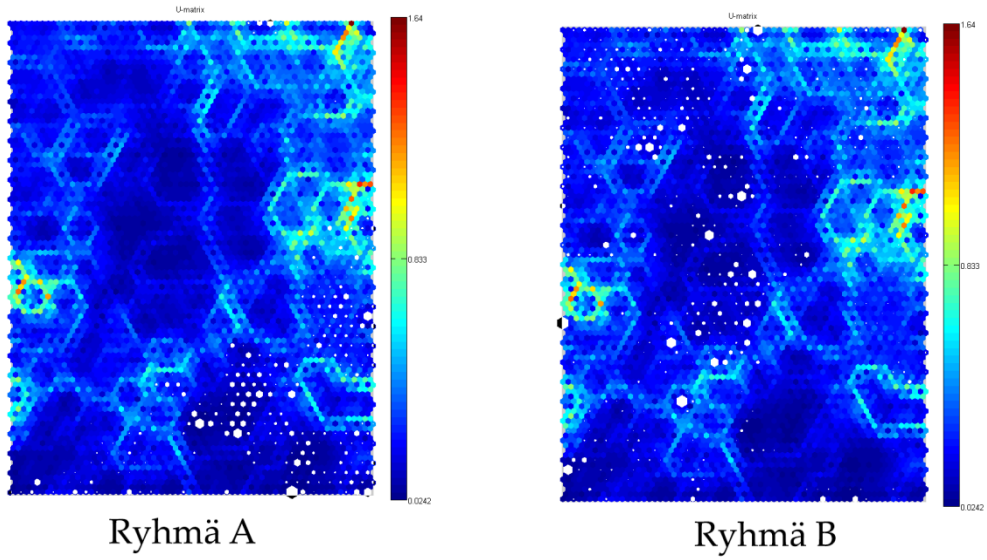
Tosin tulkinta ylioppimisesta on sovelluskohtainen, sillä tavoitteena saattaa olla esimerkiksi yleisten tilojen korostaminen. Kuvassa 12 klusterien yhdistämisen kriteerinä on käytetty pienintä keskiarvojen etäisyyttä. Molemmat SOM:it saavat pienimmän Davies-Bouldin-indeksin (DB-indeksi) klusterien lukumäärällä 2.

DB-indeksin perusteellaa Gauss-SOM:in klusterit ovat laadukkaampia kuin kupla-SOM:in klusterit, kun klustereita on alle 17, mutta kun klustereita on 17 tai enemmän, niin kupla-SOM:in klusterointi on laadukkaampi. Kun klustereita on korkeintaan 60, kupla-SOM:in klusterointi on parhaimmillaan 51 klusterilla.



Kuva 13. Kupla-SOM, jossa oikean alareunan klusteri on korostettu. Sama klusteri esiintyy myös kaikissa noodien 7, 8 ja 9 komponenteissa.

Kuvan 13 komponenttiesityksistä voidaan havaita, että noodien 7-9 oikeassa alareunassa on suuri klusteri, jossa arvot ovat aina pieniä. Noodin 6 komponentit edustavat muita komponentteja ja noodeja, joissa ei esiinny oikean alareunan klusteria.

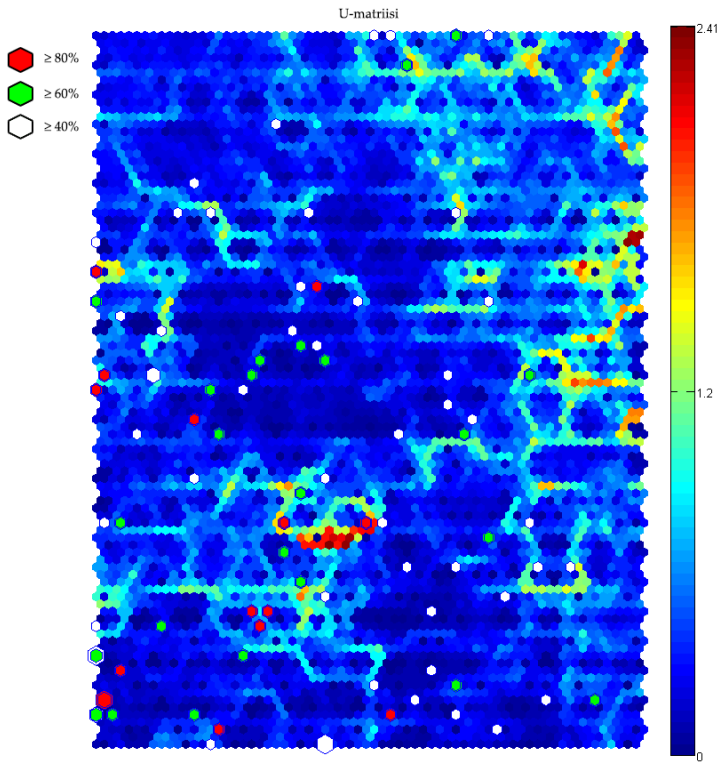


Kuva 14. Kahden eri ryhmän laboratorioiden BMU:t osumahistogrammeina.

Lisäksi kuvan 13 perusteella vahvistuu aiempi havainto ylioppimisesta. Kaikilla komponenttipinnoilla näkyy paljon saarekkeita, mikä viittaa heikkoon yleistämiskykyyn. Kartta on lisäksi mahdollisesti liian suuri.

Kuvasta 14 voidaan havaita, että vasemmanpuoleisessa esityksessä osumat ovat kartan oikeassa alareunassa eli kuvan 13 korostaman klusterin sisällä. Oikeanpuoleisessa U-matriisissa osumat ovat jakautuneet tasaisemmin, mutta osumia ei juuri ole oikean alakulman klusterissa. Sama ilmiö toistuu myös monen muun ryhmän kohdalla. Oikean alakulman klusteri korostuu voimakkaasti analyysin tuloksessa, joten noodit 7, 8 ja 9 kannattaa poistaa analyysin seuraavasta vaiheesta.

Kuvassa 15 on esitetty laboratorioiden muutosta mittausjakson aikana. Jotkin laboratoriot pysyvät jopa 80 % ajasta samassa noodissa. Laboratorioiden yleiset tilat ovat jakautuneet melko tasaisesti kartalle. Vasemmassa alakulmassa on neuroneita, joihin liittyy tavallista enemmän osumia. Merkin koko viittaa vastaavaan noodiin liittyvien osumien lukumäärään.

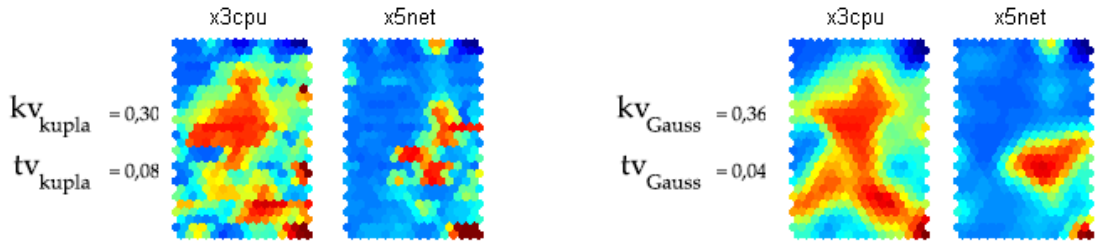


Kuva 15. Karttaan on korostettu ne neuronit, joihin liittyy paljon saman laboratorion osumia.

6.4. Toinen testauskierros

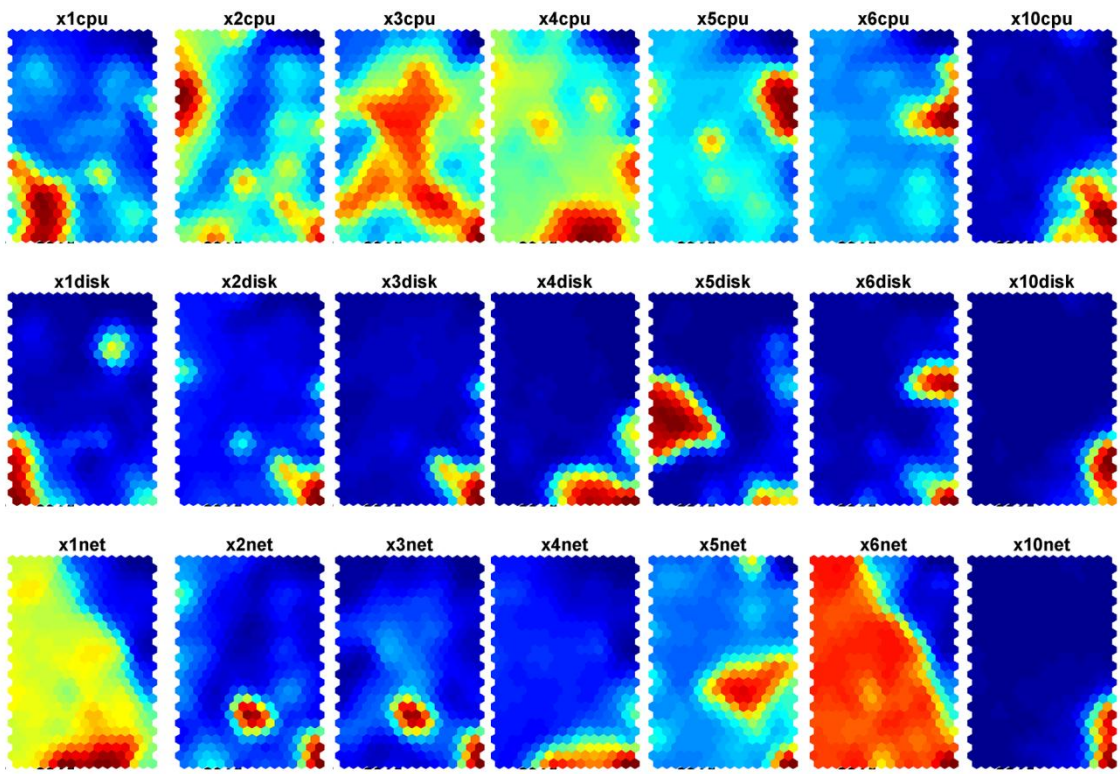
Ensimmäisissä testeissä tehtiin mahdollisesti liian suuria karttoja, joten kokeillaan seuraavaksi pienempiä karttoja. Myös noodit 7, 8 ja 9 kannattaa poistaa analyysista.

Kun tehdään 75 % pienempiä karttoja ilman noodeja 7,8 ja 9, niin havaitaan, että kuplanaapurustoa käyttämällä ei saada miellyttäviä tuloksia. Monien komponenttien pinnat ovat saarekkeiden täyttämiä. Kuplanaapuruston käyttäminen ei vaikuta mielekkäältä.



Kuva 16. Kuplanaapuruston käyttäminen tuottaa karkeamman järjestyksen kuin Gaussin naapuruston käyttäminen.

Kuvasta 16 voidaan havaita, että Gaussin-naapurustoa ja pieniä karttoja käyttämällä löydetään miellyttäviä karttoja. Käyttämällä pientä Gaussin-karttaa monet komponentit koostuvat pääsääntöisesti enää vain muutamista isoista klustereista. Tämän seurauksena esimerkiksi komponenttien välisen korrelaation havaitseminen helpottuu.



Kuva 17. Paras SOM-kartta toisen testikierroksen perusteella.

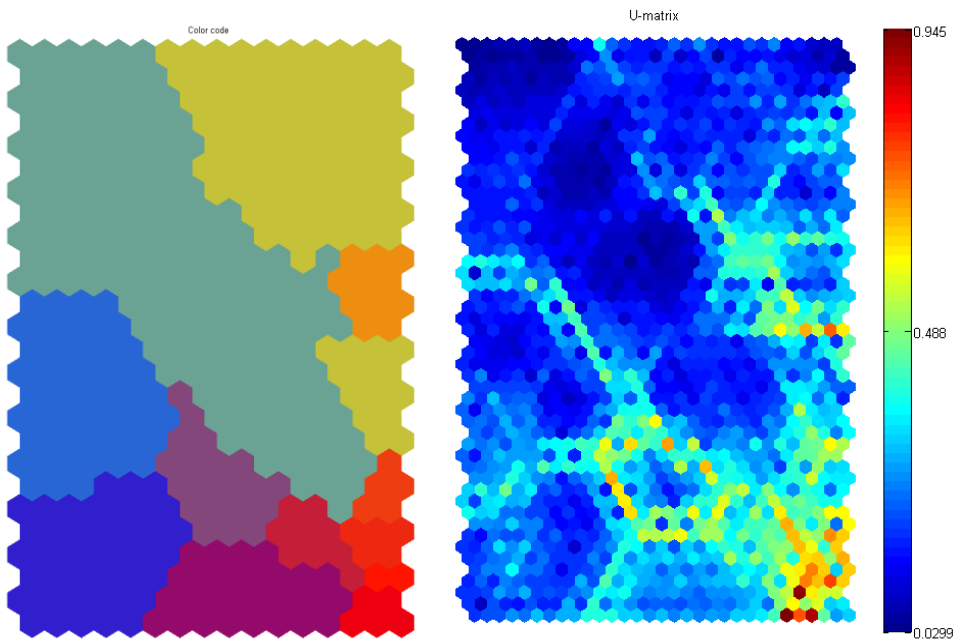
Kuvassa 17 esitetty kartta näyttää hyvältä. Spatiaalinen järjestys vaikuttaa löytyvän. Oikeassa yläkulmassa näyttäisi sijaitsevan kaikkien pienimpiä mittaustuloksia vastaavat mallit, sillä kaikki komponentit ovat siellä sinisiä. Pääsääntöisesti suuria mittaustuloksia vastaavat klusterit sijoittuvat kartan alaosaan.

Kartasta on helppo havaita eri komponenttien väliset suhteet. Esimerkiksi komponentit 2net ja 3net korreloivat keskenään.

Kuvasta 17 havaitaan, että laboratorioiden tilat ovat paremmin vertailtavissa sen jälkeen kun noodeista 7-9 luovuttiin. Tietysti myös kartan koon muuttamisella on vaikutusta. Samasta esityksestä nähdään myös U-matriisit, joiden perusteella kartta jakautuu melko hyvin klustereihin.

6.5. Havaintojen analysointi

Toisen testikierroksen aikana löydettiin hyvä SOM-kartta, jonka visualisoinnin tulkitseminen on mielekästä. U-matriisin perusteella kartta on myös erotettavissa selviin klustereihin. Yksi mahdollinen klusterointi on esitetty kuvassa 18. Pohditaan kuvan 18 klusterointia vielä hieman tarkemmin.



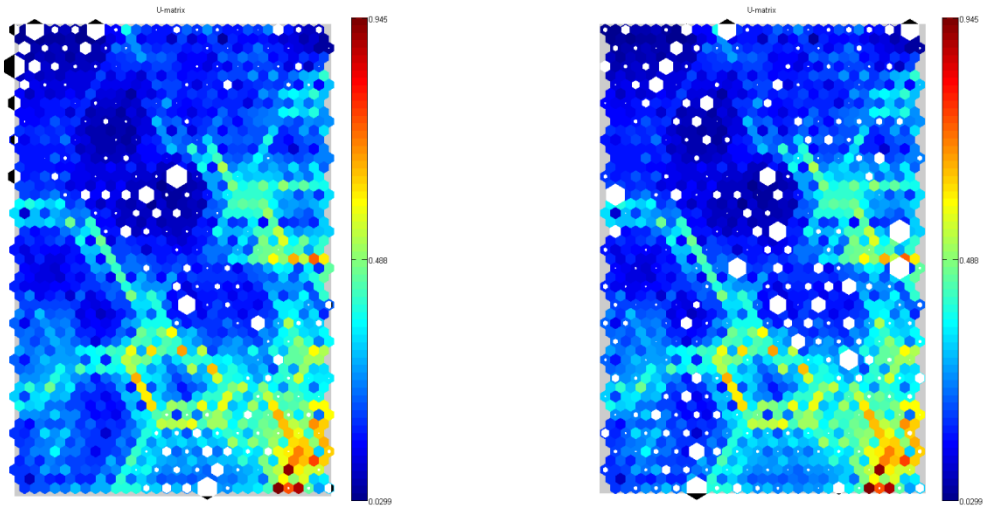
Kuva 18. Parhaaseen DB-indeksiin perustuva täyden kytkennän (engl. *complete linkage*) avulla tehty klusterointi.

Oikean yläkulman klusteriin liittyy pieniä arvoja, mikä selviää kuvan 17 komponenttiesityksen avulla. Tosin 5cpu ja 6cpu–muuttujiin liittyy suuria arvoja pienintä käyttöä vastaavan klusterin alueella. Muuttujien 1net ja 6net välillä näkyy korrelaatiota, ainakin pienten arvojen osalta. Suurin tummanvihreä

alue vastaa suurta osaa hahmoavaruudesta, sillä klusteri on suuri. Esimerkiksi kuvan 19 ryhmien osumista suuri osa sijoittuu suurimman klusterin alueelle.

Näiden tulosten perusteella on mahdollista rakentaa esityksiä, joilla järjestelmän toimintaa voi seurata. Järjestelmä on monimutkainen ja sen kanssa työskentelee paljon ihmisiä, joten data-analyysin avulla voi mahdollisesti luoda pohjaa erilaisille keskusteluille.

Alustavien tulosten perusteella SOM:in Matlab-toteutus sopii hyvin EDA:n tarkoituksiin. Matlabin avulla SOM:in sisäistä esitystä on jatkuvasti mahdollista tutkia ja analysoida lisää.



Ryhmä A Ryhmä B

Kuva 19. Kahden eri ryhmän laboratorioiden BMU:t osumahistogrammeina (Vastaavat ryhmät kuin kuvassa 14).

6.6. Laboratorioiden erilaiset käyttäytymismallit

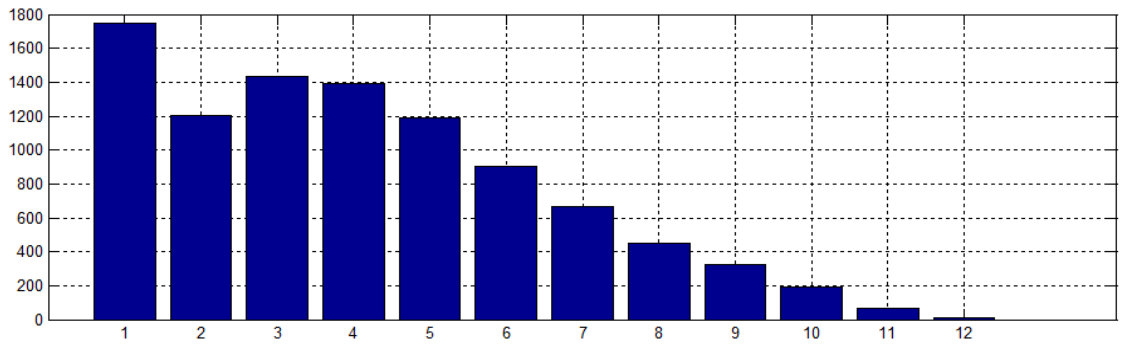
Esiprosessoinnin yhteydessä esiteltiin aikasarjojen visualisointeja. Joidenkin aikasarjojen avulla on mahdollista havaita selkeitä tapauksia, joissa lähes kaikkien komponenttien arvot ovat poikkeuksellisen pieniä. Kuvan 9 aikasarjaa käytettiin esimerkkinä tällaisista matalista mittaustuloksista.

Selkeät pieneen käyttöasteeseen viittaavat käyttäytymismallit ovat kuitenkin melko harvinaisia. Tyypillisesti useilta noodeilta mitataan melko tyypillisiä arvoja, joita ei voi selvästi yhdistää vähäiseen käyttöön.

Käyttöasteen mittaamisen kannalta on oleellista tunnistaa erilaisia laboratorioihin liittyviä toimintamalleja. Aiemmat SOM-visualisoinnit vahvistavat ennakkokäsitystä, jonka mukaan laboratorioilla on paljon erilaisia toimintamalleja.

Monen laboratorion toimintamalliin liittyy vähäinen tilojen vaihtelu, mikä havaitaan tarkastelemalla laboratorioihin liittyviä uniikkeja BMU:ita.

Kuvan 20 perusteella on oltava olemassa mielekkäässä käytössä olevia laboratorioita, joihin kuitenkin liittyy vain melko pientä tilojen vaihtelua. Jos tällaisten laboratorioiden tyypillisten tilojen joukko muuttuu selvästi, niin käyttöaste on saattanut heikentyä. Tätä havaintoa voidaan hyödyntää esimerkiksi ohjattuun oppimiseen perustuvan analyysin toteuttamiseksi. Esiprosessoinnin yhteydessä tehtyjen aikasarjatarkastelujen perusteella havaittiin, että monen laboratorion tapauksessa tyypilliset tilat esiintyvät tasaisesti koko tarkkailujakson ajan. Esimerkiksi säännölliset piikit esiintyvät monissa tapauksissa kerran vuorokaudessa tiettyinä vuorokaudenaikoina.



Kuva 20. Uniikkien BMU:iden lukumäärä vuorokaudessa.

Esitellään seuraavaksi menetelmä, jonka avulla arvioidaan kahden laboratorion käyttäytymismallien samanlaisuutta. Menetelmän yhteydessä SOM:ia käytetään spatiotemporaalisena pintana.

Laboratorioihin liittyy aikasarjamaisia syötejonoja, joille voidaan määritellä BMU-jono. Kahden laboratorion BMU-jonoja vertailemalla laboratorioiden samanlaisuutta voidaan arvioida. Käytetyn menetelmän perusteella laboratoriot vastaavat toisiaan täydellisesti, jos molempien laboratorioiden BMU-jonoissa esiintyy sama lukumäärä samoja alkioita. Alkioiden järjestyksellä ei ole merkitystä.

Jos $(a_k)_{k=1}^n$ on syötejono, jossa a_k on syöte k ja n on syötteiden lukumäärä, niin vastaava BMU-jono on $(b_k)_{k=1}^n$, $b_k = bmu(a_k)$.

Vastaavuuden laskemiseksi vertaillaan kahta laboratoriota, joita vastaavat BMU-jonot b ja b'. Olkoon neuronien (ja BMU:iden) joukko $\mathrm{U} = \{1,2,\dots,m\}$. Laboratorioiden syötejonojen pituudet ovat vastaavasti n ja n'. Lasketaan vastaavuus

$$v = \frac{2}{n+n'} \sum_{i=1}^{m} min\,\{lkm(\mathrm{U}_i, b), lkm(\mathrm{U}_i, b')\},$$

jossa lkm laskee esiintymien U_i lukumäärän BMU-jonoissa b ja b'. Jos BMU-jonojen vertailussa neuronia U_i vastaa $x \leq y$ esiintymää, niin vastaavuuden laskemisessa huomioidaan vain $2x$ esiintymää. Vastaavuus $0 \leq v \leq 1$ on täydellistä, jos $v = 1$. Vastaavuutta ei ole lainkaan, jos $v = 0$. Vastaavuus voi olla täydellistä, vaikka jonojen b ja b' alkioiden järjestys olisi eri.

Sovelletaan seuraavaksi vastaavuuden laskentatapaa kaikkien laboratorioiden ja tietyn luokan laboratorioiden välillä.

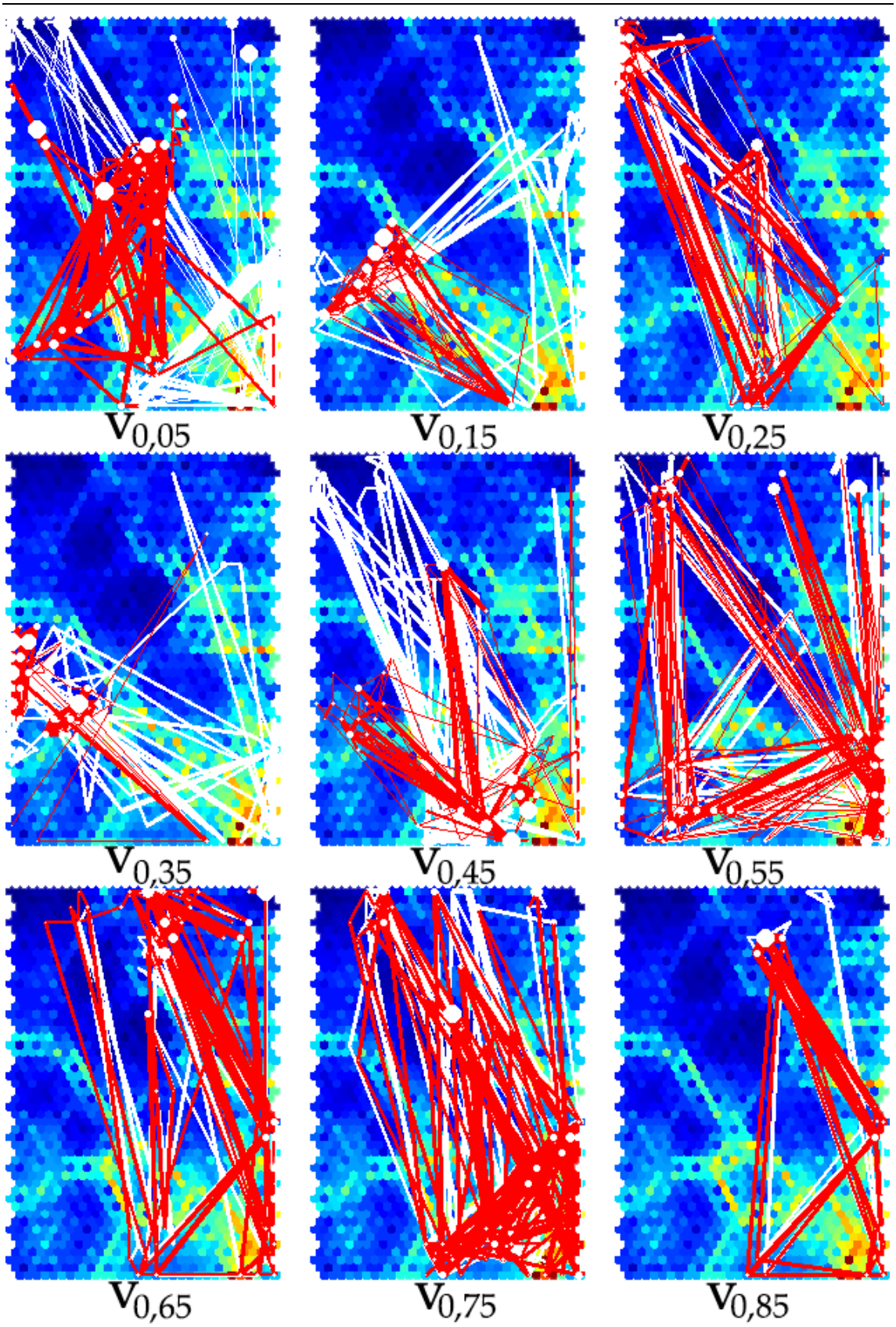
6.6.1. Vastaavuus kaikkien laboratorioiden välillä

Lasketaan aluksi vastaavuus kaikkien laboratorioiden välille. Kuvassa 21 on esitetty satunnaisesti valittuja laboratoriopareja tiettyjen vastaavuusarvojen läheisyydestä. Tehdään seuraavaksi muutamia kuvaan 21 liittyviä havaintoja.

Parin $v_{0,05}$ pieni vastaavuus korostuu selvästi, sillä valkoisella laboratoriolla on paljon osumia kartan vasemmassa yläkulmassa ja oikeassa alakulmassa, mutta punaisen laboratorion osumista suuri osa on lähellä kartan keskustaa ja vasenta alakulmaa. Lisäksi peräkkäisten BMU:iden määrittämät punaiset ja valkoiset kaaret leikkaavat usein toisensa. Laboratorioilla on osumia kuitenkin samassa naapurustossa lähellä kartan alareunaa.

Parin $v_{0,15}$ osumista suuri osa sijoittuu vasemman reunan puolessavälissä sijaitsevaan klusteriin. Laboratorioiden osumista suuri osa näyttää osuvan toistensa läheisyyteen.

Parista $v_{0,55}$ lähtien laboratorioiden käyttäytymismalli vaikuttaa erittäin samanlaiselta. Monet kaaret ovat yhdensuuntaisia laboratorioiden välillä.



Kuva 21. Esimerkkejä erilaisista vastaavuusarvoista.

6.6.2. Vastaavuus saman luokan laboratorioiden välillä

Arvioidaan seuraavaksi laboratorioiden keskinäistä vastaavuutta luokkakohtaisesti. Käytetään kolmea luokittelukriteeriä: ryhmä, vaihe ja ryhmän ja vaiheen kombinaatiot (kombinaatioluokittelu).

Vähintään yhden laboratorion omistavia ryhmiä on 37 kappaletta. Enimmillään yhteen ryhmään kuuluu 31 laboratoriota. Erilaisista vaiheista on muodostettu kuusi ryhmää. Kuudenteen ryhmään on yhdistetty kaikki harvinaiset ryhmät.

Taulukosta 1 havaitaan, että laboratoriot ovat jakautuneet epätasaisesti eri vaiheiden ja ryhmien kesken. Yhteensä 222 kombinatioluokasta vain 50 (22,5 %) sisältää vähintään kaksi laboratoriota, mikä on vähimmäismäärä vastaavuusvertailun tekemiseksi. Määritellään, että luokan laboratoriot ovat vertailukelpoisia, jos luokkaan kuuluu vähintään kaksi laboratoriota.

Kaikkien laboratorioiden välisten kombinaatioiden vastaavuusvertailun keskiarvo on 0,042, mitä voidaan käyttää vertailuarvona. Samaan kombinaatioluokkaan kuuluvien laboratorioiden välillä ei ole aina vahvaa vastaavuutta, sillä vertailukelpoisista kombinaatioluokista 13:n keskiarvo on pienempi kuin vertailuarvo.

Joillekin kombinaatioluokille lasketut tulokset ovat kuitenkin selvästi yli vertailuarvon, sillä esimerkiksi 15 kombinaatioluokan vastaavuuden keskiarvo on yli 0,15 ja seitsemän kombinaatioluokan vastaavuuden keskiarvo on yli 0,30. Kuitenkin kombinaatioluokkien frekvenssit ovat usein niin pieniä, että tilastollisesti merkitsevän päättelyn tekeminen ei ole mahdollista.

Lisäksi kymmenen ryhmän välillä, mukaan lukien kaksi suurinta ryhmää, vastaavuusarvo on suurin eri vaiheisiin kuuluvien laboratorioiden välillä. Esimerkiksi ryhmän neljä vaiheella kaksi on suurin frekvenssi (20) ja paras vastaavuus (0,365), mutta koko ryhmän paras vastaavuus on kuitenkin suurempi kuin vaiheen neljä vastaavuus (0,479).

Suurimpiin vaiheisiin kuuluu huomattavasti enemmän laboratorioita kuin yhteenkään kombinaatioluokkaan, joten ei ole yllättävää, että neljän vaiheen vastaavuusarvo on suurempi kuin yhdenkään niihin liittyvän kombinaatioluokan vastaavuusarvo.

	Vaihe 1			Vaihe 2			Vaihe 3			Vaihe 4			Vaihe 5			Vaihe 6			Kaikki vaiheet		
Ryhmä	#	ka	max	#	ka	max	#	ka	max	#	ka	max	#	ka	max	#	ka	max	#	ka	max
4	7	0,051	0,214	20	0,034	0,365	0			0			3	0,119	0,357	1			31	0,043	0,479
10	5	0,040	0,398	2	0,000	0,000	1			22	0,152	0,721	1			0			31	0,086	0,784
1	6	0,156	0,438	19	0,162	0,719	0			0			2	0,096	0,096	0			27	0,163	0,719
6	2	0,313	0,313	11	0,033	0,210	0			2	0,167	0,167	4	0,002	0,010	2	0,047	0,047	21	0,036	0,365
18	3	0,131	0,208	14	0,041	0,411	0			3	0,249	0,359	0			1			21	0,063	0,411
2	12	0,334	0,758	5	0,366	0,607	0			0			0			0			17	0,312	0,758
17	6	0,088	0,366	6	0,048	0,261	2	0,030	0,030	1			2	0,081	0,081	0			17	0,071	0,646
27	2	0,023	0,023	1			0			14	0,058	0,633	0			0			17	0,056	0,633
22	3	0,053	0,097	6	0,009	0,117	0			0			7	0,049	0,385	0			16	0,041	0,711
14	2	0,305	0,305	7	0,082	0,469	0			0			2	0,471	0,471	0			11	0,093	0,471
13	0			10	0,058	0,398	0			0			0			0			10	0,058	0,398
5	0			8	0,059	0,357	1			0			0			0			9	0,047	0,357
9	0			8	0,035	0,206	0			0			1			0			9	0,030	0,206
16	0			7	0,068	0,352	2	0,188	0,188	0			0			0			9	0,092	0,352
7	0			7	0,046	0,156	0			0			0			1			8	0,050	0,159
8	0			6	0,138	0,417	0			0			1			0			7	0,131	0,417
31	0			6	0,150	0,578	0			1			0			0			7	0,178	0,839
12	4	0,558	0,833	1			1			0			0			0			6	0,234	0,833
15	0			5	0,106	0,372	0			0			1			0			6	0,122	0,438
28	0			2	0,000	0,000	0			0			3	0,089	0,267	0			5	0,030	0,267
30	2	0,084	0,084	0			2	0,163	0,163	0			1			0			5	0,156	0,918
36	0			0			0			5	0,156	0,249	0			0			5	0,156	0,249
11	0			3	0,110	0,286	0			0			0			0			3	0,110	0,286
25	1			2	0,302	0,302	0			0			0			0			3	0,101	0,302
3	0			0			1			0			0			1			2	0,023	0,023
19	0			2	0,031	0,031	0			0			0			0			2	0,031	0,031
20	0			2	0,023	0,023	0			0			0			0			2	0,023	0,023
21	0			1			0			0			0			0			1		
24	0			0			1			0			0			0			1		
26	0			1			0			0			0			0			1		
29	0			1			0			0			0			0			1		
32	0			1			0			0			0			0			1		
33	0			1			0			0			0			0			1		
34	0			0			0			1			0			0			1		
35	0			0			1			0			0			0			1		
37	0			1			0			0			0			0			1		
38	0			0			0			0			0			1			1		
Kaikki	55	0,071	0,858	166	0,039	0,849	12	0,045	0,432	49	0,081	0,721	28	0,028	0,471	7	0,033	0,201	317	0,042	0,918

Taulukko 1. Vastaavuus saman luokan laboratorioiden välillä.

7. Yhteenveto

Tämän tutkielman tavoitteena oli selvittää mahdollisuuksia Nokia Networksin laskentaresurssien käyttöasteen seuraamiseksi. Laskentaresursseista riippuvat laboratoriot on virtualisoitu, minkä todettiin helpottavan käyttöä kuvaavien tunnuslukujen keräämistä. Nokia ei ole aiemmin hyödyntänyt laboratorioiden suorituskykyä kuvaavaa dataa, joten data piti hakea tietokannasta ja esiprosessoida.

Datan hakemisen todettiin olevan haastavaa, sillä datan kysely hidasti liikaa tuotantoympäristöä. Tietokannan varmuuskopioista oli kuitenkin mahdollista etsiä tarvittava data, vaikka datan suuri määrä aiheutti lisähaasteita.

Käyttöasteen mittaamiseksi on usein toivottu analysointityökaluja, kuten esimerkiksi työkalua, joka luokittelisi laboratoriot hyviin ja heikkoihin. Laboratorio on hyvä vain silloin, kun sen käyttöaste on hyvä. Laboratoriot ovat kuitenkin hyvin monimutkaisia järjestelmiä ja niillä todettiin olevan paljon erilaisia käyttötarkoituksia, joten niiden käytön mielekkyyden arviointi todettiin vaikeaksi.

Koska tässä tutkielmassa käytettyä dataa ei ollut aiemmin käsitelty ja laboratorioiden oletettiin noudattavan monia eri toimintamalleja, päätettiin toteuttaa eksploratiivinen data-analyysi.

Tutkielman päämenetelmäksi valittiin SOM, jonka uskottiin sopivan hyvin tutkielman sovelluskohteeseen. SOM:ista on olemassa Matlab-ympäristöön tehty toteutus, joten algoritmia ei ollut pakko toteuttaa itse. Lisäksi Matlab-ympäristössä oli mahdollista toteuttaa skriptejä, joiden avulla SOM:in tuloksia jatkokäsiteltiin ja -analysoitiin.

SOM:ia on tutkittu paljon ja se on toiminut hyvin monissa eri sovelluskohteissa eri tieteenaloilla. SOM:in todettiin olevan neurolaskentamenetelmä, sillä se pohjautuu aivokuoren toimintaan. SOM:in käyttöön todettiin liittyvän paljon yksityiskohtia, jotka pitää ymmärtää. Menetelmän havaittiin kuitenkin olevan melko joustava yksityiskohtien suhteen, sillä jos yksityiskohdat on huomioitu edes kohtuullisen hyvin, niin kilpailevaan oppimiseen perustuva SOM pystyy löytämään hahmoavaruuden tärkeimmät piirteet. SOM sopeutuu aivokuoren toimintojen tavoin hyvin erilaisiin tunnistustehtäviin.

SOM:ia testattiin muutamia kertoja alustavasti, ennen kuin lopullisen kartan asetukset löytyivät. SOM pystyi lopulta löytämään miellyttävän esityksen hahmoavaruudesta, jonka avulla on mahdollista tutkia eri komponenttien välistä korrelaatiota. Myös spatiaalinen järjestys on mahdollista havaita SOM-visualisointien avulla.

Laboratorioiden vastaavuutta mitattiin laskemalla samojen BMU:iden lukumäärä. Visualisointien avulla osoitettiin, että käytetty vastaavuutta mittaava menetelmä pystyi tunnistamaan samanlaiset laboratoriot melko hyvin. Joidenkin tapausten kohdalla havaittiin, että ryhmän ja vaiheen muodostamilla luokittelukriteereillä on mahdollista selittää toimintamallia.

Vastaavuuden mittaamiseksi käytettyä menetelmää on mahdollista jatkokehittää, sillä tässä tutkielmassa käytetyn vastaavuutta mittaavan menetelmän todettiin olevan melko joustamaton. Menetelmä hyväksyy vain täsmälleen samat BMU:t vastaavuusvertailuun, eikä esimerkiksi lähinaapuruston osumia huomioida, vaikka visualisoinnin perusteella monien laboratorioiden välillä oli paljon yhteisiä osumia tietyssä kartan osassa.

Tutkielma jätti vielä paljon kysymyksiä auki, mutta jonkinlainen ymmärrys kohteen toiminnasta pystyttiin muodostamaan. Jo esiprosessoinnin yhteydessä tuli ilmi, että käyttäjien yksittäisten toimintojen muodostamia jälkiä on todennäköisesti vaikea tunnistaa tutkielmassa käytetystä datasta. Kun SOM:ia käytettiin spatiotemporaalisena pintana, laboratorioihin havaittiin liittyvän vain muutama uniikki BMU päivässä.

Viiteluettelo

[Amari, 1980] Shun-ichi Amari, Topographic organization of nerve fields. *Bulletin of Mathematical Biology* **42**, (1980), 339–364.

[Arbib, 1987] Michael A. Arbib, Levels of modelling of mechanisms of visually guided behavior. *The Behavioral and Brain Sciences* **10,** (1987), 407--465.

[Bengio, 2009] Yoshua Bengio, Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning* **2**, 1 (Jan. 2009), 1-127.

[Collobert and Bengio, 2004] Ronan Collobert and Samy Bengio, Links between Perceptrons, MLPs and SVMs. In**:** *21th International Conference on Machine Learning*, (2004), 23-30.

[Cortes and Vapnik, 1995] Corinna Cortes and Vladimir Vapnik, Support-Vector Networks. *Machine Learning* **20,** (1995), 273-297.

[Cottrell *et al.*, 1997] Marie Cottrell, Jean-Claude Fort and Gilles Pagés, Theoretical aspects of the SOM algorithm. In**:** *Proceedings of the WSOM 97, Workshop on Self-Organizing Maps,* Espoo, Finland, (Jun., 1997), 246–267.

[Fritzke, 1994] Bernd Fritzke, Growing cell structures—a self-organizing network for unsupervised and supervised learning. *Neural Networks* **7**, (1994), 1441–1460.

[Haykin, 1994] Simon Haykin, *Neural Networks. A Comperenhensive Foundation.* Prentice Hall, (1994).

[Heino, 2010] Petteri Heino, *Pilvipalvelut*. Talentum Media Oy.

[Hubel and Wiesel, 1962] David Hubel and Torsten Wiesel, Receptive fields, binocular and functional architecture in the cat's visual cortex. *Journal of Physiology* **160,** (Jan. 1962), 106–154.

[Kaski and Kohonen, 1996] Samuel Kaski and Teuvo Kohonen, Explotary data analysis by the self-organizing map: structures of welfare and poverty in the world. In**:** *Neural Networks in Financial Engineering. Proceedings of the Third International Conference on Neural Networks in the Capital Markets,* (1996), 498-507.

[Kohonen, 1995] Teuvo Kohonen, Emergence of invariant-feature detectors in the adaptive-subspace self-organizing map. *Biological Cybernetics* **75,** (Nov. 1996), 281-291.

[Kohonen, 2013] Teuvo Kohonen, Essentials of the self-organizing map**,** *Neural Networks* **37**, (Jan. 2013), 52–65.

[Malsburg, 1973] Christopher von der Malsburg, Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik* **14**, (1973), 85–100.

[Minsky, 1961] Marvin Minsky, Steps toward artificial intelligence. *Proceedings of the Institute of Radio Engineers* **49**, (1961), 8-30.

[Mountcastle, 1957] Vernon Mountcastle, Modality and topographic properties of single neurons of cat's somatic sensory cortex. *Journal of Neurophysiology* **20**, (Mar. 1957), 408–434.

[Rosenblatt, 1962] Frank Rosenblatt, *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms.* Spartan, Washington DC (1962).

[Rumelhart *et al.,* 1986] David E. Rumelhart, Paul Smolensky and James L. McClelland, Learning internal representations by error propagation. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* **1**, (1986), 318–362.

[Simula *et al.,* 1996] Olli Simula, Esa Alhoniemi, Jaakko Hollmen, and Juha Vesanto, Monitoring and modeling of complex processes using hierarchical self-organizing maps. In**:** *International Symposium on Circuits and Systems, ISCAS '96., IEEE*, (May, 1996), 73-76.

[SOM Toolbox] SOM Toolbox Team
Available at: http://www.cis.hut.fi/projects/somtoolbox/documentation/, http://www.cis.hut.fi/projects/somtoolbox/package/papers/techrep.pdf.

[VMware] VMware vSphere 5.1 Documentation, PerformanceManager. Available at: https://www.vmware.com/support/developer/vc-sdk/visdk400pubs/ReferenceGuide/vim.PerformanceManager.html