MAANPUOLUSTUSKORKEAKOULU

TEKOÄLY JA AUTOMAATIO TULEVAISUUDEN LAIVASTOJOUKOISSA

Pro gradu -tutkielma

Yliluutnantti
Lauri Vasankari

Sotatieteiden maisterikurssi 10
Merisotalinja

Maaliskuu 2022

Kurssi	Linja
Sotatieteiden maisterikurssi 10	Merisotalinja
Tekijä Yliluutnantti Lauri Vasankari	
Opinnäytetyön nimi **Tekoäly ja automaatio tulevaisuuden laivastojoukoissa**	
Oppiaine, johon työ liittyy Sotatekniikka	Säilytyspaikka Maanpuolustuskorkeakoulun kirjasto
Aika Maaliskuu 2022	**Tekstisivuja** 80 **Liitesivuja** 71

TIIVISTELMÄ

Tekoäly, miehittämättömät järjestelmät ja näiden autonomia eri asteineen ovat kasvavia teknologian trendejä niin siviili- kuin sotilasteollisuudessa. Tekoälytutkimuksen 2010-luvulla tekemien harppauksien mahdollistaessa edelleen monimutkaisempien järjestelmien kehittämistä ja käyttöönottoa tekoälyn tieteenala tunnustetaan todennäköisesti disruptiiviseksi teknologiaksi asevoimien toimintaympäristöissä jo lähitulevaisuudessa. Puolustusvoimilla on strateginen intressi tekoälykykyjen seurantaan ja kehittämiseen.

Tämä opinnäytetyö tarkastelee tekoälyn kehitystä ja sen lähitulevaisuuden mahdollisuuksia Suomen laivastojoukkojen näkökulmasta, kategorioituna OODA-prosessin mukaisiin osa-alueisiin. Työssä tarkastellaan teoreettisesti ja kokeellisesti nykyteknologialla mahdollisia tapoja tuottaa ja kehittää tekoälysovelluksia Suomen laivaston käyttöön nykyisten ongelmatilanteiden ratkaisemiseksi ja ratkaisuun liittyvän suorituskyvyn kehittämiseksi.

Tutkimuksen päämenetelmä on pehmeä systeemimetodologia, jolla hahmotetaan laivastojoukkojen systeemiä ja siitä löydettäviä toimintoja, joiden tehostamiseen tai muuttamiseen voidaan hyödyntää tekoälyä ja automaatiota. Tutkimuksen aineisto on kerätty kirjallisuuskatsauksena ja laadullisella analyysillä tekoälyn ja siihen pohjaavien menetelmien ja järjestelmien osalta. Tekoälyn ja koneoppimisen tarkastelun ja kokeelliseen mallinnukseen liittyen menetelminä ovat tekoälyn matemaattiset teoriat sekä algoritmit ja niiden testaus käytännössä julkisilla data-aineistoilla.

Tutkimuksen johtopäätöksinä tekoälyn hyödyntäminen kapeissa, ihmistä avustavissa, sovelluksissa on mahdollista nopealla aikataululla muun muassa sensoridatan tulkinnassa ja tilannekuvan luomisessa. Suurimpina haasteina ovat kerätyn datan laajuus, laatu ja käytettävyys toimivien tekoälymallien tuottamiseksi. Niiltä osin, kun kaupalliset ja sotilaalliset intressit ovat yhteneviä, kuten merenkulussa, voidaan laivastojoukoissa hyödyntää kaupallisia järjestelmiä tekoälyn ja järjestelmäautonomian käyttöönotossa, mutta Suomen merivoimien tulee kehittää kykyä ja valmiuksia koostaa hyödyllisiä data-aineistoja ja asiantuntijajärjestelmiä tekoälyn käyttöönoton mahdollistamiseksi. Tekoälyn osaamista, kehityksen ja käyttöönoton koordinointia sekä yhteistyötä tutkimuksen ja kehitystyön piirissä tulee kehittää tukemaan päätöksentekoa tulevaisuuden hankkeita, uusia suorituskykyjä ja toimintamalleja silmällä pitäen.

AVAINSANAT

tekoäly, automaatio, autonomia, koneoppiminen, syväoppiminen, laivasto, merivoimat, systeemi

SISÄLLYSLUETTELO

TEKOÄLY JA AUTOMAATIO TULEVAISUUDEN LAIVASTOJOUKOISSA

1. Tutkimusperusteet

1.1 Johdanto, tutkimuksen aihe ja tutkimustilanne

Tekoälyn, koneoppimisen ja autonomisten järjestelmien kiihtyvä kehitystyö ja käyttöönotto sekä näiden seurannaisena uudet uhkakuvat luovat mahdollisuuksia ja tarpeita muokata laivastojoukkojen toimintatapoja ja -menetelmiä nyt ja lähitulevaisuudessa. Vuoden 2021 puolustusselonteko toteaa, että ”*teknologinen kehitys erityisesti digitalisaation, tekoälyn, koneautonomian, sensoriteknologioiden sekä uusien operatiivisten toimintaympäristöjen osalta vaikuttaa ratkaisevasti sotilaallisten suorituskykyjen kehittämiseen*” [1]. Suomen puolustusministeriön ”*Strategic Guidelines for Developing AI Solutions*” linjausten mukaisesti Suomen puolustusvoimien täytyy, uskottavan puolustuksen ylläpitämiseksi, pystyä kehittämään tekoäly- ja digitalisaatiokykyjä. [2 s. 1] Raportissa todetaan samalla, että niin kutsutut tekoälyn implementaatiot ovat toistaiseksi erittäin kapean alan sovelluksia, jotka suoriutuvat vain yksittäisistä tarkkaan rajatuista tehtävistä. Nykyiset tekoälysovellukset soveltuvat parhaiten sellaisiin tehtäviin, joissa vaaditaan suurta prosessointikapasiteettia tai halutaan poissulkea ihmisen aiheuttamat muuttujat prosessoinnissa. AI-ROADMAP Tutkimuskokonaisuuden *Mitä AI on*? loppuraportissa [3] kartoitetaan kattavasti tekoälyn käsitteistöä, ennusteita tulevaisuuden kehitykselle ja käyttöönotolle sekä tulevaisuuden sotilassovelluksia. Puolustusvoimien tutkimuslaitoksen Tekoälyn tiekartta - raportin suositeltuja toimenpiteitä Suomen puolustusvoimien tekoälyosaamisen ja -kykyjen kehittämiseksi ovat muun muassa jatkotutkimuksen käynnistäminen tekoälyn edellytyksenä olevien tietomassojen muodostamiseksi ja rakentamiseksi, tavoista ja menetelmistä tekoälyn opettamiseksi sekä osaajien painopisteen muodostamiseen tietoaineistojen käsittelykykyyn sekä tekoälyn opettamiseen. Tämän lisäksi suositellaan välittömänä toimenpiteenä tekoälysovellusten ja -tuotteiden tuntemuksen laajentamista vertailevana kehittämisenä sekä jatkotutkimusta tekoälyteknologioiden kehitysennusteista tukemaan muutosaskeleita. [4 s. 3-4] Tämä opinnäytetyö pyrkii vastaamaan osaltaan näihin tavoitteisiin laivastojoukkojen näkökulmasta.

Tällä hetkellä Suomen laivastojoukkojen lähitulevaisuudennäkymät määrittää käynnissä oleva Laivue 2020-hanke, jonka tarkoitus on päivittää Merivoimien suorituskykyjä laivastojoukkojen osalta ja taata Merivoimien operatiivinen toiminta seuraavien vuosikymmenien ajaksi korvaamalla yhteensä seitsemän poistuvaa tai jo poistunutta taistelualusta, joihin kuuluvat Pohjanmaa-luokan miinalaiva, Hämeenmaa-luokan miinalaivat sekä Rauma-luokan ohjusveneet. Laivue 2020 on tämän tutkimuksen näkökulmasta konventionaalisesti suunniteltu, täysmiehitetty alusluokka, joka tuo Merivoimien käyttöön vaatimusmäärittelyn mukaisesti suorituskykyjä niin valvonnan kuin ilma-, pinta- ja vedenalaisen sodankäynnin vaatimuksiin lisäten merkittävästi Suomen merivoimien kykyä täyttää

lakisääteiset tehtävänsä, joita ovat merialueiden valvonta ja alueloukkausten torjuminen, meriyhteyksien turvaaminen sekä merellisten hyökkäysten torjunta. [5 s. 2; 6; 7]

Entinen Merivoimien tutkimusjohtaja Jan Brunberg kirjoittaa Tuleva Sota 3 kirjassa olevassa julkaisussa 2030-luvulta alkaen merisotaa leimaavan kolme trendiä, joita ovat sensoriverkoston kyvykkyyden kasvu, miehittämättömien ja autonomisten järjestelmien yleistyminen sekä aseiden kantaman, tarkkuuden ja nopeuden kasvu. Tekstissä visioidaan tulevaisuudessa Itämerta valvovan huomattavasti nykyistä kyvykkäämpi ja verkottuneempi sensoriverkosto, jotka yhdistetään miehittämättömiin järjestelmiin näiden arkipäiväistyessä. Miehittämättömien järjestelmien ennakoidaan olevan merkittävä osa yksiköiden kykyä ulottaa valvontaa ja vähentää itseensä kohdistuvaa riskiä sekä operoimaan pidempiä ajanjaksoja. Autonomiasta todetaan sen lisääntyvän ja mahdollistavan miehittämättömien järjestelmien itsenäisiä tehtäviä, mahdollisesti osana parvea tai ryhmää. [8 s. 128-131] Tekoäly, automaatio ja miehittämättömät autonomiset alusyksiköt ovat vääjäämättömältä vaikuttava kehitystrendi, jonka hyödyntämistä on tarkasteltava kriittisesti. Alueellisesti ja kansallisten resurssien puitteissa automaation ja autonomisten tai miehittämättömien järjestelmien käyttöönotto on verrattain haastavaa huomioiden Suomen merialueen erityispiirteet kuten saariston ja matalat vedet, paikallisesti suuren määrän kauppa- ja huvialusliikennettä sekä suhteellisesti pienen valtion resurssit kehitystyöhön. Kuitenkin tarkastelua ja mahdollisia applikaatioita puoltaa muun muassa kotimainen ja pohjoismainen kehitystyö sekä tekoälyn ja automaation mahdollistamat hyödyt Merivoimien tehtävien toteuttamisessa. Esimerkiksi pohjoismaisittain pitkällä autonomisten alusten kehityksessä ollut AAWA-projekti (*Advanced Autonomous Waterborne Applications*) hyödynsi kehitystyössä suomalaisia yliopistoja ja toteutti tekoälyä hyödyntävää koetoimintaa Saaristomerellä. [9]

Tämän lisäksi puolustusvoimien henkilöstöresurssien vaje [10] sekä sodanajan suorituskyvyn nojaaminen muun muassa merenkulun pätevyyksien osalta korkeasti koulutettuihin, vuosien kokemuksen omaaviin ammattilaisiin [11] tekee materiaalisesti kriisiaikana korvaamattomista alusyksiköistä myös henkilöstöresurssillisesti korvaamattomia ja monistamattomia. Tekoäly sekä automaattiset ja autonomiset järjestelmät mahdollistavat uudenlaisen ulottuvuuden suorituskykyihin, jossa ammattitaitoisen henkilöstön osaamista voidaan tehostaa, monistaa ja projisoida nopeasti sekä laaja-alaisesti.

1.2 Tutkimuksen tavoite, tutkimuskysymykset ja rajaukset

Tutkimuksen kirjoitushetkellä Suomen merivoimissa ei ole olemassa autonomisten järjestelmien tai tekoälyn konseptia. Tämän tutkimuksen tavoitteena on alustukseksi kartoittaa tekoälyn ja autonomisten järjestelmien kehitystä, maturiteettia ja mahdollisuuksia, muodostaen käsitys nykytilanteesta ja lähitulevaisuudesta. Tämän jälkeen arvioidaan tekoälyn eri menetelmien käytettävyyttä sotilaallisiin sovellusalueisiin kategorioittain pehmeän systeemiajattelun metodologialla. Kokeellisesti on tarkoitus havainnollistaa Merivoimien nykytilassa keräämän datan käyttökelpoisuutta kapeiden tekoälysovellusten mallintamiseen alueellisen valvonnan tukemisessa ja tilannekuvan muodostamisessa.

Tutkimuksen on tarkoitus tuottaa käsitys laivastojoukkoja koskevasta tekoälykehityksestä lähitulevaisuudessa eli 2030-luvun alkupuolelle asti, sekä kartoittaa Suomen merivoimien valmiutta ja kehityskohteita tekoälyn kouluttamiseen vaaditun datan keräämisessä ja analysoimisessa.

Tutkimuskysymys eli tutkimuksen pääkysymys on:
Miten automaatiota ja tekoälyä voidaan hyödyntää Suomen laivastojoukoissa nyt ja lähitulevaisuudessa?
Alakysymyksiä ovat:

1. Mitä tarkoitetaan tekoälyllä, automaatiolla ja autonomialla?
2. Mitä mahdollisuuksia Suomen laivastojoukoissa on tekoälyn ja automaation käyttöönotolle?
3. Mitä haasteita Suomen laivastossa kohdistuu tekoälyn käyttöönottoon?
4. Mitä toimenpiteitä toimivien tekoälysovellusten tuottaminen ja käyttöönotto edellyttää Merivoimilta?

Tutkimus on rajattu koskemaan tekoälyä hyödyntävien järjestelmien mahdollisuuksia ja haasteita teknologiana. Tutkimus ei käsittele tekoälyä eettiseltä tai moraaliselta kannalta päätöksenteon tai sotilaallisen voimankäytön näkökulmista. Käsiteltävät menetelmien kategoriat on rajattu koskemaan toimintoja teknisinä suoritteina, kuten tilannekuvan muodostamista tai miehittämättömän aluksen autonomista operointia. Koska esimerkiksi tekoälyn määritelmään liittyy erilaisia näkemyksiä ja käsitteitä, määritetään keskeisimpien käsitteiden rajaukset tämän opinnäytetyön osalta niitä koskevissa alaluvuissa erikseen.

1.3 Tutkimusmenetelmät ja tutkimuksen rakenne

Tutkimus on laadullinen tutkimus, jonka tuloksia validoidaan liitteissä raportoiduilla määrällisillä osioilla. Opinnäytetyössä tarkastellaan kirjallisuuskatsauksen menetelmin tekoälystä ja autonomisista järjestelmistä, pääosin merelliseen ympäristöön liittyen, tehtyjä tutkimuksia ja kirjallisuutta sekä arvioidaan Suomen laivastojoukkojen ja toimintaympäristön asettamia vaatimuksia ja mahdollisuuksia sovittaa tekoälyä ja autonomisia järjestelmiä niiden käyttöön. Kirjallisuuskatsauksen perusteella muodostetaan pehmeän systeemiajattelun metodologialla käsitys erilaisten tekoälymenetelmien hyödyntämisestä sotilaallisten ongelmien ratkaisussa neljässä eri kategoriassa. Työssä tehtävät kokeelliset mallinnukset erinäisistä nykymenetelmillä toteutettavista, kapeista tekoälysovelluksista Suomen merivoimien keräämällä datalla toteutetaan hyödyntämällä CRISP-DM (”*cross-industry standard process for data mining*”) prosessia, jonka mukaisesti muodostetaan käsitys Merivoimien tilannekuvan luomisen ympäristöstä ja tavoitteista sekä kerätystä valvontadatasta, käsitellen kyseinen data hyödynnettävään muotoon ja mallintaen tämän datan perusteella kapeita tekoälysovelluksia arvioiden niiden toimivuutta ja mahdollista käyttöönottoa [12 s. 10].

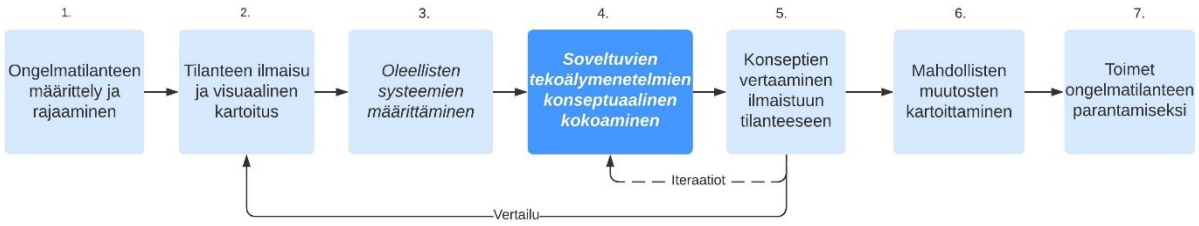
Tutkimuksessa käytettävä systeemiajattelu on epistemologia, jossa todellisuutta hahmotetaan systeemeinä. Nämä systeemit koostuvat inhimillisen toiminnan tarkastelun osalta neljästä perusajatuksesta: ilmaantuvuudesta, hierarkiasta, kommunikaatiosta ja kontrollista. Luonnollisten ja suunniteltujen systeemien osalta tarkastelun tärkeimmät ominaisuudet ovat ilmaantuvuusominaisuudet, sillä systeemiajattelun ilmaantuvuusperiaatteen mukaan täydet entiteetit sisältävät ominaisuuksia, jotka ovat merkityksellisiä ainoastaan osana kokonaisuutta. Pehmeä systeemiajattelu (”*Soft Systems Thinking*”) on muodostunut vastaamaan kompleksisten ongelmien hahmottamiseen ja mallintamiseen sellaisissa ongelmissa, joihin niin kutsuttu kova systeemiajattelu (”*Hard Systems Thinking”*) ei suoraviivaisuudessaan sovellu. Kova systeemiajattelu ja kova systeemimetodologia (”*Hard Systems Methodology”*) tunnetaan myös systeemisuunnittelun insinöörityönä (”*Systems Engineering”*), jossa hyvin määritettyyn ongelmaan on muodostettavissa saavutettavissa oleva päämäärä tai tavoitetila, johon pääsemiseksi suunnitellaan systeemi. Pehmeä systeemimetodologia taas soveltuu ongelmiin, joissa tavoitetila tai päämäärä ei ole yhtä hyvin määritelty. [13, s. A5-A6, A9-A10, 314, 318] Kova systeemimetodologia kuvataan systeemitieteilijän ja Cardiffin kunniaprofessorin Brian Wilsonin toimesta käsittävän seuraavat viisi vaihetta [14 s. 6-8]:

1. Ongelman määrittely
2. Soveltuvien tekniikoiden kokoaminen
3. Tekniikoiden käyttäminen ratkaisujen johtamiseen
4. Parhaan ratkaisun valinta
5. Ratkaisun jalkauttaminen

Tässä tutkimuksessa voitaisiin hyödyntää ja osaltaan hyödynnetäänkin kovaa systeemimetodologiaa tekoälymenetelmien tarkasteluun kokeellisesti, sillä nämä ongelmat ovat tarkkarajaisia ja omaavat yksiselitteisen päämäärän. Tarkasteltava tutkimuskysymys sisältää kuitenkin ongelmatilanteita, jotka eivät ole yhtä hyvin määriteltyjä, eikä käytettävissä ole riittävästi dataa näiden ongelmien tarkkarajaiseen määrittelyyn ja eksplisiittisten päämäärien muodostamiseen. Tästä syystä suureen osaan tekoälymenetelmien arvioinnista käytetään pehmeää systeemimetodologiaa (”*Soft Systems Methodology”,* jatkossa SSM). SSM määritellään usein koostuvaksi seitsemästä vaiheesta. [14, s. 7] Näitä vaiheita ovat [13 s. 163; 14 s. 7]:

1. Ongelmallisen tilanteen määrittely
2. Tilanteen ilmaiseminen
3. Relevanttien konseptien valitseminen / Oleellisten systeemien määrittäminen
4. Konseptien järjestäminen älykkääksi rakenteeksi / Konseptuaalisten mallien kokoaminen
5. Tämän rakenteen hyödyntäminen tilanteen tutkimiseksi / Konseptien vertaaminen ilmaistuun tilanteeseen
6. Tilanteen muutosten määrittely (esimerkiksi ongelmat, jotka tulee ratkaista)
7. Muutosten jalkauttaminen / Toimet ongelmatilanteen parantamiseksi

Tässä tutkimuksessa käytetään menetelmänä tästä käsityksestä mukailtua, spesifisti tekoälymenetelmien arviointiin tarkoitettua pehmeän systeemimetodologian prosessia, joka on esitetty alla.



KUVA 1: Mukailtu SSM-metodologia tutkimusongelman ratkaisemiseksi [13; 14]

Tutkimuksessa jaetaan tekoälyn soveltamisalat neljään kategoriaan mukaillen John Boydin OODA-silmukkaa ja siitä johdettua Leslie M. Blahan näkemystä koneellisesta päätössilmukasta [15]. Näitä kategorioita tarkastellaan yllä olevan SSM prosessin mukaisesti muodostaen ensin käsitys ongelmatilanteesta ja ilmaisemalla se ymmärrettävästi. Tämän jälkeen määritetään oleelliset systeemit eli kokonaisuuden osat, joissa tekoälyn menetelmiä voidaan käytännöllisimmin hyödyntää. Tässä tapahtuu samanaikaisesti rajauksen tarkennus, sillä esimerkiksi kolmansien osapuolten tuottamaan informaatioon vaikuttaminen ei ole tämän tutkimuksen tarkastelun piirissä. Oleellisiksi systeemeiksi pyritään kuvaamaan ne, joihin voidaan vaikuttaa organisaation omin toimenpitein. Neljännessä vaiheessa kootaan lähteiden ja kokeiden perusteella soveltuviksi arvioitujen tekoälymenetelmien kirjo, jota vaiheessa 5 verrataan alkuperäiseen tilanteeseen ja iteroidaan tarvittaessa. Kun konsepteista on löydetty parhaat kandidaatit, tarkastellaan vaadittavat muutokset näiden menetelmien täysimääräiseksi hyödyntämiseksi. Tutkimus pyrkii suorittamaan prosessin ensimmäiset kuusi kohtaa jokaiseen neljästä kategoriasta, muodostaen johtopäätöksiksi esitykset soveltuvista tekoälymenetelmistä kuhunkin ja näiden edellyttämistä muutoksista käyttöönoton mahdollistamiseksi. Huomionarvoista on, että vaikkei tutkimuksen tarkastelu keskity autonomisiin järjestelmiin, on täysin

autonominen järjestelmä systeemi, joka ratkaisee tässä tutkimuksessa esitettyjen kategorioiden ongelmatilanteet itsenäisesti. Näin ollen kategorioiden systeemit tukevat koneellisen autonomian saavuttamista.

Tutkimuksen rakenne muodostuu johdannosta, kahdesta perusteluvusta ja kokoavasta johtopäätösluvusta. Johdannon jälkeen toisessa luvussa käsitellään tekoälyn ja koneoppimisen teoriaa sekä lyhyesti automaatiota ja robotiikkaa muodostamaan käsitys tekoälyn käsitteestä tämän opinnäytetyön viitekehyksessä sekä teoriapohja kokeellisten liitteiden toteuttamiselle ja tekoälyn menetelmien hyödyntämismahdollisuuksien kartoitukselle. Koska tekoälyn saralla laivastojoukkojen osalta vahvin trendi on nimenomaan autonomisten lavettien kehitys, tarkastellaan myös miehittämättömien järjestelmien tyyppejä ja kontrolliasteita sekä autonomisten järjestelmien toimintaperiaatteita. Kolmannessa luvussa muodostetaan lyhyesti käsitys laivastojoukkojen tehtävistä ja toimintaympäristöstä, jonka pohjalta tarkastellaan tekoälyn menetelmien hyödyntämistä kategorioittain, sovitettuna OODA-silmukan rakenteeseen. Lisäksi esitellään liitteissä 2 ja 5 tutkimusraportin turvaluokitukseen sovitetut tulokset kokeellisten mallien osalta, jolla havainnollistetaan määrällisesti datan ja koneoppimisen hyödyntämistä operatiivisen tilannekuvan luomisen tukemisessa kolmessa eri tapauksessa. Neljännessä luvussa eli johtopäätöksissä muodostetaan kolmannen luvun analyysien perusteella vastaukset tutkimuksen pääkysymykseen.

1.4 Aikaisempi tutkimus ja tutkimuksen keskeinen lähdeaineisto

Vaikka järjestelmäautonomia ei ole synonyymi tekoälylle, suuri osa tekoälytutkimusta keskittyy nimenomaisesti autonomisten järjestelmien mahdollistamiseen ja kehittämiseen. Tutkimustilanne tekoälyn ja merellisten autonomisten järjestelmien osalta on Suomessa suurelta osin yliopistojen ja kaupallisten toimijoiden tutkimusta ja kehitystyötä. Kotimaisista tutkimuksista ja niiden tuloksista osa menee liikesalaisuuksien alle, mutta tutkimus- ja kehitystyötä on tehty muun muassa AAWA (*Advanced Autonomous Waterborne Applications*) aloitteen suunnalta, jossa suomalaiset yliopistot ovat tehneet yhteistyötä alan yritysten kanssa tekoälyn ja autonomisen merenkulun saralla [16]. Samoin ruotsalainen puolustusteollisuuskonserni Saab tekee tutkimusyhteistyötä Aalto yliopiston kanssa nimeten tutkimuskohteiksi muun muassa tekoälyn [17]. Ulkomailla tehdään laajaa tutkimusta ja kehitystyötä myös asevoimien toimesta, pääasiassa Yhdysvalloissa ja Kiinassa [18; 19]. RAND tutkimuslaitoksen raportissa Yhdysvaltojen osuudeksi autonomisista merellisistä patenteista lasketaan 38.8% ja Kiinan osuudeksi 35.92% yhteismäärän ollessa noin 3200. [20 s. 83] Raportin patenttimäärien perusteella suurimmat toimijat ovat maittain suuruusjärjestyksessä Yhdysvallat, Kiina, Etelä-Korea, Saksa, Japani, Venäjä, Ranska ja Iso-Britannia, muiden maiden kattaessa alle yhden prosentin osuus lasketuista patenteista. Yhtiöittäin tai toimijoittain suurin patenttien haltija on Yhdysvaltain laivasto ja toisena Google, korkeimmin sijoittuneen kymmenen toimijan joukossa ollen yhdysvaltalaisten toimijoiden lisäksi neljä kiinalaista toimijaa ja yksi saksalainen. [20 s. 83-84] Yhdysvaltojen laivastolla on lisäksi oma tekoälyn tutkimuskeskus osana laivaston tutkimuslaboratoriota [21]. Saatavilla olevat asevoimien tutkimusraportit eivät lähtökohtaisesti paljasta teknisiä yksityiskohtia tai todellisia suorituskykyjä.

Sam J. Tangredi ja George Galdorisi ovat toimittaneet keväällä 2021 kirjan ”*AI at War: How Big Data, Artificial Intelligence, and Machine Learning are Changing Naval Warfare”*, joka käsittelee tämän tutkimuksen kanssa yhteneviä tutkimuskysymyksiä. Kirja kuvaa esipuheessa itseään uraauurtavaksi askeleeksi rakentaa perusta tekoälyn käyttämiseksi asevoimien päätöksenteossa. Lukuisten kirjoittajien tekstejä koostava kirja kuvaa kattavasti Yhdysvaltojen laivaston hahmottamia merisodankäynnin ongelmia ja niiden ratkaisemista tai parantamista tekoälyn keinoin. [22 s. xi]

Julkaisussa ”*Navigation and Control of Autonomous Marine Vehicles”* todetaan, että merellisten autonomisten järjestelmien kehitystyö on jäänyt jälkeen maalla ja ilmassa sekä vedenpinnan alla toimivien autonomisten järjestelmien kehityksestä, siitä huolimatta, että kaikki edellä mainitut ulottuvuudet ovat kategorisesti yhtä tärkeitä. Pinnalla toimivat järjestelmät (USV, *unmanned surface vehicle*) ovat verrattain alemmalla maturiteettiasteella oletettavasti merellisen ympäristön haastavuudesta sekä julkisen että yksityisen sektorin tarpeista johtuen. Lisäksi todetaan, ettei sotilaallisista USV järjestelmistä ole kattavaa kirjallisuutta saatavilla tiedon turvaluokituksista johtuen. [23 s. 1-2]

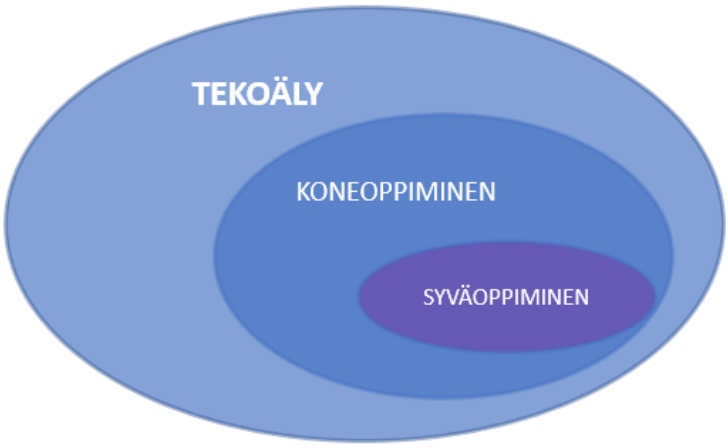
Maanpuolustuskorkeakoulussa on tehty autonomisten järjestelmien ja tekoälyn aihepiiristä useita pro gradu -tutkielmia ja diplomitöitä viime vuosina, kuten maavoimien näkökulmasta toteutettu Lauri Eerolan ”*Autonomisten maataistelurobottien torjunta pataljoonan taistelussa 2030-luvulla*” [24]. Eero Havu on tehnyt diplomityön tekoälyn vaikutuksesta Venäjän sotataitoon ja Tomi Lyytinen koneoppimisen hyödyntämisestä ilmaoperaation suunnittelussa [25; 26]. Lisäksi Lauri Kilpeläinen on tehnyt pro gradun aiheesta ”*Tekoälyavustettu taistelunjohto*”, joka kartoittaa taktiikan näkökulmasta tekoälyn hyödyntämistä ilmavoimien taistelunjohtokeskuksen toiminnassa. [27] Laivaston näkökulmasta autonomisten järjestelmien tutkimus on painottunut miinanetsinnän ympärille. Näistä viimeisimpiä on Mikko Timoskaisen ”*Vedenalaisen autonomisen järjestelmän vaikutus aluksen turvallisuuteen”* [28]. Oikeudellisesta näkökulmasta on tehty muun muassa Björn Röbergin diplomityö johtamisen ja sotilaspedagogiikan laitokselle sodan oikeussäännöistä ja merellisistä miehittämättömistä järjestelmistä [29] ja puolustusvoimien tutkimuslaitos on toteuttanut selvityksen tekoälyn maturiteetista osana toista tutkimus- ja kehitystehtävää [30]. Edellä mainituista lähinnä tämän tutkimuksen tarkastelua ovat Lyytisen ja Kilpeläisen työt. Laivaston pintatorjuntalaivueiden ja alustaisteluosastojen suorituskykyjen ja tehtäväkentän näkökulmasta ei ole tämänhetkisen tietämyksen mukaan tehty tutkimusta, joka kartoittaisi teknologianäkökulmasta tekoälyn käyttöönottomahdollisuuksia Suomen laivastojoukoissa.

Tutkimuksen teorian keskeinen lähdeaineisto koostuu tekoälyn ja koneoppimisen kirjallisuudesta sekä opetusmateriaaleista ja tekoälytutkimuksen raporteista sekä tieteellisistä julkaisuista. Tekoälyn ja koneoppimisen teoria pohjautuu pääasiassa yliopisto-opetuksessa yleisesti käytetyn Stuart Russellin ja Peter Norvigin *”Artificial Intelligence- A Modern Approach”* kirjan kolmanteen painokseen hyödyntäen myös Neapolitan ym. kirjaa *”Artificial Intelligence: with an Introduction to Machine Learning Second Edition”* ja Wolfgang Ertelin *”Introduction to Artificial Intelligence”* kirjaa sekä Aalto yliopiston oppimateriaaleina käytettyjä *Foundations of Machine Learning* sekä *Basic Principles of Machine Learning* oppikirjoja [31; 32; 33; 34; 35]. Syväoppimisen päälähteenä on Massachusetts Institute of Technology julkaisema Ian Goodfellown, Yoshua Bengion ja Aaron Courvillen*”Deep Learning”*. [36] Autonomisten järjestelmien lähdeaineisto koostuu pääasiassa yksityiskohtaisista, tieteellisistä julkaisuista, jotka käsittelevät muun muassa autonomisten merenkulkujärjestelmien toimintaa ja arkkitehtuuria. Liitteiden kokeissa hyödynnetty data on julkisten lähteiden aineistoja, Merivoimien arkistodataa merellisistä harjoitusolosuhteista sekä Meritaistelukeskuksen ja Merioperaatiokeskuksen taltioimia operatiivisessa valmiusalustoiminnassa tuotettuja tunnistuskuvia.

2 Tekoäly ja autonomiset järjestelmät

Tässä luvussa käydään läpi tekoälyä ja koneoppimista sekä automaatiota ja robotiikkaa käsitteistön ja tutkimuksen rajausten selkeyttämiseksi ja tarvittavan taustatiedon koostamiseksi laadullista analyysia varten.

2.1 Tekoäly ja algoritmit



KUVA 2: Tekoäly on poikkitieteellinen tieteenala, joka sisältää muun muassa koneoppimisen osa-alueen

2.1.1 Tekoälyn määritelmä

Tekoälyn, englanniksi AI (*artificial intelligence*), määritelmä on moniulotteinen ja tulkintoja on useita. Poole, Mackworth ja Goebel käyvät artikkelissaan ”*Computational Intelligence: A Logical Approach*” läpi tekoälyn määrittelyä ja käsitteistöä. Lopputulemana on, ettei tekoäly ole terminä yksiselitteisesti oikea, vaan tutkijat päätyvät käyttämään ”*computational intelligence*” termiä, eli laskennallista älyä. Myös ”synteettinen äly” nähdään artikkelissa hyvänä terminä kuvaamaan laajasti käytössä olevaa AI käsitettä oikeammin, sillä siitä huolimatta, että tekoäly on keinotekoinen (”*artificial*”) eikä luonnollinen, se osoittaa silti määritelmällisen älyn ominaisuuksia. Yksinkertaistetusti äly on kyky hankkia ja hyödyntää tietoa ja taitoa, eli oppia. [37] Laskennallinen äly on terminä hyvä, sillä se ei erottele älyä ilmentävän agentin luonnollisuutta tai synteettisyyttä vaan pelkästään käsiteltävän älyn laatua. Suomen asevoimissa käytetään usein termiä keinoäly, jonka terminologinen ero tekoälyyn on varsin marginaalinen. Erillisenä terminä ”keino” kuvaa tapaa tai menetelmää tehdä jotain, kun taas ”teko” on tekemisen keskeneräinen tai valmis tuotos. Tässä mielessä keinoäly voidaan nähdä semanttisesti kuvaavampana terminä. Molemmat termit kuitenkin tuottavat yhdyssanan alkuosana merkityksen keinotekoiselle versiolle jälkiosan merkityksestä. [38; 39] Tässä työssä käytetään termiä tekoäly sen vakiintuneen kielellisen aseman vuoksi.

Tekoäly-termin ja siten myös tekoälyn isänä tunnettu matemaatikko John McCarthy vastaa artikkelissa ”*What is Artificial Intelligence?*” kysymykseen ”mitä on tekoäly?” siten, että tekoäly on tieteenala ja tekniikka älykkäiden

koneiden kehittämiseksi, erityisesti tietokoneohjelmien muodossa. Määritelmässä tekoälyn todetaan liittyvän tietokoneellisesti mallinnettuun inhimillisen älykkyyden ymmärrykseen, mutta tekoälyn ei tarvitse rajautua metodeihin, jotka ovat biologisesti havaittavissa. [40]

Professori Wolfgang Ertel kokee kirjassaan *Introduction to Artificial Intelligence* tohtori Elaine Richin määritelmän tekoälystä parhaiten aikaa kestäväksi, sillä se määrittelee tekoälyn tutkimustyöksi, joka pyrkii saamaan tietokoneet suorittamaan tehtäviä, joissa ihmiset ovat toistaiseksi parempia. [32 s. 2; 41 s. 1] Määritelmä on ajattomuutensa ansiosta erittäin hyvä, kuvaten samalla tekoälytutkijoiden työtä viimeisten vuosikymmenien ajalta. Tässä opinnäytetyössä käytetään tätä tekoälyn määritelmää, sillä se mukailee John McCarthyn alkuperäistä ajatusta ja sopii tämän työn hypoteesiin ihmisten suorittamien tehtävien tekemisestä koneellisesti. Tässä työssä puhuttaessa tekoälysovelluksista tarkoitetaan tekoälytutkimuksen mahdollistamia ja tutkimusta hyödyntäviä mallinnuksia ja niiden käyttöä lopputuotteena tai sen osana.

Tekoälyn käsitteeseen liittyy keskeisesti agenttiajattelu eli ajatus rationaalisista, älykkäistä agenteista, jotka havainnoivat ympäristöään sensorein ja toimivat tuossa ympäristössä jonkin aktuaattorin avulla. Keinotekoisen agentin toimenpiteitä määrittää funktio, joka kuvaa agentin havainnot toiminnaksi. Kyseistä, abstraktia funktiota toteutetaan agentin konkreettisessa ohjelmassa, jolloin muodostuu älykäs agentti, mikäli funktio ja ohjelma implementoivat älykkääksi luokiteltua käytöstä. [31 s. 35] Käsiteltäessä tekoälyä nousee luonnollisesti esiin kysymys älystä ja älykkyydestä. Massachusetts Institute of Technologyn professori Max Tegmark määrittelee tekoälyä käsittelevässä kirjassaan nykyaikaisesti mutta yksinkertaisesti älykkyyden kyvyksi saavuttaa monimutkaisia tavoitteita. [42 s. 39] Tekoälyn pioneeri Alan Turing kehitti ihmisenkaltaisen älykkyyden testaamiseen nimeään kantavan testin. Kyseisessä testissä kuulustelijana toimiva ihminen kommunikoi tekstipäätteen kautta sekä ihmisen että ihmisenä esiintyvän älykkään järjestelmän kanssa, ja hänen tulee pystyä päättelemään, kumpi vastapuolista ei ole ihminen. Mikäli kuulustelija ei kykene erottamaan ihmistä koneesta, on kyseinen kone testin perusteella älykäs. Näin ollen Turingin testin läpäisevä keinotekoinen agentti olisi nimenomaisesti älykäs agentti. John Searle esitti vasta-argumenttina Turingin testille ajatusmallin kiinalaisesta huoneesta. Kiinalaisessa huoneessa on ihminen, joka ei ymmärrä kiinaa, sekä tarkat ohjeet toteuttaa tietty ohjelma. Ihminen saa ulkoa syötteet kiinaksi ja prosessoi ne ohjeiden mukaisesti tulkiten kiinalaisia kirjoitussymboleja ymmärtämättä niiden merkitystä eli toteuttaen ohjelmaa ymmärtämättä sitä. Tämän argumentin perusteella älykkäältä vaikuttaminen ei ole todistus älystä. Searle johti käsitteen vahvasta tekoälystä, jolla on kyky ymmärtää suorittamansa tehtävät ja omata kognitiivisia tasoja. Kiinalaiseen huoneen perusteella hän päätteli tällaisen tekoälyn olevan mahdoton. Heikolla tekoälyllä vastaavasti tarkoitetaan tekoälyä, joka kykenee toimimaan ja vaikuttamaan älykkäältä välttämättä ymmärtämättä käsittelemäänsä. [33 s. 3] Nykyhetkellä on olemassa vain tämänkaltaisia heikkoja tekoälyjä. Tässä työssä ei käsitellä älyn ja tekoälyn käsitteitä filosofiselta kannalta, eikä myöskään näiden aiheuttamia tutkimuskysymyksiä tekoälyn etiikasta tai moraalista.

Nykyisellään kaikki olemassa olevat tekoälysovellukset ovat siis heikkoja eli niin sanottuja kapeita tekoälyjä (NAI, *narrow artificial intelligence*), sillä ne toimivat optimaalisesti jossain tietyssä, usein hyvin rajoitetussa ympäristössä. [3 s. 2; 42 s. 39] Jonkin osa-alueen, kuten merenkulkujärjestelmän, automatisointi ja toiminnallinen autonomia edellyttää yhden tai useiden kapeiden tekoälysovellusten syötteiden yhdistämistä jonkinlaisen päätöksente-

koalgoritmin käyttöön autonomisen kokonaisuuden muodostamiseksi. Tällöin järjestelmä kykenee toimimaan jossain määrin adaptiivisesti vaihtelevassa ympäristössä, ilmentäen toiminnassaan ihmisenkaltaisia kognitiivisia kykyjä.

Terminologisesti tekoäly jaetaan aiemmin mainittuun kapeaan tekoälyyn ja yleiseen tekoälyyn, englanniksi AGI (”*Artificial General Intelligence*”). AGI vastaa käsitteenä hyvin pitkälti Searlen määritelmää vahvasta tekoälystä ja on siis ihmisen tasoinen tekoäly. NAI ja AGI lisäksi puhutaan universaalista älystä, jolla tarkoitetaan kykyä muodostaa yleinen äly datan ja resurssien pohjalta, sekä superälystä (”*super intelligence*”), jolla tarkoitetaan hypoteettista ihmisen älyn huomattavasti ylittävää älykkyyttä. Tekoälyn tapauksessa superäly mielletään usein yleisen tekoälyn kehittämäksi tekoälyksi, jonka evoluutio superälyksi olisi eksponentiaalisesti kiihtyvä iteraatio. [4 s. 2; 42 s. 39] Superälyä ja siihen liittyvää kiihtyvän iteraation singulariteettia ei käsitellä tässä tutkimuksessa tämän enempää.

Tekoälyn käsitteestä on huomioitava myös se, että määritelmän on ollut tapana siirtyä tarkoittamaan aina toistaiseksi saavuttamattomia sovelluksia. [43 s. 242] Saavutetut kehitysasteet ovat arkipäiväistyneet ja muuttuneet määritelmällisesti esimerkiksi automaatioksi tai ohjelmistorobotiikaksi.

Tekoälyn hyödyntämisessä on kaksi lähestymistapaa: joko tehostaa nykymuotoista toimintatapaa tai tuottaa täysin uusia toimintatapoja. Tässä opinnäytetyössä tarkastellaan laivastojoukkojen toimintaa molemmilla lähestymistavoilla.

2.1.2 Tekoälyn historia

Tekoälyn taustalla on Russelin ja Norvigin taustoituksessa kolme matemaattista, perustavanlaatuista osa-aluetta: logiikka, laskenta ja todennäköisyys. Logiikka juontaa jo antiikin Kreikasta, mutta laskennan mahdollistavan propositio- tai Boolean-logiikan esitteli George Boole vuonna 1847. Vuonna 1879 Gottlob Frege laajensi propositiologiikan ensimmäisen kertaluvun predikaattilogiikaksi, jota käytetään edelleen. [31 s. 7-8]

1930-luvulla Kurt Gödelin, Alonso Churchin ja Alan Turingin tutkimus ja innovaatiot logiikassa ja teoreettisessa tietojenkäsittelytieteessä loivat perustan tekoälylle, joka on ollut itsenäisenä tieteenalana olemassa 1950-luvulta alkaen. Yhtenä keskeisimmistä pohjustuksista toimi Gödelin täydellisyysteoreema, joka todistaa, että ensimmäisen kertaluvun predikaattilogiikka on täydellistä, jolloin jokainen predikaattilogiikalla formuloitu tosi lause voidaan todistaa formaalin laskennan säännöin. Tämän teoreeman pohjalta voitiin myöhemmin muodostaa automaatiota formaalin laskennan implementaatioina. Gödel myös osoitti epätäydellisyysteoreemalla korkeampien kertalukujen predikaattilogiikan muodostavan todistamattomissa olevia tosia lauseita, osoittaen näin formaalien järjestelmien rajoitteita. Epätäydellisyysteoreeman osoittaessa, että jotkin kokonaislukujen funktiot eivät ole laskettavissa, tarkasteli Turing laskettavissa olevien funktioiden piirteitä ja muodosti samalla muun muassa pysähtymisongelman, joka osoitti älykkään järjestelmän rajoitteita. Pysähtymisongelman mukaan kone ei voi kertoa yleisesti, antaako kyseinen ohjelma tuloksen syötteelle vai jatkuuko ohjelman ajo ikuisesti. [32 s. 5-7; 31, s. 7-8]

Tekoälyn muodostuminen käytännölliseksi tieteenalaksi mahdollistui vasta ohjelmoitavien tietokoneiden kehittämisen myötä. Newell ja Simon kehittivät ensimmäisen automaattisen todistajan, Logic Theoristin, jota kutsutaan

myös ensimmäiseksi tekoälyohjelmaksi. Logic Theorist paitsi ratkaisi matemaattisia ongelmia, myös osoitti tietokoneiden kykenevän käsittelemään lukujen ohella symboleja. John McCarthy kehitti LISP-ohjelmointikielen symbolisten rakenteiden prosessointiin. Molemmat näistä esiteltiin McCarthyn järjestämässä Darthmouthin konferenssissa vuonna 1956, jota pidetään tekoälyn syntyhetkenä. Näistä seurasi symbolisen tekoälyn kehittäminen, joka oli johtava tekoälyn paradigma 1980-luvulle asti. [32 s. 8-9] Symbolisella tarkoitetaan tässä yhteydessä korkean tason kieltä, joka on ihmisen luettavissa ja joka ohjelmointikielen tapauksessa käännetään konekieliseksi aritmeettista suorittamista varten. Klassisen teorian mukaan ihmisen kognitio on siis pohjimmiltaan laskentaa suorittava järjestelmä, joka manipuloi symbolisia ilmentymiä säännöstönsä mukaisesti. Newell ja Simon kehittivät Logic Theoristin pohjalta vuonna 1975 fyysisen symbolien systeemin hypoteesin (PSSH, *physical symbol system hypothesis*), jonka mukaan fyysisten symbolien systeemillä on kaikki tarvittavat ja riittävät keinot muodostaa yleisen älyn (”*general intelligence*”) toimintoja. Tässä hypoteesissa fyysisten symbolien järjestelmä on fysiikan lakeja noudattava kokonaisuus, joka muodostuu fyysisistä malleista eli symboleista, jotka ovat lausekkeen eli symbolirakenteen komponentteja. Fyysisten symbolien mielivaltainen joukko voi olla esimerkiksi raapustuksia paperilla tai tapahtumia digitaalisessa tietokoneessa. Näitä symboleja manipuloidaan yksiselitteisillä säännöillä, jotka koostuvat myös fyysisistä symboleista tai merkeistä. Näin ollen kyseinen systeemi koostuu symbolien ilmentymistä ja näiden keskinäisistä fyysisistä suhteellisuuksista kuten lähekkäisyydestä. Fyysisten symbolien järjestelmä on siis kone, joka tuottaa ajan kuluessa kehittyvän kokoelman symbolirakenteita, jossa symbolien kokonaisuus on kuvaavuudeltaan suurempi kuin yksittäisten symbolien summa. [44 s. 116; 45] Ajatus on ymmärrettävämpi, jos se mielletään orgaaniseksi rakenteeksi: esimerkiksi ihminen voidaan tulkita vastaavanlaiseksi järjestelmäksi, joka tuottaa ajan kuluessa muistiinsa sijoitettuna eräänlaisen symbolijärjestelmän, joiden keskinäiset yhteydet mahdollistavat inhimillisen kognition. Pohjimmiltaan Newell ja Simon osoittivat siis tietokoneiden kykenevän tämän hypoteesin perusteella ilmentämään yleistä älyä [44]. Erotuksena symbolisesta lähestymistavasta on neuroverkoissa ja koneoppimisessa tavanomainen numeerinen laskenta. Numeerisella laskennalla viitataan tyypillisesti algoritmeihin, jotka ratkaisevat matemaattisia ongelmia iteratiivisella prosessilla sen sijaan, että löytäisivät analyyttisesti symbolisen lauseen oikealle vastaukselle. [36 s. 77]

Symbolisella, logiikkaan perustuvalla implementaatiolla saavutettiin aikaisia läpimurtoja. Kyseinen lähestymistapa tavoitteli AGI-tyyppisen älykkään ohjelman luomista. Kehitetyt ohjelmat kuitenkin toimivat vain rajoitetuissa ympäristöissä ratkaisten suhteellisen yksinkertaisia ongelmia epäonnistuen skaalautumaan vaikeampien ongelmien ratkaisuun. Skaalautumisen epäonnistumisen vuoksi näitä metodeja kutsutaan heikoiksi metodeiksi. Koska yleisesti soveltuvan, moniin ongelmiin skaalautuvan tekoälysovelluksen luominen ei onnistunut, tutkijat alkoivat kehittää erityisiin, rajattuihin ongelma-alueisiin keskittyneitä tietopohjaisia järjestelmiä, jotka tekivät usein asiantuntijoille tyypillisiä tehtäviä. Tällaisissa symbolisen tekoälyn asiantuntijajärjestelmissä tiedosta muodostetaan säännöstö tiettyyn, rajattuun alaan, jossa päättelyn toteuttaa algoritmi käyttämällä kyseistä säännöstöä. Asiantuntijajärjestelmän säännöstö muodostetaan usein varsinaisen asiantuntijan tiedon ja ymmärryksen pohjalta, mallintaen ihmisasiantuntijan päättelyketjuja jäljittelevä rakenne. [33 s. 4-5; 32 s. 8-9] Säännöstö rakentuu propositio- ja predikaattilogiikan formaalissa kielessä matemaattisilla loogisilla konnektiiveilla muodostuen implikaatioista, konjunktioista ja disjunktioista, jotka ohjelmointikielissä ovat usein muodossa IF, AND, NOT, OR ja THEN, sekä matemaattisina symboleina ”→”, ”∧”, ”¬” ja ”∨” sekä ”jos ja vain jos” konnektiivi ”↔”. Näin ollen yksinkertainen asiantuntijasäännöstö nisäkkään tunnistamiseksi muista eliöistä voidaan kirjoittaa formaalisti muotoon:

$$((tasalämpöinen) \wedge (selkärankainen) \wedge (synnyttää\ eläviä\ poikasia)) \vee (nokkaeläin) \leftrightarrow (nisäkäs)$$

tai vaihtoehtoisesti ohjelmointikielisemmässä muodossa

```
IF        tasalämpöinen
AND       selkärankainen
AND       synnyttää eläviä poikasia
OR        nokkaeläin

THEN      laji = nisäkäs
```

Kun säännöstöä tulkitaan päättelyalgoritmin toimesta, saadaan muodostettua asiantuntijapäätelmiä automaattisesti käytettävissä olevan tiedon ja havaintojen pohjalta. [33 s. 5-6, 30-33] Monimutkaisessa asiantuntijajärjestelmässä on erittäin haastavaa huomioida kaikkia päättelyyn vaikuttavia sääntöjä ja näiden poikkeuksia. [33 s. 31] Mikäli säännöstö on epätäydellinen, ovat sen pohjalta tehtävät päätelmät usein virheellisiä. Mikäli yllä olevasta esimerkistä olisi jätetty poikkeus asettamatta ja säännöstön pohjalta arvioitaisiin nokkasiiliä, päädyttäisiin lopputulokseen, ettei se ole nisäkäs. Asiantuntijajärjestelmien heikkoudeksi muodostui epävarmuustekijöiden huomioiminen ja epävarmuuden mallintaminen, sillä mikäli tietopohja oli suuri ja siihen liittyi vaikeasti määriteltäviä epävarmuustekijöitä, suoraviivainen säännöstölogiikka tosien ja epätosien väittämien välillä ei vastannut ihmisen päättelykyvyn mallia erityisen hyvin. Näin ollen päädyttiin muun muassa todennäköisyyspohjaiseen eli probabilistiseen lähestymiseen tekoälyssä. [33 s. 6; 32 s. 71-72] Probabilistinen säännöstö antaisi binäärisen totuusarvon sijaan sumean logiikan mukaisen todennäköisyyden toteumalle reaalilukuna nollan ja yhden väliltä, mallintaen paremmin päättelyyn liittyvää epävarmuutta ja kuvaten liukuvasti päättelyn lopputulosta.

Ennen todennäköisyyspohjaisen ja bayesilaisen päättelyn tekoälyä 1980-luvun lopulla oli uuden konnektionismin ajanjakso, joka osoitti matemaattisesti mallinnettujen, numeeristen neuroverkkojen kykenevän oppimaan esimerkkien pohjalta ratkaisuja ongelmiin, jotka symbolisen tekoälyn tapauksessa olisivat vaatineet huomattavan määrän ohjelmointia. Konnektionismin puitteissa ei kuitenkaan kyetty taltioimaan neuroverkon oppimaa kaavamaisuutta tai säännöstöä yksinkertaisiksi kaavoiksi tai loogisiksi säännöiksi. Neuroverkkojen yhdistäminen loogisiin säännöstöihin ja ihmisasiantuntijoiden tietotaitoon osoittautui äärimmäisen vaikeaksi. [32 s. 9]

Numeerisen konnektionismin lisäksi symbolista tekoälyä ja PSSH:ta kohtaan on esitetty useita vastaväitteitä, joista Nils Nilsson on tarkastellut neljää pääteemaa artikkelissa ”*The Physical Symbol System Hypothesis: Status and Prospects*” [46]. Yksi näistä teemoista on kognitiotieteilijä Steven Harnadin saman nimisessä artikkelissa käsittelemä ”*The Symbol Grounding Problem*”, vapaasti suomennettuna ongelma symbolien perustamisesta johonkin. Perustamista kuvaavampi termi voisi olla sidonta tai liittäminen, sillä tarkoitus on kuvata symbolin liittämistä sen merkitykseen ja tarkoitteeseen sekä sitä problematiikkaa, joka tästä liittämisestä tai liittämättömyydestä syntyy. Harnad ottaa uudelleen esiin Searlen kiinalaisen huoneen malliesimerkkinä älystä, joka ei ymmärrä käsittelemiään symboleita. Hänen mukaansa kognitiivinen ajattelu ei voi koostua nykyisten tietokoneiden toiminnan pohjana olevasta symbolien manipulaatiosta paitsi ratkaisuksi esitetyssä mallissa, jossa symbolit liitetään kahteen ei-symboliseen representaatioon. Ensimmäinen näistä on ikoninen representaatio, joka on Harnadin kuvauksessa analoginen distaalisen objektin proksimaalinen sensoriprojektio, karkeasti yksinkertaistettuna sensorihavainto. Toinen on kategorinen representaatio, opittujen ja synnynnäisten piirteiden tunniste, jonka tarkoitus on löytää muuttumattomat objekti- ja tapahtumaluokkien piirteet niiden sensoriprojektioista, eli yksinkertaistetusti luokittaa sensorihavaintojen piirreavaruudet. Alkeissymbolit ovat tällaisten objekti- ja tapahtumaluokkien nimiä, jotka on osoitettu niille

kategorisen representaation perusteella. Korkeamman tason symboliset representaatiot on liitetty näihin alkeissymboleihin ja ne käsittävät keskinäisten kategoristen suhteiden kuvailun. Harnadin mukaan konnektionismi on luonnollinen kandidaatti mekanismille, joka oppii muuttumattomat piirteet kategorisista representaatioista yhdistäen nimet eli alkeissymbolit niihin proksimaalisiin distaalisten objektien projektioihin, joita ne tarkoittavat. Näin ollen konnektionismi voidaan nähdä täydentävänä komponenttina mallintaessa ihmismielen kognitiota symbolisen ja ei-symbolisen eli numeerisen representaation hybridinä. [45]

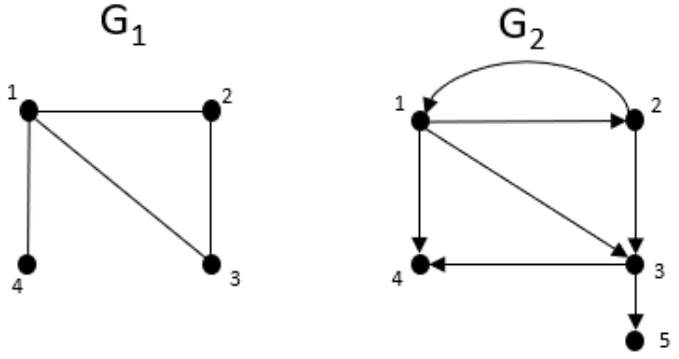
Konnektionismin ja symbolisen tekoälyn kädenväännössä lupaavimmaksi tieksi eteenpäin tekoälyn kehityksessä muodostui edellä mainittu probabilistinen päättely, joka toimii ehdollisilla todennäköisyyksillä propositiologiikan kaavoissa. Ehdollinen todennäköisyys perustuu Bayesin teoreemaan, joka on esitetty kaavassa 1. Kaava laskee ehdollista todennäköisyyttä tapahtumalle A ehdolla B. Teoreema toimii perustana suureen osaan moderneja lähestymistapoja mallintaa epävarmuutta tekoälyjärjestelmissä [31 s. 9]

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

KAAVA 1: Bayesin teoreema. [31 s. 9]

1970-luvulta alkaneen probabilistisen päättelyn kehityksen myötä, tultaessa 1990-luvulle, monet diagnostiikka- ja asiantuntijajärjestelmät päätyivät hyödyntämään Bayesin teoreemaa ja Bayes-verkkoja. Bayes-verkot esittävät tietoa epävarmasta alasta verkottaen joukon satunnaismuuttujia ja niiden välisiä ehdollisia riippuvuuksia solmuiksi ja niiden välisiksi suunnatuiksi kaariksi. Bayes-verkot liittyvät tekoälyssä ja koneoppimisessa käytettyyn suunnattujen syklittömien verkkojen malliin. [33 s. 6, 142; 47 s. 1] Verkkoteoria ja erityisesti suunnattujen syklittömien verkkojen kuten Bayes-verkkojen käyttö on korostunut tilastotieteen ja tekoälyn tutkimuksessa, jossa useat tunnetut koneoppimisen metodit kuten neuroverkot mielletään joissain tarkasteluissa Bayes-verkkojen erityistapauksiksi. [47 s. 4; 48 s. 4]

Bayes-verkkojen lisäksi myös päätöspuut ja neuroverkot ovat käsitettävissä ja kuvattavissa verkkoteorian avulla. Matemaattisessa verkkoteoriassa verkot muodostuvat solmuista, jotka kuvataan usein pisteinä, sekä niiden välisistä kaarista, joita kuvataan yleensä nuolilla tai viivoilla. [33 s. 149; 49 s. 109-110] Alla on kuvattu kaksi yksinkertaista verkkoa.



KUVA 3: Suuntaamaton verkko G_1 ja suunnattu, syklinen verkko G_2

Kuvassa 3 on verkko G_1, jonka kaaret ovat suuntaamattomia eli niitä voi kulkea molempiin suuntiin. Näin ollen kaikista solmuista {1,2,3,4} on olemassa vähintään yksi reitti jokaiseen muuhun solmuun. Verkossa G_2 kaaret ovat suunnattuja, jolloin esimerkiksi solmusta 5 ei ole reittiä mihinkään muuhun solmuun mutta solmuista 1, 2 ja 3 on reitti solmuun 5, ja solmujen 1 ja 2 väliin muodostuu sykli, sillä solmujen välillä voi kulkea molempiin suuntiin. [33 s. 149-150; 49 s. 109-110]

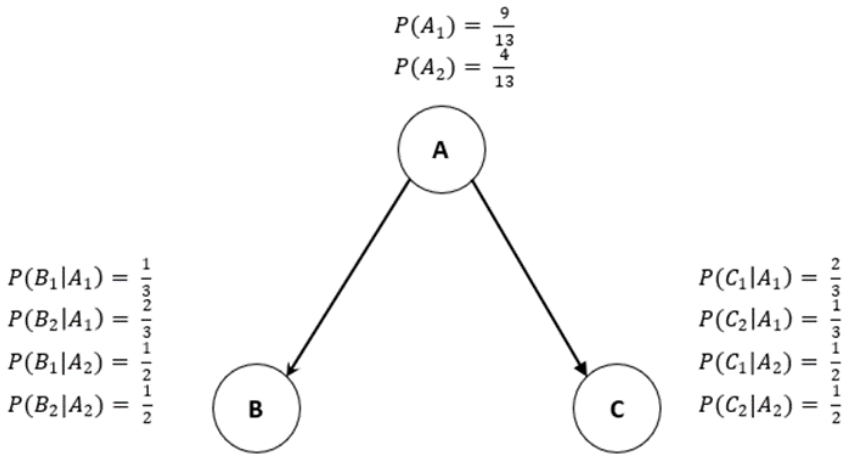
Verkkoteorialla voidaan kuvata useita erilaisia ongelmia eri ympäristöistä ja sillä on useita liittymäkohtia koneoppimiseen ja etenkin syväoppimisen teoriaan ja käytäntöön, joita käsitellään seuraavassa osiossa. Verkoilla voidaan kuvata esimerkiksi lineaarista prosessia eri vaihtoehtoineen ja sykleineen. Painotetussa verkossa kaarilla on painokertoimet eli painot, jotka ovat joko kiinteät tai muokattavissa. [49 s. 109-110, 153-154] Painot taas voivat kuvata mitä tahansa satunnaismuuttujien merkityksiä fyysisistä etäisyyksistä kumulatiivisiin todennäköisyyksiin. Verkkoteoriaan ja puurakenteisiin liittyvät myös useat hakualgoritmeihin pohjautuvat, myös tekoälyksi mielletyt algoritmit, jotka optimoivat älykkäälle agentille annetun suorituskykymittarin. Tällainen mittari voi olla esimerkiksi verkon polkujen painokertoimien summa lähtötilanteesta maalitilanteeseen. Näin voidaan esimerkiksi mallintaa lyhimpiä tai pisimpiä reittejä solmujen välillä kaarien painojen summina. [31 s. 64-65, 75-86] Hakualgoritmit ja polkujen optimointi ovat autonomisten alusten osalta erittäin tärkeitä esimerkiksi merenkulkujärjestelmien kehityksessä. Optimaalisten reittien ohella verkosta voi muodostua datasta oppimalla päätöspuu, jonka eri haarat muodostavat analyysityökalun ja päätöksentekokykyisen tekoälysovelluksen [33 s. 102].

Bayes-verkosta laskettava ehdollisten todennäköisyyksien tulo tuottaa sitä luotettavamman approksimaation, mitä useampi priori ja enemmän informaatiota on hyödynnettävissä. [31 s. 511-513] Approksimaatio tietyn tapahtuman posterioritodennäköisyydelle muodostuu kaavan 2 tulona.

$$\prod_{i=1}^{n} P(x_i | x_{i-1} \dots, x_1)$$

KAAVA 2: Tulon ketjusääntö Bayes-verkon posteriorin laskemiseksi [31 s. 514]

Kuvassa 4 on havainnollistettu Bayes-verkon periaatetta. Tapahtumalla A on kaksi eri tilaa alaindeksiensä mukaisesti, joilla on tietyt todennäköisyydet tapahtua perustuen otannasta saatuun frekvenssiin tai asiantuntija-arvioon. Näiden tapahtumien perusteella muodostuvat tapahtumien B ja C ehdolliset todennäköisyydet.



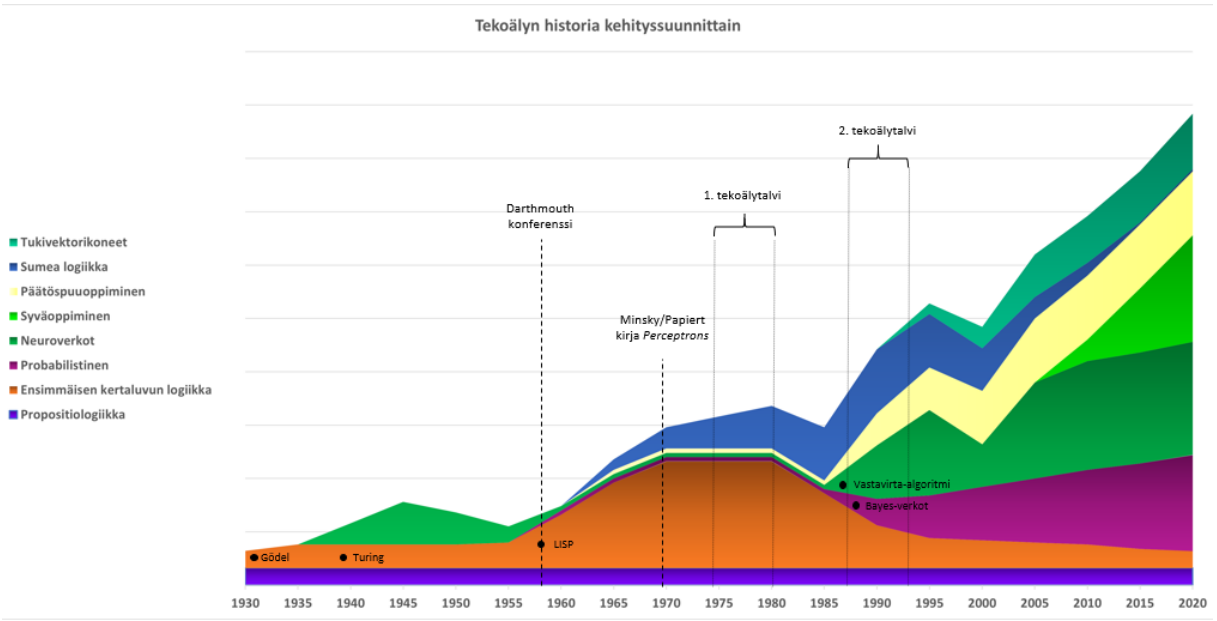
KUVA 4: Esimerkki kolmen solmun Bayes-verkosta [33 s. 153]

Bayes-verkkojen ohella myös neuroverkot palasivat tekoälyn kehityksen keskiöön 1980-luvun lopulla vastavirta-algoritmin uudelleenkeksimisen myötä, sillä alun perin kyseinen algoritmi keksittiin jo 1969. Ensimmäiset neuroverkkomallit tehtiin jo vuonna 1943 McCullochin ja Pittsin liittäessä ajatus myös propositiologiikkaan, mutta alun jälkeen kehityssuunta jäi taka-alalle. Tähän vaikutti etenkin Minskyn ja Papertin julkaisu, jossa osoitettiin Perseptronin, yksineuronisen luokittelijan, kykenevän esittämään vain lineaarisia funktioita. Tämä johti neuroverkkotutkimuksen rahoituksen romahtamiseen. [32 s. 6-8; 31, s. 22]

1990-luvun alussa myös datan louhinta kehittyi tekoälyn osa-alueeksi osana tilastollista data-analyysia, jolla saatiin eristettyä informaatiota suurista datamassoista. Datan louhinta ei itsessään tuonut tekoälyyn uusia tekniikoita vaan osoitti suurten datamassojen tarpeellisuuden täsmällisen ymmärryksen ja mallien toteuttamisessa. Datan louhinta on terminä hieman virheellinen, sillä itse datan louhimisen sijaan kyseessä on datassa olevien, piilevien ominaisuuksien paljastamisesta koneoppimisen menetelmin. Esimerkiksi datamassan pohjalta opittu päätöspuu hyödyntää verkkoteoriaa ja koneoppimisalgoritmeja datan louhinnan periaattein. [32 s. 10, 179-180]

Toistaiseksi tekoälyn historia käsittää, hieman tulkinnasta riippuen, kaksi niin kutsuttua tekoälytalvea (”*AI winter*”), joiden aikana tekoälytutkimuksen rahoitus on vähentynyt merkittävästi. Ensimmäinen tekoälytalvi alkoi vuonna 1973 saaden alkusysäyksen professori James Lighthillin raportista, jossa käytännössä todettiin tekoälyn epäonnistuneen saavuttamaan sille asetetut odotukset ja tulevaisuuden vaikuttavan epävarmalta. [50] Rahoitus palasi normaaliksi ja kasvoi huomattavasti jälleen vuodesta 1980 alkaen tekoälyteollisuuden kasvaessa miljoonista miljardiluokan liiketoiminnaksi vuoteen 1988 mennessä. Toinen tekoälytalvi seurasi uutta korkeasuhdannetta, jolloin suuri osa tekoälyä kehittäneistä yhtiöistä kaatui niiden epäonnistuessa lunastamaan yliampuvia odotuksia ja lupauksia. [31 s. 24] Tekoälyn kehityksen historia on esitetty pinottuna aluekaaviona kuvassa 5.

Vuoden 2010 aikoihin tekoälyn tutkimus ja kehitys alkoivat lunastaa sille asetettuja odotuksia muun muassa syväoppivien neuroverkkojen onnistuttua oppimaan luokittelemaan valokuvia erittäin korkealla tarkkuudella. [32 s. 11] Alkuperäisten neuroverkkojen evoluutio syväoppimiseksi mahdollistui uusien algoritmien ja merkittävästi kasvaneen tietokoneiden laskentakapasiteetin myötä. [33 s.7] Ertel nimeää tämän käännekohdan kirjassaan tekoälyn vallankumouksen aluksi, sillä kuvien tulkinta mahdollistaa ympäristönsä kanssa vuorovaikutuksessa olevat älykkäät robotit kuten itseajavat autot ja takaisinkytketyt neuroverkot puheen tunnistuksen, jotka ovat olleet vaikeita tehtäviä ratkaista muilla menetelmillä [32 s. 11; 33 s. 7].



KUVA 5: Tekoälyn kehitys pinottuna aluekaaviona [mukailtu 32 s. 8; 31 s. 24; 50]

Nykyisellään tekoäly on eräänlainen laaja työkalulaatikko useiden, hyvinkin erilaisten ongelmien ratkaisemiseksi. Suurin osa työkaluista on pitkään jatkuneen kehitystyön ansiosta laadittu helppokäyttöisiin ohjelmistokirjastoihin, joita tässäkin opinnäytetyössä hyödynnetään kokeellisten mallien tuottamiseksi. Tekoäly on myös erittäin poikkitieteellinen tieteenala, sillä siihen liittyy paitsi logiikkaa, matematiikkaa, tilastotiedettä ja kontrolliteoriaa myös esimerkiksi kuvan tai signaalin käsittelyä, filosofiaa ja neurobiologiaa. Lisäksi onnistuneeseen tekoälyprojektiin liittyy kiinteästi ymmärrys käyttökohteen ominaisuuksista ja vaatimuksista. [32 s. 10]

2.1.3 Algoritmit

Tekoälyn toteutuksen ytimessä ovat algoritmit. Algoritmi on toimintaohje, jota seuraamalla voidaan ratkaista jokin laskennallinen ongelma. Algoritmille annetaan syöte, joka kuvaa ratkaistavaa ongelmaa tai sen tapausta, ja algoritmin tulee tuottaa vastaus annettuun syötteeseen. [49 s. 1] Ohjelmointikielellä yksinkertainen algoritmi, joka määrittää suurimman luvun listassa, joka sisältää lukuja, voisi olla pseudokoodina seuraavanlainen:

```
Algoritmi SuurinLuku
1          (Syöte) ListaLukuja = lista;
2          (Tulos) SuurinLuku = suurin;
3          jos(lista.koko = 0){}
4                    palauta SuurinLuku = null;
5          jos(lista.koko != 0)jokaista listan lukua kohden(luku)
6                                        {suorita
7                                        jos(luku  > suurin)
8                                        suurin = luku}
9                    palauta suurin;
```

Kyseinen algoritmi tarkastaa rivillä 3, onko lista tyhjä. Mikäli on, on tulos ”*null*”, joka palautetaan. Jos lista ei ole tyhjä, algoritmi käy listan läpi luku kerrallaan tallentaen suurimman vastaan tulleen luvun muuttujaan ”suurin”, ja päästyään listan loppuun palauttaa kyseisen luvun tuloksena.

Algoritmia voidaan kuvata joko ohjelmointikielellisesti kuten edellä, sanallisesti tai esimerkiksi vuokaaviona tai verkkona. Tekoäly on usein monimutkainen ja edistyksellinen algoritmi, joka käsittelee erittäin monimutkaisiakin syötteitä. Syöte voi olla esimerkiksi miljoonia parametreja käsittävä vektori [34 s. 16]. Sotilasnäkökulmasta havainnollistaen voidaan todeta, että esimerkiksi toimenpidekortti taistelualuksen laitekokeiluista on eräänlainen symboleja ja niiden tiloja käsittelevä algoritmi, vaikka sen operaatiot eivät näyttäydy välittömästi aritmeettisina. Laitekokeiluja suoritettaessa aluksen päällikkö antaa käskyn eli syötteen, jolloin henkilöstö toteuttaa toimenpidekortin toimenpiteet ja tuottaa tuloksen, esimerkiksi ilmoituksen siitä, että tehtäväpaikat on miehitetty ja alus on taisteluvalmis. Tässä mielessä taistelualuksen kaikki toimenpiteet ovat käsitettävissä algoritmisina, predikaattilogiikalla formuloitavina prosesseina.

Yksi älykkäiden ohjelmien automaattiseen tuottamiseen liittyvä algoritmijoukko ovat geneettiset algoritmit. Alun perin 1950-luvulla kone-evoluutioksi kutsuttu idea imitoi biologista evoluutiolla tuottamalla pieniä mutaatioita algoritmin toimintaan ja valiten näistä parhaiten toimivat uusien mutaatioiden pohjaksi, pyrkien tuottamaan hyvin toimivan ohjelman mihin tahansa ohjelmistollisesti ratkaistavaan ongelmaan. [31 s. 21] Geneettisellä algoritmilla pystytään nykyään optimoimaan ohjelman toiminta tietyn tavoitefunktion suhteen luomalla stokastisesti tuotetun populaation solmuista uusia lapsisolmuja ja valiten parhaiten tavoitefunktion toteuttavat lapset seuraavaan iteraatioon ja kohdistaen niihin mutaation seuraavia lapsisolmuja ja näiden iteraatiota varten. [31 s. 126-127]

2.2 Koneoppiminen

Koneoppiminen on tekoälyn osa-alue ja tietojenkäsittelytieteiden haara, joka on kehittynyt kaavatunnistuksen ja laskennallisen oppimisen teorian tekoälyn tutkimuksesta. Koneoppiminen on oppimista ja algoritmien rakentamista annetun datan pohjalta. Se on sellaisten algoritmien rakentamista, jotka oppivat data-aineistoista ja kykenevät löytämään ilmiöitä ja tekemään ennusteita oppimansa perusteella. Näin ollen koneoppiminen on prediktiivistä analytiikkaa. Koneoppimiseksi mielletyt metodit on jaettu lähtökohtaisesti kahteen alueeseen, ohjattuun ja ohjaamattomaan oppimiseen (*supervised learning* ja *unsupervised learning*). Riippuen käytettävästä lähdemateriaalista joko näiden osana tai rinnalla on vahvistusoppiminen. Näiden kolmen lisäksi on verrattain uutena ja suurimmassa nosteessa olevana osa-alueena kehittynyt syväoppiminen (*deep learning*). Kaikki neljä aluetta käsitellään tarkemmin omissa alaluvuissaan. [51] Laveasti ilmaistuna koneoppiminen voidaan määritellä laskennallisiksi menetelmiksi käyttää olemassa olevaa tietoa suorituskyvyn parantamiseksi tai tarkkojen prediktioiden tekemiseksi. [35 s. 1] Toisin muotoiltuna koneoppimisen on tarkoitus löytää datasta keskinäisiä suhteita ja kaavamaisuuksia, jotka ovat joko datan tunnistamattomia ominaisuuksia tai sopivat ennalta määritettyihin tavoitearvoihin. [51] On huomioitava, että kaikkiin malleihin liittyy epävarmuustekijöitä, jotka aiheuttavat virhettä tuloksessa. Tilastotieteilijä George E.P. Boxin toteamukseksi käsitetty ajatus, että ”*kaikki mallit ovat virheellisiä, mutta jotkut niistä ovat hyödyllisiä*”, kuvaa kärjistäen terveen kriittistä suhtautumista erilaisten mallien käyttöön. Lisäksi myöhemmin tehtäviä tarkasteluja varten on huomioitava, että vaikka sotilasympäristössä edellytetään periaatteessa absoluuttisen tarkkoja päätöksentekoprosessin välineitä, on ihmisen päätöksentekomalli yhtä altis erilaisille epävarmuustekijöille ja vinoumille kuin hyvä synteettisesti luotu malli.

Koneoppiminen koostuu kolmesta komponentista, jotka ovat data, hypoteesiavaruus tai -joukko sekä häviöfunktio, joka tunnetaan myös virhefunktiona. Data koostuu datapisteistä eli näytteistä, jotka käsittävät piirteitä, jotka kuvaavat kyseisen datapisteen ominaisuuksia, sekä mahdollisesti datapisteen tunnisteesta eli leimasta tai annotaatiosta, joka määrittää datapisteen luokan tai arvon. [35 s. 4] Yksinkertainen analogia löytyy biologiasta, jossa eläinlajiin liittyy sille tyypilliset piirteet, joiden kokonaisuus kuvaa eläimen kuuluvaksi tiettyyn lohkoon tai lajiin, kuten edellisen luvun asiantuntijajärjestelmän esimerkissä. Hypoteesiavaruus on datasta muodostettavissa olevien laskennallisten mallien kirjo, joilla datapisteiden piirteistä saadaan muodostettua arvio tai ennuste sen annotaatiosta. Häviö- tai virhefunktiolla arvioidaan kunkin hypoteesiavaruuden mallin laatua mittaamalla kyseisen mallin virhettä parhaan vaihtoehdon eli pienimmän virheen tuottamiseksi. Häviöfunktion toimintaperiaate on laskea virhe mallin muodostaman ennusteen ja sille tunnetun todellisen arvon välillä, ja mallin sovittaminen dataan tapahtuu valitsemalla sellaiset parametrit, joilla kyseinen virhe on minimoitu. [35 s. 4-5]

Koneoppimisella tuotettu malli on pohjimmiltaan yleistys saatavilla olevasta datasta, jolla on tarkoitus kyetä luokittelemaan mallille tuntemattomia näytteitä. Mikäli havainnoitava data-aineisto on esimerkiksi kahteen luokkaan lineaarisesti separoituva, voidaan kyseiseen dataan sovittaa malli, joka kykenee luokittelemaan aineiston selkeästi kahteen luokkaan, muodostaen binäärisen luokittajan. Usein data ei kuitenkaan ole täysin separoituva, jolloin jokaiselle hypoteesiavaruuden mallille muodostuu virheluokituksia, binäärisen luokittajan tapauksessa joko vääriä positiivisia tai vääriä negatiivisia näytteitä. Mallin tarkkuutta voidaan arvioida esimerkiksi *receiver operating charasteristic* (ROC) visualisoinnilla, jossa kynnysarvoihin sidottuna esitetään oikein positiivisesti luokiteltujen ja väärin positiiviseksi luokiteltujen datapisteiden suhde. Kuvaajan alle jäävä pinta-ala (*area under curve*, AUC)

kuvaa mallin tarkkuutta. [35 s. 7, 255-256] Mallin relevanssia voidaan arvioida tarkkuuden ja takaisinkutsun mittareilla. Tarkkuus (”*precision”*) kertoo oikein positiivisiksi luokiteltujen datapisteiden ja kaikkien positiiviseksi luokiteltujen datapisteiden suhteen, kun taas takaisinkutsu kertoo oikeiden positiivisten osuuden oikeista positiivisista ja vääristä negatiivisista näytteistä. Usein koneoppimisessa käytetään toista tarkkuusmäärettä (”*accuracy”*), joka kertoo oikein luokiteltujen datapisteiden määrän koko luokitellusta aineistosta. Tämä tarkkuus riippuu kuitenkin huomattavasti aineiston koostumuksesta, sillä aineistossa, jossa muita vaihtoehtoja on esimerkiksi viisi prosenttia, saavuttaa malli 95% tarkkuuden luokittelemalla kaikki näytteet suurimman näyteluokan mukaisesti.

Seuraavaksi tarkastellaan koneoppimisen eri osa-alueet yksinkertaisin esimerkein sotilaallisiin käyttötarkoituksiin sitoen teorian yleiskäsityksen muodostamiseksi ja mahdollisten käytännön sovellusten havainnollistamiseksi. Eritellyistä osa-alueista on huomioitava, etteivät niiden rajat ole näin selväpiirteisiä käytännössä.

2.2.1 Ohjattu oppiminen

Ohjattu koneoppiminen tarkoittaa sitä, että käytetty data-aineisto on annotoitu jokaisen datapisteen osalta kuulumaan johonkin luokkaan tai kategoriaan datan käsittelijän toimesta. [34 s. 4] Klassisessa esimerkissä datapiste voi olla vektori, jonka piirteet ovat kiinteistön ominaisuuksia kuten pinta-ala, postinumeroalue, kerroslukumäärä, tontin omistussuhde ja niin edelleen, ja datapisteen annotaatio (”*label*”) olisi kiinteistön toteutunut kauppahinta. Toteutuneista kiinteistökaupoista voidaan kerätä laaja aineisto, jossa datapisteiden piirteet korreloivat tiettyyn hintaan. Näin voidaan laskea esimerkiksi lineaarisella regressiolla tärkeimmistä piirteistä malli, joka kuvaa eri muuttujien vaikutusta oletettavaan myyntihintaan. Mallilla pystytään ennakoimaan aineistoon kuulumattoman, uuden myyntikohteen kauppahintaa. Vastaavasti sotilassovelluksissa pystytään muodostamaan datapisteeksi esimerkiksi vektori ilmamaalin piirteistä. Tutkan havaitsemalla ilmamaalilla on lentonopeus, lentokorkeus, etäisyys tunnettuun lentokäytävään, käännösnopeus, etäisyys ja suuntapoikkeama lähimmistä ilmamaaleista, korkeuden vaihtelun nopeus, tutkasignaalin voimakkuus ja niin edelleen, jotka korreloivat sille tilannekuvaoperaattorin antamaan annotaatioon, kuten neutraaliin tunnisteeseen. Sotilaskohteiden piirteet poikkeavat oletettavasti siviilikohteiden lentoprofiileista ainakin joidenkin parametrien osalta, jolloin voidaan aikaisempien tilannekuvaoperaattorien tunnistaman aineiston pohjalta kouluttaa malli, joka tunnistaa ilmamaaleja automaattisesti ajamalla tilannekuvaan ilmestyviä datapisteitä mallin läpi ja saamalla niille luokituksen tai todennäköisyyden tietylle luokitukselle.

Yleisimmin käytettyjä ohjatun oppimisen algoritmeja ovat lineaarinen regressio, SVM (*support-vector machine*) eli tukivektorikoneet, päätöspuut, logistinen regressio, naiivi Bayes, päätöspuualgoritmit sekä neuroverkot. Tässä osiossa havainnollistetaan ohjattua oppimista hyödyntämällä erilaisia regressioita ja päätöspuuoppimista niiden ymmärrettävyyden ja visualisoitavuuden vuoksi.

Aiemmin mainittu yhden selittävän muuttujan lineaariregressiomalli voidaan kirjoittaa kaavana muodossa

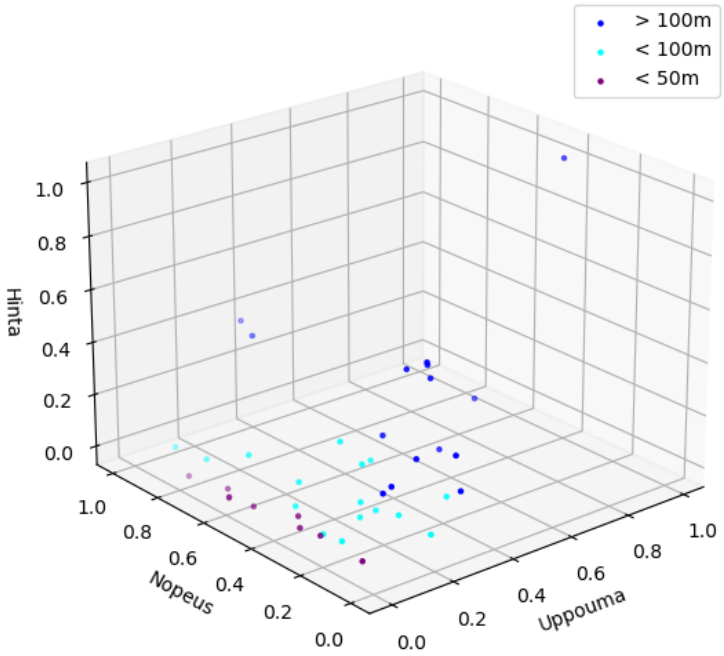
$$Y = \alpha + \beta x + e$$

jossa Y on datapisteen annotaatio eli esimerkiksi sota-aluksen hinta, α on vakio, joka määrittää y-akselin leikkauspisteen eli arvon painokertoimien ollessa nolla, ja β on muuttujan x, esimerkiksi uppouman, pituuden tai nopeuden, painokerroin. Satunnaismuuttuja *e* on normaalijakautunut virhemuuttuja odotusarvolla 0, jonka varianssi estimoidaan datasta. [52 s. 520, 564]

Usean selittävän muuttujan lineaariregressiomalli olisi vastaavasti

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \dots \beta_n x_n + e$$

jossa muuttujat $\{x_1, x_2 \dots x_n\}$ ovat tässä esimerkissä kiinteistön parametreja ja vastaavasti $\{\beta_0, \beta_1 \dots \beta_n\}$ regressioparametrien painokertoimia, joiden arvo määräytyy regressiosuoran sovittamisesta aineistoon. [52 s. 577]



KUVA 6: Sota-alusten ominaisuuksia kolmiulotteisessa piirreavaruudessa [Liite 1]

Kuvassa 6 on visualisoitu Winstead P.J. opinnäytetyössään listaamien sota-alusten piirreavaruutta uppouman, nopeuden ja hinnan osalta [53]. Kuvan tarkoitus on havainnollistaa, mitä tarkoitetaan piirreavaruudella ja sen separoitavuudella. Pisteiden värit indikoivat eri runkopituuksia. Visuaalisesti arvioimalla huomataan kevyiden ja lyhyiden alusten olevan edullisimpia, hinnan kasvaessa uppouman ja pituuden funktiona. Kun datapisteitä on lukuisia, saadaan koneoppimisella laskettua esimerkiksi sellainen regressio, jonka neliöityjen etäisyyksien summa datapisteistä on minimoitu, eli se kuvaa koko data-aineistoa mahdollisimman pienellä koko aineiston kattavalla ja kyseiselle mallille sovitetulla laskennallisella virheellä, tässä tapauksessa ennakoiden esimerkiksi uuden alusluokan hintaa sen uppouman, pituuden ja suunnitellun nopeuden perusteella.

Erilaiset regressiomallit ovat tunnettuja tilastotieteellisiä menetelmiä, ja tilastotieteen ja koneoppimisen osa-alueet ovat osittain päällekkäisiä eivätkä näiden rajapinnat ole yksiselitteisiä. Koneoppiminen poikkeaa tavanomaisesta tilastotieteestä pääasiassa siinä, että tilastotieteessä valitaan tai oletetaan malli, joka sopii dataan, kun taas koneoppimisessa hyödynnetään suuria määriä dataa siihen sopivan mallin luomiseksi. [52 s. 520, 693] Toisaalta koneoppimisessa datasta ja koneoppimisongelmasta muodostuu hypoteesiavaruus, johon pyritään valitsemaan parhaiten soveltuva malli.

Seuraavissa kuvissa havainnollistetaan simuloidulla ilmamaalidatalla visualisoituna yksinkertaisilla 2-ulotteisilla kuvaajilla sekä dataan sovitetuilla regressiomalleilla. Simuloitua ilmamaalidataa tuotettiin ensin 40 datapistettä ja

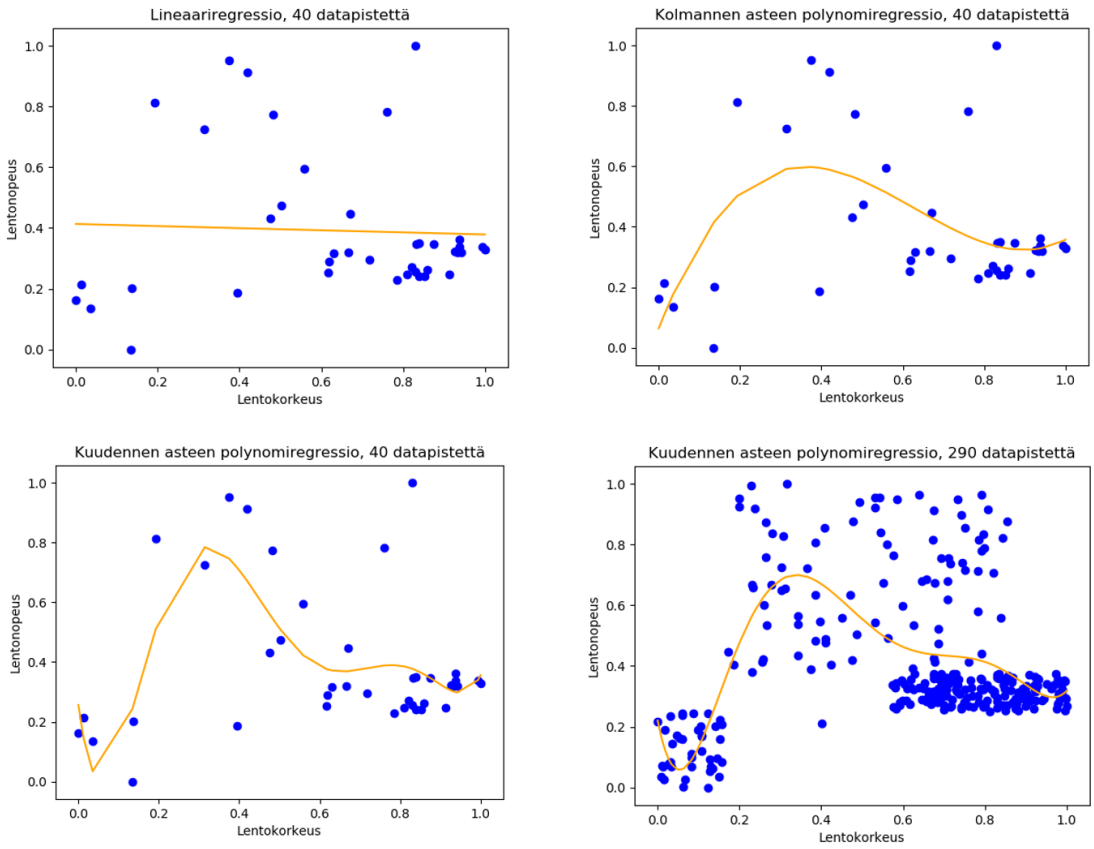
lopuksi 290 datapistettä, muodostamalla datapisteiden piirteiden arvot satunnaisgeneroimalla arvoja tyyppikohtaisella vaihteluvälillä kolmesta erilaisesta ilmamaaliluokasta, joiksi valittiin reittikone, LSF (”*low slow flyer”*) sekä hävittäjäpommittaja.

Suunnan ja paikkatiedon osalta voisi mallintaa esimerkiksi sitä, lentääkö maali muodostelmassa tai lähellä tunnettua lentokenttää, mutta tämän havainnollistuksen piirissä kaksiulotteisen piirreavaruuden muuttujiksi valittiin yksinkertaisuuden vuoksi nopeus ja korkeus. Visualisoituna ja arvoiltaan normalisoituna vaihteluvälille [0,1] kyseisten muuttujien keskinäiset suhteet datassa ja niihin sovitettu lineaariregressio näyttävät kuvan 6 vasemman yläkulman mukaisilta. Normalisointiin käytetty metodi käyttää kaavaa 2.

$$x_{norm} = (\frac{x-\min(x)}{\max(x)-\min(x)})(\max(x) - \min\,(x)\,) + min\,(x)$$

KAAVA 2: Normalisointi, jossa x on yksittäinen piirrevektori

Datan normalisointi samalle vaihteluvälille on toteutettu kaikissa tämän kappaleen visualisoinneissa piirreavaruuden ulottuvuuksien yhtenäistämiseksi, pois lukien päätöspuuoppiminen, ettei esimerkiksi lentokorkeuden suuremmat numeeriset arvot vinouta kyseisen piirteen merkitsevyyttä ja kuvan mittakaavaa. Kuvan 7 kuvaajiin on valittu x-akselin muuttujaksi eli selittäväksi muuttujaksi lentokorkeus ja y-akselin selitettäväksi muuttujaksi lentonopeus. Käytännössä malli selittää kohteen lentonopeutta sen havaitun korkeuden perusteella, mikä ei ole itsessään erityisen hyödyllistä. Tässä kohtaa datan visualisoinnissa ja regressiosuoran sovittamisessa luokittelijan mallintamisen sijaan tarkastellaan datan ominaisuuksia, eli sitä, onko joidenkin kahden muuttujan välillä lineaarinen tai polynomista riippuvuutta.



KUVA 7: Simuloitu ilmamaalidata ja siihen sovitetut regressiokuvaajat [Liite 1]

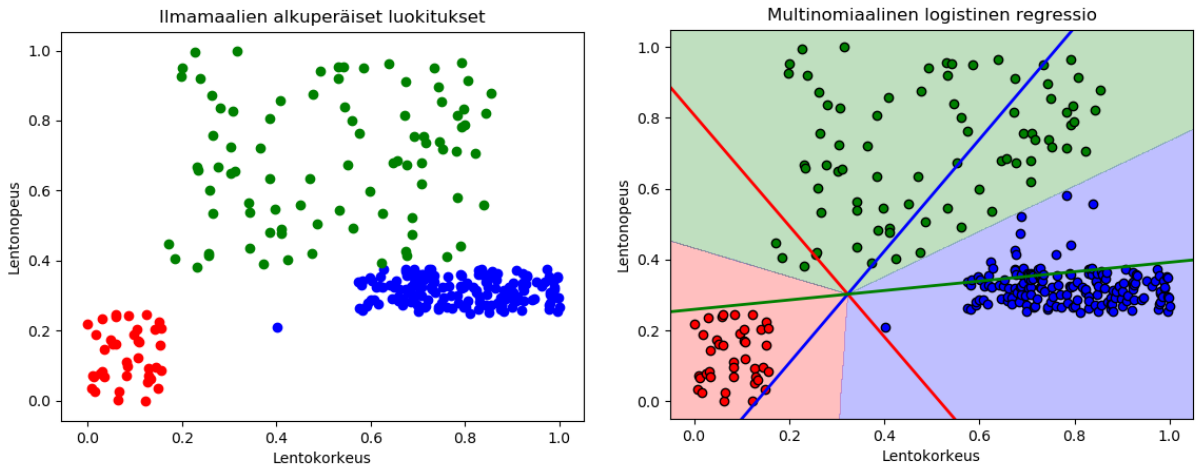
Kuvan 7 vasemmassa yläkulmassa nähdään, että lineaariregressiosuora kuvaa huonosti näiden simuloitujen ilmamaalien lentonopeuden ja -korkeuden suhdetta, sillä suuri osa datapisteistä on kaukana regressiosuorasta. Datapisteiden tihein massa koostuu reittikoneiden toisiaan muistuttavista korkeus- ja nopeusprofiileista samalla kun mielenkiintoiset ja harvinaisemmat maalit jäävät pääsääntöisesti mallin reuna-alueille. Seuraavissa kuvissa samaan dataan on sovitettu polynominen regressioviiva, joka kuvaa dataa huomattavasti paremmin. Polynominen regressio muodostuu kertomalla piirrekartta (”*feature map*”, ϕ) skalaarimuuttujilla $x^{(i)}$:

$$y^i = \phi\left(x^{(i)}\right) = ((1, x^{(i)}, \dots, (x^{(i)})^n)^T \in \mathbb{R}^{n+1}$$

KAAVA 3: Polynominen regressio piirre- tai ominaisuuskartan ja skalaarimuuttujien tulona

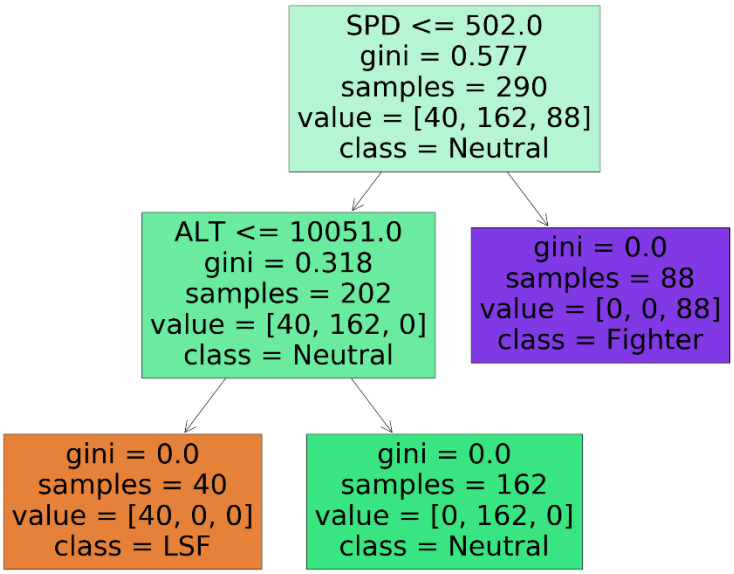
Skalaarimuuttujilla kertomisen jälkeen sovitetaan lineaarinen regressio näin saatuihin, muunnettuihin piirrevektoreihin [34 s. 32-33]. Toisin sanoen polynominen regressio mallintaa ei-lineaarisista riippuvuutta selittävän arvon $x^{(i)}$ ja sitä vastaavan selitettävän arvon y^i välillä estimoimalla tuntemattomat parametrit datasta lineaarisesti. Polynomin asteet määrittävät käytännössä mallin kompleksisuuden eli optimoitavien parametrien määrän. Mikäli asteita nostetaan ja optimoitavien parametrien määrää siten lisätään, kuvaa saatava malli dataa tarkemmin. Kuvan 7 oikeassa ylälaidassa ja vasemmassa alalaidassa on luotu kolmannen ja kuudennen asteen polynomiset regressiot, joista kolmannen asteen polynomi kuvaa dataa varsin luontevan näköisesti, kun taas kuudennen asteen polynomi on liian kompleksinen ja sisältää yliopitun näköisiä, teräviä muutoksia regressioviivassa harvassa olevien datapisteiden ympärillä. Lisättäessä datapisteiden määrää myös kuudennen asteen polynomiregression oppiminen tuottaa paremmin dataa kuvaavan, niin sanotusti istuvamman regressioviivan.

Kuvasta 7 nähdään, että polynominen regressio saadaan vastaamaan valittujen piirteiden keskinäisiä suhteita huomattavasti lineaarista regressiota paremmin, mutta suuren hajonnan omaavat datapisteet jäävät silti kauas mallin regressioviivasta. Ylipäätään, jotta lineaarista tai polynomista regressiota voidaan käyttää prediktorina tai luokittelijana tulisi annotaation olla jatkuva muuttuja, sillä lineaarinen ja polynominen regressio palauttavat arvoinaan jatkuvia reaalilukuja annettujen piirteiden pohjalta. Mikäli simuloidusta datasta haluttaisiin muodostaa varsinainen diskreetti luokittelija, tulisi sen olla esimerkiksi moniluokkainen logistinen regressio. Tällöin maalidatan diskreettiä numeerista annotaatiota voitaisiin hyödyntää suoraan, kuten on visualisoitu kuvaan 7. Logistinen regressio on binäärinen luokittaja, joka kuvaa piirrevektorin $w \in \mathbb{R}^n$ kahteen luokkaan, esimerkiksi $y \in \{0,1\}$ [34 s. 34]. Multinomiaali logistinen regressio on binäärisen logistisen regression laajennus, joka kuvaa piirrevektorin annettuun määrään luokkia, tässä tapauksessa kolmeen luokkaan. Binäärisen luokittelijan tapauksessa muodostuu yksittäinen ”*decision boundary*” kohtaan, jossa luokitus lähtökohtaisesti muuttuu, kun taas kolmen luokan logistisen regression tapauksessa tällaisia päätösrajoja muodostuu jokaisen luokan välille eli yhteensä kolme kappaletta, kuten kuvassa 8 näkyy. Kuvassa olevat viivat kuvaavat ”yksi vastaan loput” päätösrajoja, kun taas väritetyt alueet oppimisen kouluttamaa päätösrajojen rajaamaa luokitusaluetta. Kyseisen mallin koulutustarkkuus on 96.9%.



KUVA 8: Multinomiaalinen logistinen regressio [Liite 1]

Yksi menestyneimmiksi osoittautuneita ohjatun oppimisen menetelmiä on päätöspuuoppiminen, jossa algoritmi oppii datasta päätöspuurakenteen, jonka päätössolmut kuvaavat datan ominaisuuksia. Päätöspuuoppiminen on funktio, joka ottaa syötevektorin ja palauttaa tuloksena joko diskreetin tai jatkuvan muuttujan. Oppiminen tapahtuu suorittamalla sarja testejä, jotka muodostavat puun solmut. Jokainen solmu testaa yhtä vektorin piirrettä, ja siitä lähtevät kaaret saavat jonkin mahdollisen kynnysarvon kyseiselle piirteelle, muodostaen polun seuraavaan päätössolmuun. [31, s. 697-698]



KUVA 9: Simuloidusta datasta opittu päätöspuu [Liite 1]

Kuvassa 9 on visualisoitu liitteen 1 ohjelmalla tuotettu päätöspuu samasta simuloidusta ilmamaalidatasta kuin aiemmissakin visualisoinneissa. Python scikit-kirjaston päätöspuuoppiminen laskee jokaiselle solmulle testatun piirteen kynnysarvon jakamalla datapisteet osajoukkoihin muodossa $Q_m \rightarrow Q_m^{vasen}(\theta) = \left\{\frac{x,y}{x_j} \leq t_m\right\}, Q_m^{oikea}(\theta) = Q_m \backslash Q_m^{vasen}(\theta)$, jossa kandidaatti $\theta = (j, t_m)$, j on piirre ja t_m on kynnysarvo. Kandidaatin laatu solmussa m lasketaan häviöfunktiolla $H()$ yhtälöstä $G(Q_m, \theta) = \frac{N_m^{vasen}}{N_m} H\left(Q_m^{vasen}(\theta)\right) + \frac{N_m^{oikea}}{N_m} H\left(Q_m^{oikea}(\theta)\right)$, jossa häviöfunktio valitaan sen mukaan, lasketaanko regressiota vai luokittelua. Kuvan 8 tapauksessa on käytetty Gini-funktiota

luokittelijan kouluttamiseksi, jonka kaava on $H(Q_m) = \sum_k p_{mk}(1 - p_{mk})$, jossa p_{mk} on annotaatioon y täsmäävien näytteiden määrä kerrottuna tekijällä $\frac{1}{N_m}$, jossa N_m on näytteiden määrä [54]. Käytännössä algoritmi siis etsii parametrit, joilla mahdollisimman suuri määrä näytteistä jakautuu oikeaan luokitukseen jatkettaessa puussa seuraaviin päätössolmuihin, eli oikeaan ja vasempaan lapseen. Kuvassa 8 tämä on nähtävissä siten, että ensimmäisessä solmussa näytteet jaetaan neutraaleihin ja hävittäjiin sen mukaan, että alle nopeuden 502 datapisteet ovat neutraaleja ja sen yli hävittäjiä. Häviöfunktion tuottama gini-luku kertoo solmussa väärinluokiteltujen näytteiden osuuden, joka on ensimmäisessä vasemmassa lapsisolmussa 0.318, sillä kaikki LSF-maalit ovat väärinluokiteltuja tässä kohtaa. Solmujen väri määräytyy siinä jäljellä olevien luokkien osuuksien perusteella. Algoritmi jatkaa häviöfunktion minimoimista etsimällä seuraavan kynnysarvon parametreista, jonka perusteella luokitus jakautuu jälleen vasempaan ja oikeaan solmuun saavuttaen gini = 0 arvon ja luokitellen jokaisen maalin oikein. Näin ollen simuloidut ilmamaalit ovat luokiteltavissa seuraamalla polkurakennetta intuitiivisesti havainnon piirteiden perusteella. Päätöspuuoppimisen kiistattomia hyötyjä ovat visualisoitavuus ja ymmärrettävyys, sillä verrattuna monimutkaisten syvien neuroverkkojen tekemään, moniparametriseen päättelyyn, on päätöspuuoppimisella tuotetun tekoälymallin päätöksenteon ymmärtäminen erittäin yksinkertaista. Kuitenkin monimutkaisten syötteiden tapauksessa puurakenne kasvaa väistämättä valtavaksi. Toinen miellyttävä käytännön ominaisuus on, ettei päätöspuuoppimisen tapauksessa tarvitse käyttää erityisesti aikaa datan esikäsittelyyn, sillä oppimisalgoritmi ei kaipaa esimerkiksi datan normalisointia.

Yllä kuvatuissa esimerkeissä algoritmien kouluttaminen on suoraviivaista, sillä data on simuloitua, valmiiksi annotoitua ja tuotettu välittömästi käytettävään muotoon. Usein kerätty data edellyttää käytettävästä metodista huolimatta analyysia ja muokkaamista käsiteltävään muotoon erilaisten toimenpiteiden mahdollistamiseksi, mutta ohjattu oppiminen edellyttää lisäksi jonkinasteista asiantuntijuutta ja usein mittavaa työtä datapisteiden annotoimiseksi, ellei annotoitua dataa tuoteta osana normaalia toimintaa. Esimerkiksi kuvailtu ilmamaalien tunnistamismallinnus edellyttää, että asiaan perehtynyt ammattilainen on tunnistanut tuhansia maaleja taltioituun tilannekuvaan oikein, jotta käytössä on annotoitua dataa. Koneoppimisalgoritmi oppii taltioidusta maalidatasta yhdistämään oikeat piirteet oikeaan tunnistukseen. Tämä luonnollisesti vaikeuttaa datan hyödyntämistä, jos kerätty data edellyttää merkittävää määrää työtä ollakseen hyödyntämiskelpoista. Datan laatua ja data-analyysia käydään soveltuvin osin läpi opinnäytetyön liitteissä.

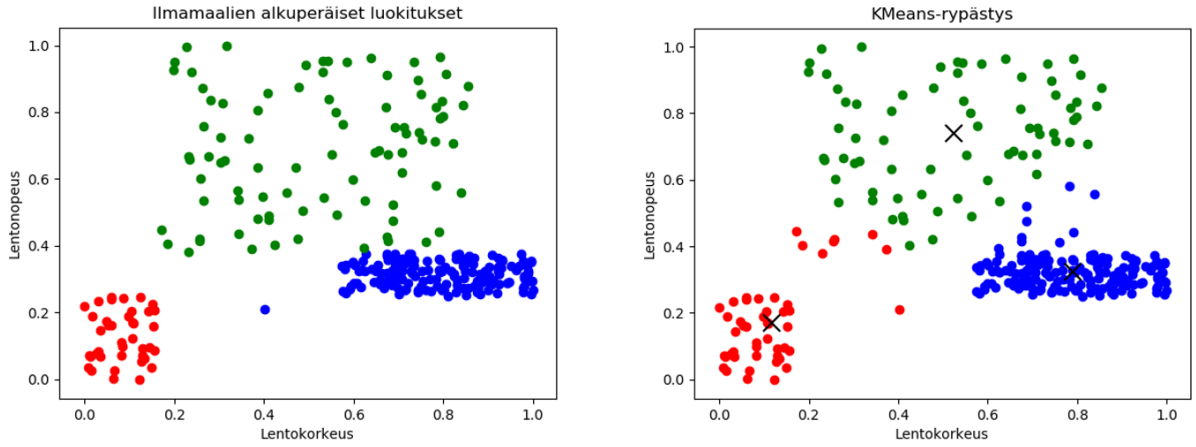
2.2.2 Ohjaamaton oppiminen

Siinä missä ohjattu oppiminen edellyttää annotoituja datapisteitä, ei ohjaamaton oppiminen edellytä tietoa datapisteiden luokittelusta annotaation muodossa. Ohjaamattomassa oppimisessa, jota kutsutaan myös datan louhinnaksi, pyritään löytämään datassa piileviä ominaisuuksia esimerkiksi siten, että algoritmi määrittää datapisteiden läheisyyttä toisiinsa ja muodostaa jossain piirreavaruudessa $\mathbb{R}^n$ lähekkäin toisiinsa nähden sijoittuneista datapisteistä ryppäitä. [33 s. 331] Mikäli data-aineistossa muodostuu toisistaan irrallisia, jollain tavalla erotettavia ryppäitä, saadaan ohjaamattomalla oppimisella laskettua malli, joka osaa kyseisessä piirreavaruudessa luokitella sille syötetyn datapisteen kuuluvaksi johonkin luokkaan eli ryppääseen. Luokittelu perustuu ryppäiden laskennallisiin keskipisteisiin ja luokiteltavan, uuden datapisteen etäisyyteen kyseisistä keskipisteistä. Rypästämiseen käytettäviä metodeja on kahdenlaisia, niin sanottuja pehmeitä ja kovia metodeja. Kovat metodit luokittelevat kunkin datapisteen tasan yhteen luokkaan, jolloin datapiste kuuluu siihen luokkaan, jonka keskipiste tai muu etäisyysarvo on lähimpänä datapisteen sijaintia kyseisessä avaruudessa. Pehmeät metodit voivat luokitella datapisteen kuuluvaksi

useaan eri luokkaan eri asteisella määrällä yhteenkuuluvuuden luotettavuutta. [34 s. 93-94] Aineistosta muodostettavien ryppäiden määrä voidaan asettaa ennalta tai käyttää metodia, joka määrittää itse muodostuvien ryppäiden määrän.

Ohjaamatonta oppimista voidaan käyttää paitsi erilaisten luokittelijoiden mallintamiseen, myös datan esikäsittelyyn, kun halutaan mallintaa esimerkiksi sitä, onko datasta mahdollista muodostaa luotettavia luokittelijoita muilla menetelmillä. Mikäli datapisteiden piirteet ovat eri luokissa hyvin samankaltaiset, ei millään oppimismenetelmällä muodostu luotettavaa mallia esimerkiksi ilmamaalien luokittelemiseen. Liitteessä 1 on tarkasteltu simuloitua dataa PCA (”*principal component analysis*”) menetelmällä, jolla voidaan määrittää ohjaamattomasti viisiulotteisen piirreavaruuden oleellisimmat piirteet. Sen avulla lasketut kaksi oleellisinta piirrettä selittävät 51.2% datan varianssista, kun kolme oleellisinta piirrettä selittävät jo 73.3% varianssista. Kahden piirteen selittävyys on nähtävissä myös datan visualisoinneista, jota varten PCA on hyödyllinen työkalu silloin, kun data on moniulotteista ja sen visualisointi ilman vastaavaa käsittelyä mahdotonta.

Seuraavassa on visualisoitu samaa simuloitua maalidataa yksinkertaisella ohjaamattoman oppimisen kovalla metodilla, k-means algoritmilla, joka pyrkii jakamaan datapisteet ennalta annettuun määrään ryppäitä, tässä tapauksessa kolmeen, sillä tiedämme datan sisältävän havaintoja kolmesta eri luokasta.



KUVA 10: K-means algoritmilla rypästetty ilmamaalidata ja ryppäiden keskipisteet [Liite 1]

Vertaamalla rypästettyä ilmamaalidataa alkuperäisiin annotaatioihin, jotka näkyvät vasemmanpuoleisessa kuvaajassa kuvassa 10, huomataan, että algoritmin laskemat ryppäät vastaavat varsin tarkasti alkuperäisiä luokituksia. Tulos ei yllätä, sillä datassa on selkeästi havaittavissa olevat ryppäät. Näin ollen tarkastellut piirteet eroavat luokkien välillä riittävästi toisistaan, mahdollistaen todennäköisesti hyvin toimivan luokittelijan kouluttamisen. On huomioitava, että kyseinen data on simuloitu nimenomaan havainnollistamistarkoituksessa, eikä data vastaa todellisuutta ilmamaalien luokittelemisessa. Lisäksi on huomionarvoista, ettei k-means algoritmi yhdistä datapisteitä annotaatioon samalla tavalla kuin ohjatun oppimisen metodit. Visuaalisesti vertaillen on kuitenkin nähtävissä, että joitain maaleja on luokiteltu väärään ryppääseen. Johtuen algoritmin toimintaperiaatteesta kyseisten maalien etäisyys niin sanotusti väärän ryppään laskennalliseen keskipisteeseen, jotka on visualisoitu kuvassa, on pienempi kuin etäisyys muihin keskipisteisiin, johtaen virheeseen rypästyksessä. Mikäli sama mallinnus toteutettaisiin muulla metodilla, voitaisiin arvioida maalien kuulumista kuhunkin ryppääseen eri periaatteella tai joustavammin. Lisäksi muuttujien lisääminen luokitteluun lisäisi ulottuvuuksia, joissa etäisyyttä mitataan, jolloin mallista tulisi mahdollisesti tarkempi, mutta sen visualisoinnista nopeasti mahdotonta.

Ohjatun ja ohjaamattoman oppimisen lisäksi on olemassa ”*semi-supervised*” eli vapaasti käännettynä puoliohjattua oppimista. Kyseisessä metodiikassa osa datamassasta on annotoitu, mutta suurin osa ei, jolloin algoritmia ohjataan tiettyyn suuntaan kuitenkin mahdollistaen ohjaamattoman datan ominaisuuksien tunnistamisen. [32 s. 236; 36 s. 236-237]

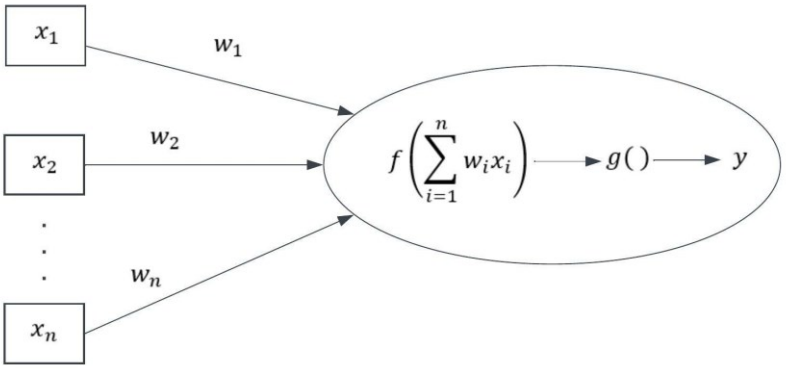
2.2.3 Vahvistusoppiminen

Vahvistusoppimisessa koneoppimisalgoritmi tuottaa toimenpiteen ympäristönsä kanssa luoden ajan kuluessa sarjan toimintoja ja niiden tuloksia. Ympäristöllä voidaan tarkoittaa tässä tapauksessa esimerkiksi yksinkertaista videopeliä tai fyysistä maailmaa. Tehdyt toimenpiteet vaikuttavat ympäristöön johtaen palautteeseen algoritmille; videopelin tapauksessa esimerkiksi pisteiden kertyminen olisi positiivinen palautesilmukka ja pisteiden menettäminen negatiivinen palautesilmukka. Algoritmin tehtävä voisi olla, videopeleille tyypillisesti, maksimoida pisteiden määrä. Tärkeä ominaisuus on iteraatiot, joita algoritmi suorittaa hyödyntääkseen saamaansa palautetta muokatakseen toimintaansa. [33 s. 333] Vahvistusoppimisen voidaan nähdä sisältävän tekoälyn koko olemus oppivan agentin asettamisesta ympäristöön ja sen menestyksekkäästä toiminnasta kyseisessä ympäristössä. Vahvistusoppimista hyödyntämällä voidaan esimerkiksi opettaa tekoäly navigoimaan itsenäisesti alusta simulaatioympäristössä ilman mahdottoman kompleksia ohjelmointia antamalla sille palautteen onnistuneesta navigaatiosta ja vastaavasti karilleajoista ja muista epäsuotuisista toimenpiteistä navigaation aikana. [31, s. 831] Vahvistusoppimisen periaatteessa on yhteneväisyyksiä kontrolliteorian mukaiseen dynaamiseen järjestelmään, jonka toiminta mukautuu palautesilmukan tuottaman tiedon perusteella. Kontrolliteorian on kuitenkin tarkoitus nimensä mukaisesti mahdollistaa dynaamisen järjestelmän hallinta sen toiminnan aikana, kun järjestelmään kohdistuu toiminnanaikaisia muuttuvia syötteitä. Esimerkiksi mekaanisen järjestelmän tapauksessa tällaisia ovat olosuhteiden aiheuttamat fysikaaliset voimavektorit. [55] Vahvistusoppimisessa palaute, positiivinen tai negatiivinen, annetaan sen sijaan tietyn suoritusketjun päätteeksi, jolloin opittava ja palautteen avulla optimoitava suorite voi olla esimerkiksi lennokin lentäminen rakennuksen sisätilojen läpi. Lopputuloksena on kyseiseen suoritteeseen ja palaute-ehtoon kuten aikaan optimoitu toimintamalli. Kontrolliteorian ja tekoälyn merkittävin ero on loogisen päättelyn ja laskennan hyödyntämisessä, joilla tekoälyssä ratkaistaan ongelmia, jotka eivät ole kuuluneet kontrolliteorian alaan. [31, s. 15]

2.2.4 Syväoppiminen

Kappaleessa 2.2.2 todetusti tekoälyn kehittäminen lähti liikkeelle inhimillisen logiikan ja aivojen rakenteen mallintamisesta, jossa jo 1940-luvulla päädyttiin neuroverkkojen mallintamiseen. Monikerroksinen eteenpäinkytketty neuroverkko (”*multilayer feedforward neural network*”) on niin sanottu universaali approksimaattori. Tämä tarkoittaa, että neuroverkko pystyy approksimoimaan minkä tahansa mitattavissa olevan funktion millä tahansa halutulla tarkkuudella, kunhan sillä on riittävästi neuroneja piilotetuissa kerroksissa, mallilla tapahtuu riittävä oppiminen eli sitä koulutetaan riittävän monta kertaa riittävän suurella määrällä dataa, ja on olemassa jonkinlainen deterministinen suhde syötteen ja tulosteen välillä [56]. Keinotekoinen neuroverkko (*Artificial Neural Network*, ANN) koostuu suuresta määrästä keinotekoisia neuroneja, joiden käyttäytyminen pohjautuu karkeasti oikeiden neuronien tapaan kommunikoida toistensa kanssa ihmisen aivoissa. Jokainen neuroni on yhteydessä useaan muuhun neuroniin ja vierekkäisten neuronien välisien yhteyksien aktivointia voidaan vahventaa tai vaimentaa. Verkko

koostuu syötekerroksesta (*input layer*), mistä nimensä mukaisesti annetaan syöte verkolle, jonka jälkeen syöte kulkee piilotettujen kerrosten läpi päätyen tulostukseen (*output layer*). [33 s. 7; 36 s. 2-3]



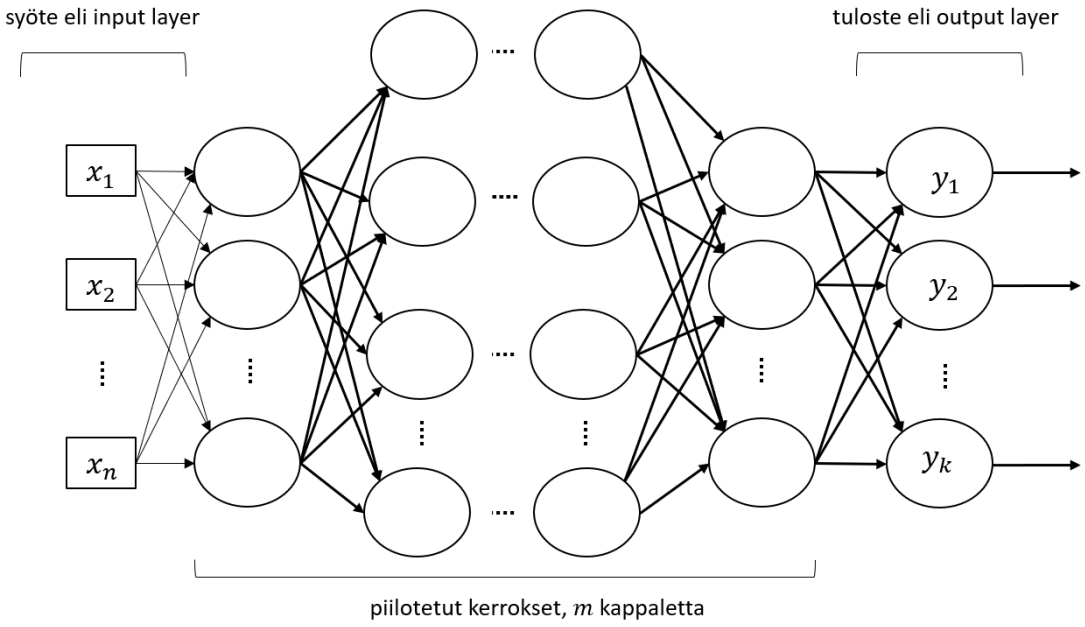
KUVA 11: Yksittäinen keinotekoinen neuroni (perseptroni) [mukailtu 31 s. 728; 33 s. 390]

Kuvassa 11 on havainnollistettu yhtä keinotekoista neuronia Neapolitanin ym. ja Russel ym. oppikirjojen mallien pohjalta. Biologisessa neuronissa eli hermosolussa on dendriittejä, jotka liittyvät solun tumaan ja joista välitetään viejähaarakkeilla impulsseja niiden terminaaleihin eli synapseihin, joista signaali jatkaa matkaa seuraavan hermosolun viejähaarakkeeseen, mikäli dendriitin välittämän impulssin voimakkuus on riittävän vahva tuottamaan synapsissa kyseisen siirtymän eli aktivaation. Samankaltaisella toimintaperiaatteella keinotekoisessa neuroverkossa on analogisesti ”dendriittejä”, kuvan 11 tapauksessa vasemmalla olevat laatikot x_1, x_2, … x_n eli syötevektorin $v = (x_1, x_2, \dots, x_n)$ arvot varsinaiselle neuronille. Syötteen tulos lasketaan painokertoimilla $\{w_1, w_2, \dots, w_n\}$ neuronin ”tumalle” eli aktivaatiofunktiolle g, joka saa käsiteltäväkseen vektorin arvojen painotetun summan $\sum_i^n w_i x_i$ tuottaen arvon *y*, joka on tämän yhden neuronin tuloste. [31 s. 728; 33 s. 389-390] Yhdellä neuronilla voidaan toteuttaa binäärinen luokittelija, perseptroni, joka on ensimmäisiä sovelluksia keinotekoisille neuroneille. Sen aktivaatiofunktio tuottaa piirrearvoja esimerkiksi kaavan 4 mukaisesti.

$$y = 1, kun\ f\left(w_0 + \sum_i^n w_i x_i\right) > 0, muutoin\ y = -1$$

KAAVA 4: Perseptronin aktivaatiofunktio [33 s. 390]

Perseptroni on binäärinen, lineaarinen luokittelija. Jotta saadaan mallinnettua monimutkaisempia piirreavaruuksia, koostuu ANN-struktuuri useista keinotekoisista neuroneista ja erilaisista aktivaatiofunktioista. Termi syväoppiminen juontaa neuronien muodostamien kerrosten määrästä, joita voi olla lukuisia verrattuna alkuperäisiin ANN-rakenteisiin. Suurempi määrä piilotettuja kerroksia, jotka voivat sisältää eri määriä neuroneja, on tullut mahdolliseksi kehittyneempien algoritmien ja kasvaneiden laskentakapasiteettien ansiosta. Kerrokset voivat olla myös erikoistuneita ja eri tavoin kytkettyjä. Kuvan 11 rakenteessa on vain täydellisesti kytkettyjä, tiiviitä kerroksia eteenpäinkytketyssä neuroverkossa, sillä jokainen neuroni on kytketty jokaiseen seuraavan kerroksen neuroniin. Erikoistuneet kerrokset ja erilaiset arkkitehtuurit voivat rajoittaa kytkentöjen määrää muodostaen siten pienemmän hakuavaruuden piirteiden painokertoimille. Syväoppimisen neuroverkkoja voidaan käyttää niin valvomattoman kuin ohjatun oppimisen menetelmin. Lisäksi syväoppimisen tutkimuksen kautta on muodostunut useita erilaisia rakenteita ratkaisemaan erilaisia koneoppimisen ongelmia. [33 s. 14, 163-164]

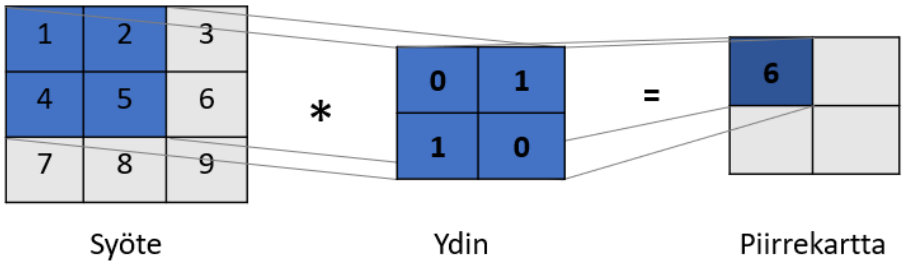


KUVA 12: ANN-struktuuri, jossa on m piilotettua kerrosta [mukailtu 33 s. 402; 36 s. 163-164]

Kuvan 12 havainnollistuksessa on asyklisenä suunnattuna verkkona kuvattu eteenpäin kytketty neuroverkko, jossa on m piilotettua kerrosta. Verkon tarkoitus on approksimoida jotain funktioita f^*, joilla syötteet x kuvataan tulosteiksi eli kategoriaan y, ja yleisimmin neuroverkko muodostuu näiden funktioiden $f^{(1..i)}$ ketjuista. Kuvan esimerkissä syötteitä on n-kappaletta ja niiden kategorioita k-kappaletta. Esimerkiksi koulutettaessa binäärinen luokittaja, olisi tulostekerroksessa kaksi solmua, joihin syötteet pyritään approksimoimaan. Oppimisprosessin aikana verkko oppii parametrit eli piirteiden painokertoimet, joilla se saavuttaa optimaalisen approksimaation virhefunktion suhteen. Oppiminen tapahtuu laskemalla datapisteiden piirrearvojen tuloksia verkon funktioilla eli neuronien aktivaatiofunktioilla ja muodostaen siten syötteitä seuraaville kerroksille. Koska koulutusdatasta ei näy syötteen ja tulosteen välisten kerrosten haluttuja syötteitä erikseen, kutsutaan näitä kerroksia piilotetuiksi. Oppimistulos riippuu syväoppimisenkin tapauksessa pääasiassa datan määrästä sekä datapisteiden keskinäisistä eroista eli separoitavuudesta. [36 s. 163-166] Alimmat eli kuvassa vasemmalla olevat kerrokset oppivat geneerisiä ominaisuuksia piirreavaruudesta, siinä missä ylemmät kerrokset oppivat näiden pohjalta tarkempia yksityiskohtia päätyen lopulta luokkakohtaisiin ominaispiirteisiin. Liitteessä 6 on havainnollistettu visuaalisesti konvoluutioverkon aktivaatiota.

Yksi suosituimmista neuroverkkojen erikoisrakenteista on kuvien tunnistamiseen ja luokitteluun kehitetty konvoluutioneuroverkko (*Convolution Neural Network*, CNN). CNN on ruudukon kaltaisen topologian omaavan datan prosessointiin erikoistunut neuroverkko. Tällaisia ovat muun muassa aikasarjat, jossa yksiulotteinen ruudukko kuvaa etenevää aikasarjaa, sekä kuvat, jotka ovat käytännössä pikseleistä koostuvia kaksi- tai kolmiulotteisia ruudukkoja. Kuten nimi viittaa, CNN käyttää tavanomaisen neuroverkon matriisikertolaskun sijaan lineaarista konvoluutio-operaatiota erillisissä konvoluutiokerroksissa. Konvoluutio on kahden funktion välinen operaatio, esimerkiksi funktioiden f ja g, joka tuottaa uuden funktion $s = f * g$. Konvoluutio määrittyy jatkuville funktioille integraalina. Konvoluutiolla muodostetulla funktiolla voidaan esimerkiksi laskea jonkin havainnon, kuten kuvan, osan painotettu keskiarvo, muodostaen häiriöitä suodattava, tiivistetty estimaatti käsiteltävistä havainnoista. Konvoluutiossa ensimmäisen argumentin nimi on syöte (”*input*”) ja jälkimmäisen ”*kernel”* eli ydin, ja näiden konvoluutio-operaatiosta seuraa vapaasti suomennettuna piirrekartta (”*feature map*”). Tällainen konvoluutio voidaan

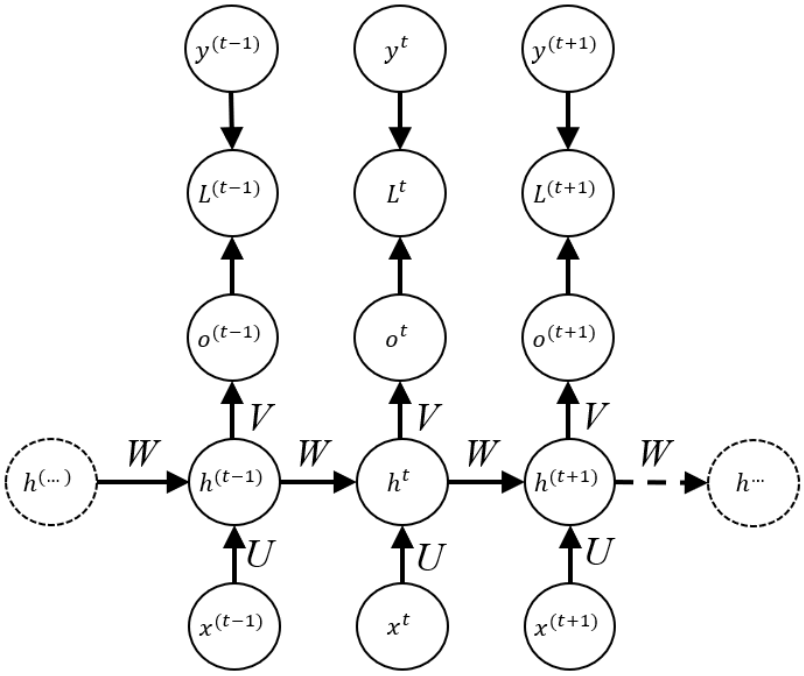
tehdä useamman kuin yhden ulottuvuuden suhteen kerrallaan, ja ytimen koko ja käyttö määrittävät tuloksena syntyvän piirrekartan muodon ja koon. Konvoluutio voi esimerkiksi kuvan tapauksessa suodattaa syötteestä kooltaan pienemmän piirrekartan tehostaen määräävimpien kaavamaisuuksien ja rakenteiden tunnistamista. Toinen kriittinen osa CNN rakenteita on niin kutsuttu yhdistämiskerros eli ”*pooling layer*”, joka suodattaa piirrekarttaa esimerkiksi valikoiden kattamansa alan parametreistä suurimman (*max pooling*) tai näiden keskiarvon (*average pooling*). [36 s. 327-330, 334-338]



KUVA 13: Konvoluutio-operaatio. Kuva mukailtu Goodfellow ym kirjan mallista [36 s. 320]

Konvoluutiokerroksessa CNN siis tiivistää edellisen kerroksen syötteen kuvan 13 kaltaisella konvoluutio-operaatiolla rajatummaksi piirrekartaksi, joka siirtyy seuraavaan kerrokseen. Sekä konvoluutio että yhdistäminen voivat aiheuttaa ominaisuuksillaan alisovittamista, mikäli rakenteella yritetään taltioida yksityiskohtaista spatiaalista informaatiota. Mikäli syötteestä halutaan löytää hyvin kaukana toisistaan olevien piirteiden yhteyksiä, saattavat konvoluutio ja yhdistäminen olla epäsopivia menetelmiä halutun piirreavaruuden mallintamiseen. [36 s. 336] Tämän opinnäytetyön liitteitä varten testattiin CNN-rakennetta myös audiodatan käsittelyyn, sillä Fourier-muunnettu audio muodostaa kaksiulotteisen piirrekartan, josta on opittavissa erilaisten ilmiöiden piirreavaruuksia erilaisten äänihavaintojen luokittelemiseksi. [36 s. 349]

Toinen oleellinen rakenne on takaisinkytketty neuroverkko (*Recurrent Neural Network*, RNN) joka muodostaa sarjadatan, kuten aikasarjojen, prosessointiin käytettävien neuroverkkojen joukon. RNN ylläpitää tietoa edellisen aikaleiman tilasta syöttäen sen nykyisen aikaleiman tilan prosessointiin. Toisin sanoen piirteiden tunnistus RNN-struktuurilla ottaa huomioon muutossuhteen edellisestä tilasta nykyiseen. Tällä tekniikalla pystytään esimerkiksi luomaan automaattisesti tekstiä tai tunnistamaan luonnollista puhetta. [33 s. 407; 36 s. 363] Tässä opinnäytetyössä RNN-rakennetta hyödynnetään tunnistamaan kaikumittaimen paluukaikuja, sillä RNN-rakenne soveltuu lähtökohtaisesti aikasarjaisen datan kuten äänen luokitteluun ja tunnistamiseen.



KUVA 14: Goodfellow ym. kirjasta mukailtu esimerkki RNN-rakenteesta [36 s. 369]

Kuvassa 14 on esimerkki takaisinkytketystä rakenteesta, jossa syöte x käsitellään siten, että piilotetut neuronit h jakavat oman tilansa kohdassa t eteenpäin eli tulevaisuuteen seuraavalle neuronille, joka käsittelee syötettä tilassa $t + 1$. U on syötteiden ja piilotettujen neuronien yhteyksien painovektori, W vastaavasti piilotettujen neuronien välisten yhteyksien painovektori ja V piilotettujen neuronien ja tulosteen yhteyksien painovektori. Tulostetta o verrataan todelliseen tulokseen y virheellä L. [36 s. 369] Näin käsiteltynä aikasarjaisesta datasta pystytään löytämään piirteitä, joissa muuttujan x edellinen tila määrittää sen nykyistä ja tulevaa tilaa. Näitä piirteitä ja niiden tilojen yhteyksiä hyödyntäen voidaan luokitella ja tulkita esimerkiksi luonnollista puhetta, jossa yhdistyy paitsi kirjainten äänteet myös äänen painot ja erityisesti lausutun kirjaimen liittäminen edelliseen ja seuraavaan. Tämän työn toisen kokeellisen osan hypoteesi on, että takaisinkytketty rakenne kykenee havaitsemaan kaikumittaimen paluukaiun vivahde-eroja nimenomaan ottamalla huomioon tilamuutosten nopeuden eri havaintojen välillä.

Syvät neuroverkot edellyttävät valtavasti dataa oppiakseen piirreavaruuden riittävällä tarkkuudella ja kyetäkseen tulkitsemaan syötteitä tulosteiksi. Koska alimmat kerrokset oppivat ongelma-alueen yleisiä piirteitä, voidaan tällaista valmiiksi koulutettua mallia hyödyntää niin kutsuttuna siirto-oppimisena samankaltaiseen ongelmaan uudelleenkouluttamalla vain ylimmät kerrokset ongelma-alueen datapisteillä. Siirto-oppimisen onnistumiselle on oleellista, että suuri osa tekijöistä, jotka selittävät variaatioita alkuperäisessä datassa, ovat yhteneviä uuden ongelma-alueen tekijöiden kanssa. [36 s. 526-528] Siirto-oppimisen hyödyntäminen yksinomaan sotilaallisissa sovelluksissa on haastavaa, sillä laajalla data-aineistolla koulutettuja malleja, joiden alkeellinen piirreavaruus olisi yhtenevä sotilaallisten käyttötarkoitusten kanssa, on verrattain vähän. Yhteneviä sovelluksia kuten alusten tunnistamista merellisestä ympäristöstä on oletettavasti kehitettävissä siirto-oppimisella esimerkiksi siten, että malli, joka tunnistaa erilaisia kohteita merellä oppisi tunnistamaan ja luokittamaan sota-aluksia uudelleenkouluttamalla mallin ylimmät eli luokittelukerrokset. Tämän työn liitteessä 6 on toteutettu alusluokkien tunnistamiseen tarkoitetun tekoälymallin kouluttaminen usein vertailtavin menetelmin, joista yksi on siirto-oppimisella toteutettu malli. Kokeen tulokset on raportoitu liitteessä ja analysoitu jäljempänä.

Syväoppimisessa voidaan hyödyntää osittain ohjattua oppimista, kun käytössä on vain osittain annotoitua dataa. Tällöin viitataan usein piirreoppimiseen, jossa samaan luokkaan kuuluvilla näytteillä on samankaltaiset piirreesitykset. [36 s. 236-237] Syväoppimista voidaan myös toteuttaa eräänlaisena vahvennettuna oppimisena tyypillisesti niin, että kahdelle neuroverkolle annetaan suuri datamäärä analysoitavaksi, jonka jälkeen toinen, niin kutsuttu generaattori pyrkii tuottamaan dataa vastaavan lopputuotteen, kun toinen, niin kutsuttu kriitikko tai diskriminaattori, arvioi onko lopputuote uskottava tausta-aineistoon verrattuna ja tuottaa palautteen generaattorille. Lopputuloksena, kouluttamisen onnistuessa, syntyy malli, joka kykenee luomaan synteettistä mutta diskriminaattorille aidon näköistä, uutta dataa. Tällaista verkkoa kutsutaan nimellä GAN (*generative adversarial network*). [57] Tekniikalla kyetään kouluttamaan tekoälyagenttia tehokkaasti ja jalostamaan esikoulutettua mallia suoriutumaan paremmin tietystä tehtävästä.

2.2.5 Tekoälyn kouluttaminen

Tekoälyn kouluttaminen koneoppimisella tapahtuu osa-alueesta riippumatta suorittamalla valitun metodin algoritmi käytettävissä olevalla datalla ja tuottamalla siten malli eli datan ominaisuuksiin sovitettujen painokertoimien joukko, joka oletusarvoisesti kuvaa datan ominaisuuksia sekä kykenee suorittamaan uuden, mallille vieraan aineiston pohjalta analyysia päätyen oikeisiin lopputuloksiin. Lopputulosten arvioinnissa kaikkiin koneoppimisen osa-alueisiin liittyy saavutettujen oppimistulosten eli mallin validointi. Koska koneoppiminen pohjautuu dataan ja sen menetelmillä koulutetun mallin odotetaan vastaavan johonkin tarpeeseen luokitella erilaisia havaintoja tai tehdä ennusteita havaintojen perusteella, on mallin tarkkuuden määrittäminen oleellinen osa kouluttamisprosessia ja ainut tapa arvioida valitun metodin suorituskykyä hypoteesiavaruuden vaihtoehtojen kesken. Aiemmin esitettyihin ohjatun ja ohjaamattoman oppimisen käytännön esimerkkeihin sitoen mallin laatua on arvioitu visualisoinneilla ja tarkkuusprosenteilla. Tarkkuuden määrittämiseksi mallia voidaan arvioida ajamalla sen läpi kyseinen data uudelleen ja vertaamalla saatuja luokituksia alkuperäisiin luokituksiin ja saamalla suoraviivaisesti prosenttiosuus oikein luokiteltujen näytteiden määrästä kuvaamaan mallin laatua. Ongelmaksi muodostuu se, että koko data-aineistoa käytettiin mallien kouluttamiseen eli malli on oppinut piirteiden painot jokaisen datapisteen osalta, jolloin validointi ei vastaa mallin toimivuutta koulutusaineistoon kuulumattomalla datalla. Tämän ongelman kiertämiseksi tulosten validointi toteutetaan yleensä eristämällä koulutusdatasta osa datapisteistä, tasaisesti jokaisesta luokasta, niin sanotuksi validointijoukoksi, joka käsittää esimerkiksi 20% datapisteistä. Malli voidaan kouluttaa käyttämällä jäljelle jäänyttä, 80% osuutta vastaavaa koulutusjoukkoa, jolloin mallin oppiminen perustuu kyseisiin datapisteisiin. Kun malli on koulutettu, voidaan sille syöttää validointijoukon datapisteet ja verrata niiden saamia luokituksia tunnettuihin todellisiin luokituksiin. Esimerkiksi aiemmin toteutettu multinomiaali logistinen regressio toimi koulutusdatalleen noin 97 prosenttisella tarkkuudella. Kuitenkin algoritmi on käyttänyt datapisteiden keskinäisten etäisyyksien mallintamiseen kaikkia datapisteitä, ja on siten voinut ylioppia (”*overfit*”) niiden piirteet ja soveltua huonosti kyseisen otoksen ulkopuolisiin havaintoihin, esimerkiksi siinä tapauksessa, että uudet datapisteet olisivat lähellä väärin luokiteltuja havaintoja, jolloin niiden sijainti päätösrajojen ja muiden datapisteiden suhteen aiheuttaisi väärän luokituksen.

Ylioppimisen vastakohta on alioppiminen, jossa malli on liian yksinkertainen kyseisen ongelman kuvaamiseen. Tämä on visualisoitu kuvassa 6, jossa lineaarinen malli on liian yksinkertainen kuvaamaan dataa. Vastaavasti liian monimutkainen malli ylioppii piirreavaruuden, mikä on nähtävissä kuvan 6 vasemman alareunan kuvassa, kuu-

dennen asteen polynomisen regression osalta. Ylioppiminen tai ylisovittaminen on datan määrän ja kompleksisuuden funktio: jos dataa on paljon, ylisovittamisen vaara on pieni, kuten kuvan 6 oikea alareuna osoittaa, sillä datan määrän moninkertaistaminen mahdollistaa kompleksisemmankin mallin sovittamisen dataan. Koneoppimisessa mallin riittävän kompleksisuuden arviointi noudattaa Ockhamin partateränä tunnettua ajatusta ”*pluralitas non est ponenda sine necessitate”*, eli vapaasti käännettynä ”moniarvoisuutta [\monimutkaisuutta] ei pitäisi asettaa ilman tarvetta”. Tämä tunnetaan toisessa muodossa siten, että kahdesta tieteellisestä teoriasta, jotka selittävät samaa asiaa, tulee valita yksinkertaisempi. [31 s. 696; 32 s. 211; 58]

Muodostamalla validointijoukko, jonka alkiot ovat eri datapisteitä kuin koulutusjoukon datapisteet, voidaan arvioida mallin suorituskykyä ja mahdollista yli- tai alioppimista yleisemmin. [33 s. 94-95, 108; 34 s. 73-74; 36 s. 107-108] On myös muita validointimetodeja, kuten LOOCV (*”leave-one-out cross validation”*), jossa arvioidaan virhe luomalla n mallia, joista jokaisen koulutusdata on kokoa $n - 1$, ja virhe arvioidaan vertaamalla pois jätettyä datapistettä koulutettuun malliin ja laskemalla näiden virheiden neliöity keskiarvo. Vastaavalla toimintalogiikalla voidaan suorittaa k-Fold ristiinvalidointi, jossa data jaetaan k osajoukkoon ja LOOCV periaatteen omaisesti malli koulutetaan $k - 1$ joukolla arvioiden virhettä suhteessa pois jätettyyn joukkoon. [33 s. 94-95] Joissain tilanteissa, kuten syvien neuroverkkojen kouluttamisessa, on hyödyllistä käyttää kolmea joukkoa, jolloin data on jaettu koulutusjoukkoon, validointijoukkoon ja testijoukkoon, esimerkiksi 80-10-10 suhteessa, jolloin koulutus- ja validointijoukon datalla koulutettu malli voidaan erikseen testata testijoukolla [36 s. 239].

Käytettäessä kouluttamiseen syviä neuroverkkoja laskentakapasiteetin tarve kasvaa huomattavasti, sillä mitä useampia ja laajempia toisiinsa kytkeytyneitä neuroneja sisältäviä kerroksia on, sitä enemmän muodostuu mahdollisia permutaatioita neuronien kautta kulkevien polkujen välille. Syväoppimisessa aineistosta rajattu koulutusjoukko jaetaan yleensä edelleen pienempiin osiin, jotka syötetään neuroverkoille toisensa jälkeen. Jokainen osa vastaa yhtä gradientin päivitysväliä. [59 s. 1-2] Jokaisella kerralla, kun datapisteet kulkevat neuroverkon läpi, mallin muodostamat painokertoimet muuttuvat jonkin verran, kunnes oppimista ei enää tapahdu. Kun kaikki osajoukot eli koko datapisteiden koulutusjoukko on käsitelty algoritmin toimesta, on suoritettu yksi epookki. Epookkien määrä, eli kuinka monta kertaa kaikki datapisteet ajetaan mallin läpi, määritetään ennen koulutuksen aloittamista. Tällöin validointijoukolla voidaan arvioida algoritmin oppimista kouluttamisen aikana jokaisen epookin kohdalla erikseen mahdollisen ylioppimisen tai alioppimisen havaitsemiseksi ja liiallisen kouluttamisen välttämiseksi. Suorittamalla useita epookkeja saadaan usein parannettua syväoppimisen mallin tarkkuutta, mutta jossain kohtaa oppimistulos ei enää merkitsevästi parane, jolloin kouluttaminen voidaan päättää. Valmiiksi koulutetun mallin toimivuus voidaan taas todentaa testijoukolla edellä mainitusti, kun kaikki epookit on suoritettu. [36 s. 107, 239-240]

Useiden koneoppimisalgoritmien kouluttamisessa hyödynnetään yllä sivuttua gradienttimenetelmää (”*gradient descent*”), jolla pyritään tässä tapauksessa löytämään virheavaruuden minimikohta eli niiden painojen kombinaatio, jotka minimoivat virhefunktion tuottaman virheen. Vastaavasti tarkastellessa palkkiofunktiota, jonka arvon maksimointi johtaisi parhaaseen tilanteeseen, voitaisiin käyttää ”*gradient ascent*” versiota maksimikohdan löytämiseksi. Gradienttimenetelmässä piirreavaruudesta lasketaan osittaisderivaattavektori, jonka yksittäisillä osittaisderivaatoilla voidaan arvioida yhden piirremuuttujan vaikutusta kyseisessä avaruudessa, muutoksen kuvaten nimenomaisesti suuntaa, johon kyseisen virheavaruuden arvot kasvavat jyrkimmin. Näin ollen tätä tietoa hyödynnetään ottamalla askel (”*gradient step*”), joka tunnetaan myös oppimisnopeutena (”*learning rate*”), vastakkaiseen suuntaan säädettäessä koulutettavan mallin painokertoimia optimaalisen mallin löytämiseksi, sillä optimaalinen

painokerroin löytyy gradientin minimikohdasta, johon funktio suppenee. On myös mahdollista, että menetelmä ei löydä virheavaruuden globaalia minimikohtaa vaan jonkun lokaalin minimin, esimerkiksi siinä tapauksessa, että piirreavaruus on erittäin suuri, ja satunnainen laskennan aloituskohta ja tietty oppimisnopeus päätyvät johonkin lokaaliin minimiin. Tällöin kouluttamisessa voidaan tyytyä löydettyyn lokaaliin minimiin, mikäli tulos on riittävä, tai kokeilla mallin uudelleenkouluttamista. Oppimisnopeus eli gradienttiaskel voi olla vakio tai määräytyä algoritmin laskennan edetessä esimerkiksi tietyn rappeutumisarvon (”*decay*”) mukaisesti. [31 s. 719; 32 s. 267-268; 34 s. 62-65]

Gradienttimenetelmä on yksinkertainen ja tehokas tapa löytää pienimmän virheen muodostavat painokertoimet, mutta syväoppimisessa tavanomaisen gradienttimenetelmän ongelmaksi muodostuu usein datapisteiden valtava määrä, sillä virheen arvioimisessa pitäisi laskea esimerkiksi satojen tuhansien datapisteiden mukaan päivitettyjen painokertoimien muodostaman virheen summa, jolloin yksittäinenkin iteraatio gradientin arvioimiseksi ja päivittämiseksi edellyttää huomattavaa määrää laskentakapasiteettia. Syväoppimisessa hyödynnetäänkin yleisemmin stokastista gradienttimenetelmää (”*stochastic gradient descent*”, SGD), jolloin gradientti approksimoidaan käyttämällä vain pientä osajoukkoa datasta (”*batch*”) kerrallaan. Häiriöttömän gradientin sijaan saadaan estimoitu, häiriöllinen gradientti, jonka avulla saadaan suunta painokertoimien päivittämiselle ja virhefunktion arvon pienentämiselle. Gradientin estimaattia päivitetään laskemalla seuraavan osajoukon virheavaruudesta päivitykset painokertoimille oppimisnopeuden mukaisella askelvälillä. Oppimisnopeuden ollessa vakio ei kuitenkaan ole takuuta virhefunktion suppenemisesta, sillä algoritmi saattaa jäädä silmukkaan minimin ympäristöön. Asettamalla algoritmille rappeutuva oppimisnopeus algoritmi suppenee varmasti vähintään lokaaliin minimiin. [31 s. 720]

Gradienttimenetelmään liittyen on mainittava historiakatsauksessa viitattu ”*back-propagation”* eli vastavirta-algoritmi, joka on tekoälyn kannalta merkittävä kehitysaskel vuodelta 1986. Eteenpäin kytkettyjen neuroverkkojen koulutuksessa tapahtuvaa datan kulkemista syötteestä x piilotettujen kerrosten kautta tulokseksi y kutsutaan ”*forward propagation*” termillä, joka kääntyy eteenpäin eteneväksi. Kouluttamisen aikana eteenpäin etenevä algoritmi jatkaa syötteestä verkon kerrosten läpi kohti tulostetta, kunnes tuloksena on skalaarinen hinta eli virhe prediktion approksimaation ja todellisen annotaation välillä. Vastavirta-algoritmi taas käyttää tätä laskettua virhettä kulkeakseen verkossa taakse päin gradientin laskemiseksi ja gradienttimenetelmän mahdollistamiseksi yksinkertaisesti ja tehokkaasti. Algoritmi laskee gradientin käyttämällä rekursiivisesti differentiaalilaskennan ketjusääntöä laskeakseen verkon jokaiselle operaatiolle kyseisen osittaisderivaattamatriisin ja gradientin tulon. Taaksepäin ketjusäännöllä toteutettu iteraatio tehostaa laskentaa minimoimalla välivaiheet, joita aiheutuisi laskettaessa gradientti alusta loppuun eteenpäinkytketysti. Näin laskettua gradienttia voidaan hyödyntää esimerkiksi yllä mainitussa stokastisessa gradienttimenetelmässä. [36 s. 197-199]

2.3 Tekoälyn haasteet

Kehitysaskelista ja lupaavista tulevaisuudennäkymistä huolimatta tekoäly kohtaa useita haasteita. Osa etenkin asejärjestelmiin liittyvistä haasteista on eettisiä ja moraalisia, esimerkiksi vastuukysymyksiä siitä, kenen vastuulla tekoälyn tukema tai tekemä päätös on, jos se osoittautuu virheelliseksi. Tästä ongelmasta puhutaan usein termillä ”*accountability gap*”, joka kuvaa tekoälyn muodostamaa laillista ja moraalista vastuunkannon tyhjiötä, sillä tekoälyn ohjaama järjestelmä ei ole laillinen entiteetti.

Käytännöllisemmät haasteet liittyvät muun muassa laskentakapasiteettiin, jota modernit tekoälymenetelmät kuten syväoppiminen edellyttävät. Toisaalta tietokoneiden laskentateho jatkaa kasvuaan ja niin käytössä kuin näköpiirissä on uusia innovaatioita kuten pilvilaskenta ja kvanttitietokoneet [60; 61]. Ilmiselvä haaste koneoppimisella luoduissa tekoälymalleissa on käytetty data ja sen laatu. Huonolaatuisen tai huonon kattavuuden omaavan datan avulla koulutettu äly voi sisältää esimerkiksi merkittäviä vinoumia. Data saattaa olla esimerkiksi spatiaalisesti ja ajallisesti vinoutunutta, sillä näytteet, jotka on otettu ajallisesti ja paikallisesti lähekkäin muistuttavat toisiaan suuremmalla todennäköisyydellä kuin spatiaalisesti erilaiset, saman kohteen tai ilmiön näytteet. Tästä syystä yksi tärkeä tutkimusalue on data-augmentaatio, jolla kyetään luomaan olemassa olevista näytteistä uusia näytteitä ja siten parantamaan oppimistuloksien tarkkuutta [36 s. 445]. Augmentaatio voi olla esimerkiksi kuvien osalta yksinkertaista kuvankäsittelyä rajauksen ja kiertojen suhteen tai monimutkaisempaa sää- ja valaisuolosuhteiden muokkaamista. [62] Usein mallin hyvyys perustuu myös pitkälti tekijänsä asiantuntemukseen aiheesta, jolloin pelkkä tekoälyosaaminen ei takaa onnistunutta mallintamista. Tässä yhteydessä korostuu myös tekoälyn kapeus ja inkrementaalisuus; koska ei ole olemassa yleistä tekoälyä, on jokainen kapea tekoäly räätälöity kyseiseen kapeaan ongelma-alueeseen, jolloin sen kehityksessä ja käyttöönotossa piilevät aina samat riskit, vaikka käytetyt metodit ovatkin varsin universaaleja.

Tekoälyyn liittyy myös selitettävyyden ongelma. Tekoälyn menetelmiä ja niillä luotuja malleja kuvataan termeillä ”*black-box*” ja ”*white-box*”, joista niin sanotuissa mustissa laatikoissa tapahtuva päättely on erittäin vaikea ymmärtää. Mustiksi laatikoiksi mielletään yleensä lukuisista muuttujista ja näille opituista painokertoimista koostuva, monimutkainen matemaattinen malli, jonka visualisointi on vaikeaa tai mahdotonta ja hahmottaminen on myös vaikeaa. Valkoisiksi laatikoiksi taas mielletään tekoälymallit, jotka ovat itsensä selittäviä, kuten päätöspuuoppiminen, jonka esimerkki on visualisoitu kuvaan 8. Toisaalta, vaikka neuroverkoilla tuotetut mallit mielletään vaikeasti ymmärrettäviksi, voi esimerkiksi konvoluutioverkon mallilla tulkittua kuvaa tulkita myös ihmisen toimesta yhtäaikaisesti, eikä ihmisen tarvitse luottaa pelkästään tekoälyn tulosteeseen. [63] Sotilaallisissa sovelluksissa selitettävyyden tarve on esimerkiksi voimankäytön osalta kriittinen. Käytettäessä tekoälyä avustamaan esimerkiksi tulenkäytön mahdollistavan tilannekuvan luomisessa on päätöksentekijälle oleellista, että sotilaallisen voimankäytön perusta on ymmärrettävä. Mustan laatikon tapauksessa näin ei välttämättä ole, jolloin malliin on luotettava sokeammin kuin valkoiseen laatikkoon. Tällainen luotto edellyttää mallin todennettua toimivuutta todellisessa ympäristössä. Tästä juontuu toinen ongelma eli tekoälyjärjestelmien kattavan testaamisen haastavuus. Jo kohtuullisella määrällä muuttujia pystytään muodostamaan esimerkiksi päätöksentekoalgoritmille erittäin monta prioriteettijärjestystä näiden muuttujien painottamiseksi. Yhdysvaltojen laivaston AEGIS-järjestelmä on varsin moderni esimerkki erilaisia automaatioasteita sisältävästä, kompleksisesta järjestelmästä, jonka toiminta itse operaatioalueella pyritään varmistamaan laatimalla tehtävään sopiva doktriini kyseiselle järjestelmälle ja testaamalla järjestelmän toiminta kytkemättä sen asejärjestelmäyhteyksiä päälle aluksen saapuessa operaatioalueelle. Kun toiminta on

varmistettu ja mahdollisesti doktriinia muutettu, voidaan järjestelmää käyttää halutulla automaatioasteella. [43 s. 163-166] On kuitenkin vaikeaa laatia säännöstöä tai kouluttaa tekoälyä vastaamaan sodankäynnin näyttämön normaalista todellisuudesta todennäköisesti poikkeavaa tapahtuma-avaruutta. Lisäksi entistä monimutkaisempien järjestelmien todellisesti kattava testaaminen saattaa osoittautua ylipäätään mahdottomaksi tehtäväksi vaikkei se operoisikaan sotanäyttämöllä. [43 s. 149] Esimerkiksi maaliskuussa 2021 julkistettu Hyundain uusi Legend-auto kykenee operoimaan autonomisten autojen tasolla 3, jonka testaamiseksi on suoritettu 10 miljoonaa simuloitua liikennetilannetta ja ajettu 1.3 miljoonaa testikilometriä. [64] Säännönmukaisten liikennetilanteiden simuloiminen sisältää merkittävän määrän variaatioita, mutta muuttujien määrä on nähtävästi hallittavissa. Mikäli samanlaista maturiteettitestiä pyritään tekemään taistelevalle järjestelmälle, nousee vaikuttavien tekijöiden määrä moninkertaiseksi ja lisäksi suuri osa vaikuttavista tekijöistä vastustajan osalta ei lähtökohtaisesti ole tiedossa tai varmistettavissa. Tämä muodostaa ratkaisemattomalta vaikuttavan tilanteen, jolloin autonomisen järjestelmän käyttöönoton vaihtoehdoiksi jäävät ihmisen sisällyttäminen järjestelmän toimintaan tai käyttöönottoon liittyvän epävarmuuden hyväksymisen.

Yksi vakavimpia tunnistettuja ongelmia etenkin sotilaallisten sovellusten kannalta on niin sanottujen ”*adversarial examples*” eli vapaasti suomennettuna vastakkaisten näytteiden väärinluokittelu, joka koskee koneoppimismalleja. Vastakkaiset näytteet ovat pieniä ja usein huomaamattomia häiriöitä, joita lisäämällä mallin tulkitsemaan dataaineistoon saadaan malli antamaan täydellisen vääriä luokituksia ja niille vieläpä erittäin korkea luotettavuusarvo. [65] Saadut tulokset vastakkaisten näytteiden vaikutuksista tekoälyn luokitustarkkuuteen antavat viitteitä ennakoimattomista tuloksista, joita tekoälysovellusten käyttöönotto tuo mukanaan. Sotilaallisissa ympäristöissä tehtävä vastakkaisten näytteiden hyödyntäminen voidaan jakaa kahteen vaihtoehtoon, kohdistettuun (*targeted adversarial example*) ja kohdistamattomaan *(untargeted adversarial example*). Kohdistettu on suunniteltu huijaamaan kohteen syväoppinut malli tulkitsemaan objekti toiseksi, hyökkääjän valitsemaksi kohteeksi, kun taas kohdistamaton pyrkii yksinkertaisemmin saamaan vastustajan mallin tuottamaan vääriä luokituksia. Nämä voidaan edelleen luokitella kahteen luokkaan sen mukaan, ovatko kohteena olevan mallin yksityiskohtaiset parametrit tiedossa vai eivät. [66] Tekniikan tehokas käyttö edellyttää käytännössä sitä, että muokattuja kuvia pystytään syöttämään suoraan kohteen mallille. Sensoridataa tulkitsevan mallin huijaaminen onnistuu tutkimusten perusteella myös fyysisessä maailmassa esimerkiksi tulostamalla vastakkaisella näytteellä muokattuja kuvia. [67] Tästä huolimatta ei ole yksiselitteistä, että tekniikan hyödyntäminen on käytännössä mahdollista esimerkiksi taistelualusten toiminnassa, mutta haavoittuvuuden uhkaavuutta ja toisaalta mahdollisuuksia ei voida jättää huomiotta.

Tekoäly kohtaa ja tuottaa myös organisatorisia haasteita. Näitä ongelmia on muun muassa organisaatiokulttuurissa, henkilöstön osaamisessa sekä liiallisissa odotuksissa tai peloissa. Tekoälyn onnistunut käyttöönotto edellyttää muutokselle positiivista organisaatiokulttuuria sekä henkilöstön uudelleenkouluttamista ja taitojen kartuttamista tehtävänkuvien muuttuessa. Lisäksi investointien ja muutosten tulisi kyetä lunastamaan lupauksensa, joka on helpoin toteuttaa pitämällä tulevaisuuden visiot mahdollisimman realistisina. [68] Puolustusvoimien organisaation ja pääprosessien näkökulmasta haasteita on kartoitettu everstiluutnantti Petteri Hemmingin opinnäytetyössä. Työssä on tunnistettu lukuisia pääprosesseihin liittyviä haasteita niin liian hitaassa kuin nopeassa tekoälyn implementaatiossa. Autonomisten järjestelmien osalta hitaan progression riskiksi on nostettu vanhentuneiden järjestelmien käyttöönotto, kun taas nopean progression riskinä nähdään huono teknologinen valmius tai maturiteetti. [69 s. 25-26]

2.4 Automaatio ja robotiikka

Automaatio tarkoittaa sanakirjamääritelmällisesti tekniikkaa, jolla laite, järjestelmä tai prosessi operoi automaattisesti eli ilman ihmisen jatkuvaa kontrollia. Automaation toteutus voi olla laite, prosessi tai järjestelmä, joka koostuu mekaanisista tai elektronisista laitteista korvaten ihmisen tekemää työtä. [70] IEEE Guide for Terms and Concepts in Intelligent Process Automation määrittää automaation itsenäiseksi koneohjatuksi koreografiaksi, jolla operoidaan yhtä tai useampaa digitaalista järjestelmää. [71 s. 11]

Robotiikka tarkoittaa sanakirjamääritelmällisesti teknologiaa, jossa suunnitellaan, rakennetaan ja operoidaan robotteja automaation toteuttamiseksi. [72] Robotti on laite, joka suorittaa automaattisesti monimutkaisia, usein toistuvia tehtäviä kuten teollisuuden kokoamislinjaston työtehtäviä. [73]

Voidaan todeta, että robotiikka kehittää mekaanisia tai elektronisia laitteita ja järjestelmiä, joita operoidaan automaation avulla korvaamaan ihmisen tekemää, usein fyysistä, työtä. Nykyisin automaatio ja robotiikka korvaa muun muassa tuotannollisissa prosesseissa usein esiintyvää ihmisten tekemää toistotyötä. Automaation ja robotiikan kehittyessä kyetään yhä suurempia kokonaisuuksia suorittamaan automaattisesti. Teollisessa tuotannossa robotit ja automaatio vähentävät varianssia ja kustannuksia sekä lisäävät tehokkuutta verrattuna ihmisten suorittamiin prosesseihin. Työtehtävien korvaamista roboteilla on ajanut sekä teollisen tuotannon toimialaa kohdannut työvoimapula, kustannussäästöt sekä työnkuvan vaarallisuus ja riskit. Vastaavasti työnkuvan ja toimintaympäristön vaarallisuus ovat ajaneet robotiikan kehitystä avaruustutkimuksessa ja sotilaspuolella muun muassa raivaustoiminnassa. [74 s. 15-20]

Manuaalisen, perinteiseksi mielletyn tehdastyön sekä muiden fyysisen maailman työtehtävien korvaamisen lisäksi ohjelmistotekniikan automaatio korvaa tai täydentää yhä suurempaa osaa muistakin, myös kognitiivisista työtehtävistä. [75 s. 77, 79] Tuotantoteollisuuden ulkopuolisena esimerkkinä lääketieteessä hyödynnetään kasvavissa määrin robotiikkaa. [76 s. 1] Kognitiivinen automaatio, joka nähdään osana tekoälyä, on määritelmällisesti käytössä olevien koneoppimisalgoritmien hyödyntämistä toimintaympäristöspesifin datan analysointiin ja päättelyyn koneoppimisen automatisoimiseksi entistä pidemmälle siten, että automaatio vaikuttaa kognitiiviselta. [71 s. 13] RPA (robotic process automation) tarkoittaa esikonfiguroitua ohjelmistoa, joka käyttää ihmisen valvonnassa säännöstöä ja ennalta määritettyä aktiviteettien koreografiaa suorittaakseen autonomisesti yhdistelmän prosesseja, aktiviteetteja, transaktioita ja tehtäviä yhdessä tai useammassa ohjelmistojärjestelmässä tuottaakseen palvelun tai lopputuloksen. [71 s. 11]

Sotilasympäristössä robotiikan merkkipaaluiksi on nostettu muun muassa vuonna 2002 suoritettu sotilasoperaatio, jossa Yhdysvaltojen Predator UAV tuhosi ajoneuvon Hellfire-ohjuksella. [74 s. 18; 77 s. 8] Toisaalta erilaisia robotteja on ollut asevoimien käytössä jo paljon kauemmin, mutta kyseisessä tapauksessa yhdistyy lennokin miehittämättömyys ja tappavan voiman käyttö.

Yllä mainituista määritelmistä ja näiden välisistä yhteyksistä havaitaan, että automaation, robotiikan ja tekoälyn rajapinta on häilyvä. Maanpuolustuksen tieteellisen neuvottelukunnan 18.11.2014 pitämän tutkimusseminaarin määritelmän mukaisesti automaatti on ei-autonominen järjestelmä, jolla ei ole kognitiivisia ominaisuuksia kuten

oppimista, päättelykykyä ja adaptiivisuutta. Samassa tutkimusseminaarissa nostettiin esiin miehittämättömien järjestelmien ominaisuudet, sillä miehittämättömyys ei ole synonyymi autonomian kanssa. [78] Automaattinen järjestelmä voi kuitenkin hyödyntää kapeaa tekoälyä toiminnassaan, kuten kognitiivinen automaatio osoittaa. Tällöin käytetyn tekoälyn kognitiiviset ominaisuudet voivat mahdollistaa autonomisia piirteitä järjestelmässä hämärtäen tekoälyn, autonomian ja automaation rajapintoja entisestään.

Siinä missä tekoälyllä on terminä tapana siirtyä tarkoittamaan toistaiseksi saavuttamattomia kehitysasteita, käsitetään saavutetut kehitysasteet sekä yleistyvät ja arkipäiväistyvät tekoälyn menetelmät ja sovellukset usein automaatioksi ja ohjelmistorobotiikaksi. Monet tälläkin hetkellä muun muassa laivastojoukoissa käytössä olevat järjestelmät hyödyntävät tekoälyksi luokiteltavaa automatiikkaa toteuttamaan toimenpiteitä älykkäältä vaikuttavasti.

Tämän opinnäytetyön tarkastelussa automaatio ja robotiikka käsitetään toiminnallisena osana tekoälyä, jolla tuotetaan tekoälyn vaikutus halutussa ympäristössä, kuten esimerkiksi autonomisen aluksen merenkulussa. Tästä syystä opinnäytetyön otsikossa on mainittu automaatio, sillä linkitettynä työssä käytettyyn tekoälyn määritelmään molempien tavoite on toteuttaa ihmisen tekemiä toimenpiteitä vähentäen tai poistaen kokonaan ihmisen tarpeen puuttua kyseisen prosessin suorittamiseen. Automaatiota ja robotiikkaa ei käsitellä syvällisemmin tutkimuksen keskittyessä tekoälyn menetelmien hyödyntämiseen laivaston toiminnassa, mutta robotiikan osuus älykkäiden agenttien aktuaattoreina etenkin fyysisissä järjestelmissä on ymmärrettävä jatkumoksi tässä tutkimuksessa analysoiduille kokonaisuuksille.

2.5 Miehittämättömyys ja autonomia

Autonomia ja miehittämättömyys ovat yksi yhdeksästä fokusoidusta osa-alueesta Office of Naval Research laitoksen merellisen tieteen ja teknologian strategiaa Yhdysvalloissa, jolla on visio luoda integroitu miehitettyjen ja miehittämättömien alusten laivastojoukko. Defense Science Board, jatkossa DSB, määrittää autonomian kyvyksi tai joukoksi kykyjä, jotka mahdollistavat järjestelmän automaattisuuden tai, ohjelmoinnin rajoitteissa, itsemääräämisen. DSB määrittää toisessa tutkimuksessaan autonomian olevan tulos siitä, että tietty päätöksentekokyky delegoidaan auktorisoidulle taholle, jolla on siten valta toimia määritetyissä rajoissa. Yhdysvaltojen puolustusministeriön autonomiaosasto taas määrittelee autonomian laskennalliseksi kyvyksi, joka mahdollistaa älykkäät toimenpiteet, joilla voidaan toteuttaa monimutkaisia tehtäviä haastavissa ympäristöissä huomattavasti vähentäen tarvetta ihmisen kontrollille ja interventioille korostaen ihmisen ja koneen interaktiivisuutta. [20]

Miehittämättömien, autonomisten järjestelmien kuvataan tarjoavan ennenäkemätön mahdollisuus jälleen kerran muuttaa merisodankäynnin luonnetta. Siinä missä sensorijärjestelmät ovat jo nousseet ilmaan miehittämättömien ilma-alusten hyötykuormina, lähitulevaisuudessa taistelualusten hyötykuormat voidaan sijoittaa niin ilma-, pinta- kuin vedenalaisiin miehittämättömiin lavetteihin. [79 s. 290] Riittävän luotettavan autonomian mahdollistamiseksi tekoälyn menetelmien tulee kuitenkin vastata edellisessä alaluvussa mainittujen kategorioiden haasteisiin, sillä autonomisen lavetin tulee kyetä, riippuen voimankäytön säännöistä, toteuttamaan koko OODA-silmukka itsenäisesti. Kyseisiä menetelmiä ei kuitenkaan tarvitse sitoa pelkästään autonomisten järjestelmien kehitystyöhön, vaan kehitystyö voi olla inkrementaalista nykyisestä operatiivisesta ympäristöstä, siirtyen autonomisten järjestelmien hyödynnettäväksi, kun maturiteetti katsotaan riittäväksi ihmisoperaattorien työpanoksen tukemisessa. Alla on käsitelty autonomiaan liittyvää käsitteistöä, autonomisista merellisistä järjestelmistä tehtyjä tutkimusraportteja sekä kyseisten järjestelmien hyödyntämiseen liittyviä käytännön seikkoja. Tässä työssä autonomiset järjestelmät nähdään pitkän aikavälin tavoitteena uudistaa merisodankäyntiä, lähitulevaisuuden näkymän painottuessa mahdollistavien tekoälymenetelmien kehittämiseen.

TAULUKKO 1: Lloyd's Registerin mukaiset autonomialuokitukset [80 s. 1-2]

AL 0)	Manuaalinen: ei autonomisia toimintoja. Kaikki toimenpiteet ja päätöksenteko tehdään manuaalisesti, eli ihminen kontrolloi kaikkia toimenpiteitä. Järjestelmillä voi kuitenkin olla autonomisia ominaisuuksia, mutta ihminen on kiinteästi päätöksentekosilmukassa.
AL 1)	Aluksella oleva päätöksenteon tuki: Kaikki toimenpiteet tehdään ihmisoperaattorin toimesta, mutta päätöksentekoa voidaan tukea järjestelmän tuottamilla vaihtoehdoilla tai muulla toimenpiteisiin vaikuttavilla tiedoilla. Data tuotetaan aluksella olevin järjestelmin.
AL 2)	Aluksella tai muualla sijaitseva päätöksenteon tuki: Kaikki toimenpiteet tehdään ihmisoperaattorin toimesta, mutta päätöksentekoa voidaan tukea järjestelmän tuottamilla vaihtoehdoilla tai muulla toimenpiteisiin vaikuttavilla tiedoilla. Data tuotetaan aluksella tai muualla sijaitsevin järjestelmin.
AL 3)	"Aktiivinen" ihminen silmukassa: Päätökset ja toimenpiteet toteutetaan ihmisen valvonnassa. Data tuotetaan joko aluksella tai muualla sijaitsevin järjestelmin.
AL 4)	Ihminen silmukassa: Päätökset ja toimenpiteet toteutetaan autonomisesti ihmisen valvonnassa. Suuren vaikuttavuuden päätökset toteutetaan siten, että ihmisoperaattorille annetaan mahdollisuus puuttua niiden toteuttamiseen ja ohittaa järjestelmä.
AL 5)	Täysi autonomia: Harvoin valvottu toiminta, jossa päätökset tehdään ja toteutetaan täysin järjestelmän toimenpitein
AL 6)	Täysi autonomia: Valvomattomasti toimiva järjestelmä, joka toteuttaa päätöksensä täysin itsenäisesti tehtävän suorituksen aikana.

Korkeamman autonomian järjestelmä voi hyödyntää matalampia tasoja osana toimintaansa ja monimutkainen järjestelmä saattaa koostua useista eritasoisista järjestelmistä. [80] Ihmisen rooli voidaan nähdä kolmessa eri asemassa autonomia-asteesta ja kontrollistrategiasta riippuen. Ensimmäistä voidaan luonnehtia niin sanotusti etäohjauskontrollin autonomisena järjestelmänä. Etäohjauskontrolli mahdollistaa sen, että miehittämätön alus suorittaa ohjauskäskyjä, muttei omaa päätöksenteko- tai suunnittelukykyä. Ihminen on osa autonomisen järjestelmän silmukkaa ja suorittaa päätöksenteon sekä vaativammat tehtävät etäohjauksessa. Tällaisesta osallisuudesta on kyse yllä listatuissa luokituksissa AL 0-2. Seuraava vaihtoehto olisi puoliautonominen tai valvottu autonominen järjestelmä, jolla tarkoitetaan korkeamman tason kontrollistrategiaa, jossa järjestelmällä on yksinkertainen päätöksentekokyky, jolla se kykenee suorittamaan yksinkertaisia tehtäviä ihmisen puuttuessa vaikeampien tehtävien suorittamiseen. Ihminen valvoo aktiivisesti autonomista järjestelmää ja omaa kyvyn puuttua sen toimintaan tarvittaessa. Taulukon 3 luokittelussa vastaavat tasot ovat AL 3-4. Viimeinen ja edistynein vaihtoehto olisi tasoja AL 5-6 vastaava täysautonomia, jossa ihminen ei kohdista järjestelmään edes aktiivista valvontaa eikä lähtökohtaisesti puutu järjestelmän päätöksentekoon. Täysautonomisessa kontrollissa järjestelmä on kykenevä suorittamaan kaikkia tehtävätyyppejä useimmissa olosuhteissa ilman ihmisoperaattorin puuttumista. [23 s. 5-6] Toisaalla vastaavat autonomia tasot tunnetaan termeillä ”*in the loop*”, ”*on the loop*” ja ”*out of the loop*”, joissa nimensä mukaisesti ensimmäisessä ihminen on kiinteästi autonomisen järjestelmän päätöksentekosilmukassa, toisessa valvoo aktiivisesti tätä silmukkaa ja kolmannessa järjestelmä toimii tarvittaessa täysin ilman ihmistä. [43 s. 44-47]

Järjestelmä voi siis omata erilaisia kombinaatioita miehitysasteiden ja autonomia-asteiden välillä eikä autonomisuus ole synonyymi miehittämättömyydelle tai toisin päin. Järjestelmän luotettavuuden ja robustiuden voidaan nähdä korreloivan negatiivisesti kontrolliasteeseen; mitä korkeampi autonomia-aste, sitä heikompi kontrolli ja vastaavasti heikompi luotettavuus [23 s. 6]. Ihmisoperaattorin kontrolli mahdollistaa tässä näkemyksessä luotettavamman järjestelmän. Näin ollen ihmisoperaattorien tulisi pysyä kiinteänä osana autonomisten alusten operointia teoriassa siihen asti, että hypoteettinen AGI on kehitetty mahdollistamaan ihmisen tasoinen autonomia. Autonomisista järjestelmistä voidaan kuitenkin ulosmitata merkittäviä hyötyjä puuttumatta miehitysasteeseen tai siirtämällä kontrollia muualle kuin itse tehtävää suorittavalle lavetille, aivan kuin muussakin automaatiossa on tapahtunut: ihmisten rooli on muuttunut suorittajista valvojiksi. Polku miehitetyistä, konventionaalisista laveteista täysautonomisiin, miehittämättömiin alusyksiköihin edellyttää lähes välttämättä välivaiheita, joissa järjestelmän toimintaa todennetaan ja testataan ainakin osittain miehitetyssä konfiguraatiossa. Tässä työssä tarkastellut OODA-kategoriat ja niiden mahdollisuudet sekä vaatimukset tekoälyn hyödyntämiselle kartoittavat osaltaan Suomen laivastojoukkojen valmiuksia ja kehitystarpeita autonomisten järjestelmien mahdollistamiseksi.

RAND tutkimuslaitoksen kattavassa analyysissä ”*Advancing Autonomous Systems”* autonomisten merellisten järjestelmien rakenne kuvataan siten, että kriittisin tekijä autonomian saavuttamiseksi on, varsin itsestään selvästi, järjestelmän älynä toimivat algoritmit. Järjestelmän hyötykuormat taas mahdollistavat lavetin vuorovaikutuksen ympäröivän maailman kanssa. Näin ollen hyötykuormat nähdään raportissa autonomisen älyn mahdollistajina ja ihmisten vahvistajina silloin, kun ihmiset pysyvät silmukassa. Lavetti taas toimii nimensä mukaisesti alustana älykkäälle järjestelmälle ja sen hyötykuormille. [20 s. 8] Raportissa eritellään algoritmit tehtävätaksonomian mukaan yksittäisen miehittämättömän järjestelmän osalta siten, että kokonaisuus koostuu viidestä tehtäväkategoriasta, joihin kuuluu vaihteleva määrä alatehtäviä. Tehtäväkategoriat ovat merenkulku, tutkiminen, vaikutuksen tuottaminen, vastatoimet sekä resurssienhallinta. Alakategoriat muodostuvat taulukon 2 mukaisesti.

TAULUKKO 2: Autonomisten pinta-alusten algoritmien alakategoriat [20 s. 10-11]

Merenkulku	Paikanmääritys
	Reittisuunnittelu
	Törmäyksen välttäminen
	Referenssidataperustainen reitin seuranta
	Paikka-aseman ylläpitäminen
Tutkiminen	Kohteen tunnistus
	Havaitseminen ja jäljittäminen
	Seuranta
	Samanaikainen paikanmääritys ja kartoitus
	Oseanografia
Vaikutuksen tuottaminen	Kineettinen isku
	Ei-kineettinen isku
	Manipulaatio
Vastatoimet	Väistäminen
	Viansieto
	Kyberresilienssi
	Harhauttaminen
Resurssien hallinta	Virranhallinta

Autonominen merellinen järjestelmä koostuu huomattavasta määrästä toisistaan merkittävästi poikkeavia tehtäviä, joiden suorittamiseksi vaaditaan erillisiä alajärjestelmiä algoritmeineen ja hyötykuormineen. Myös miehitetyn aluksen tehtävät korreloivat tähän taksonomiaan ja alatehtäviä toteuttavat kyseiseen tehtävään erikoistuneet henkilöt. Kuten oletettavaa on, ovat vaikutuksen tuottamisen tehtävät huonoiten edustettuna tieteellisissä julkaisuissa ja silloinkin painottuen algoritmien osalta korkeamman tason taistelunjohdollisiin tehtäviin kuten maalien allokointiin eri yksiköille tai asejärjestelmille [20 s. 58]. Koska sotilaallinen vaikutuksen tuottaminen kiinnostaa pääasiassa sotilaita, pitäisi sotilailla olla myös paras näkemys halutusta vaikutuksesta ja sen tuottamisen prosessista. Näin voidaan argumentoida, että vaikutuksen tuottamisen algoritmien kehitystä pitäisi edistää nimenomaan sotilaiden toimesta.

Miehittämättömien alusten valvontaan ja kontrolliin on visioitu etäohjausasemia. Yksi visio kauppa-alusten etäohjausasemasta (*”remote operation center”*, ROC) muistuttaa miehitetyn aluksen komentoketjua koostuen aluksen päälliköstä, Fleet Mission Management (FMM) operaattoreista ja Remote Control Station (RCS) operaattoreista. Henkilöstön tehtävät vastaavat tavanomaisen aluksen vastuunjakoa, jossa etäohjausaseman päällikkö hyväksyy suunnitelmat ja toteuttaa hallinnolliset tehtävät sekä vastaa operaattoreiden toiminnasta sekä huollon henkilöstöstä asemalla. Yhteys valvottaviin ja kontrolloitaviin aluksiin muodostettaisiin satelliitilla. [81 s. 14-16] Vastaavaan konseptiin on päätynyt jo aiemmin Euroopan unionin MUNIN-projekti, jossa lähtökohta autonomisille aluksille on ranta-asema SCC (*”shore control centre”*). Ranta-aseman valvonta ja kyky puuttua autonomisen aluksen kulkuun nähtiin esivaatimuksena kustannustehokkaan konseptin luomiselle, sillä ilman ranta-asemaa autonominen alus vaatisi projektin näkemyksessä äärimmäisen monimutkaista ohjelmistoa kaikkien mahdollisten tapahtumatilanteiden hallitsemiseksi. Ranta-aseman valvonnassa alus kykenee suorittamaan suurimman osan tilanteista autonomisesti mutta voi tukeutua ranta-aseman ohjaukseen jäljelle jäävissä, haastavimmissa tilanteissa. MUNIN-projektin kehittämä tekninen konsepti autonomisen kauppa-aluksen operoinnille koostuu neljästä järjestelmästä: kehittyneestä sensorimoduulista (”*advanced sensor module*”, ASM), autonomisesta navigaatiojärjestelmästä (ANS), autonomisesta koneiston valvonnasta ja hallinnasta (”*autonomous engine and monitoring control system*”,

AEMC) sekä edellä mainitusta ranta-asemasta. [82] ANS, ASM ja AEMC voidaan nähdä korreloivan suoraan taulukon 3 algoritmikategorioihin, pois lukien vaikutuksen tuottaminen ja vastatoimet. Huomionarvoista on, että MUNIN visioi valtamerikelpoista autonomista kauppa-alusta, eikä siten korreloi täysin Suomen laivaston toimintaympäristön vaatimusten kanssa, mutta pääperiaatteet voidaan tulkita yhteneviksi eriävästä käyttötarkoituksesta huolimatta.

Etäohjausasema on oletettavasti osa myös sotilaallista autonomista järjestelmäkokonaisuutta niin teknisen toteutettavuuden kuin oikeudellisten vastuukysymystenkin johdosta. Sota-alusten kontrolloitavat toiminnot ovat moninaisempia sisältäen navigoinnin ja koneistovalvonnan lisäksi sotilaallisten sensorien ja vaikuttimien käytön. Esimerkiksi Yhdysvaltojen DODD 3000.09 edellyttää eksplisiittisesti, että autonomiset järjestelmät suunnitellaan siten, että komentajilla ja ihmisoperaattoreilla on kaikissa tilanteissa mahdollisuus harjoittaa ”*sopivaa inhimillistä arviointia voimankäytöstä*” kaikissa, myös autonomisissa asejärjestelmissä [83 s. 2].

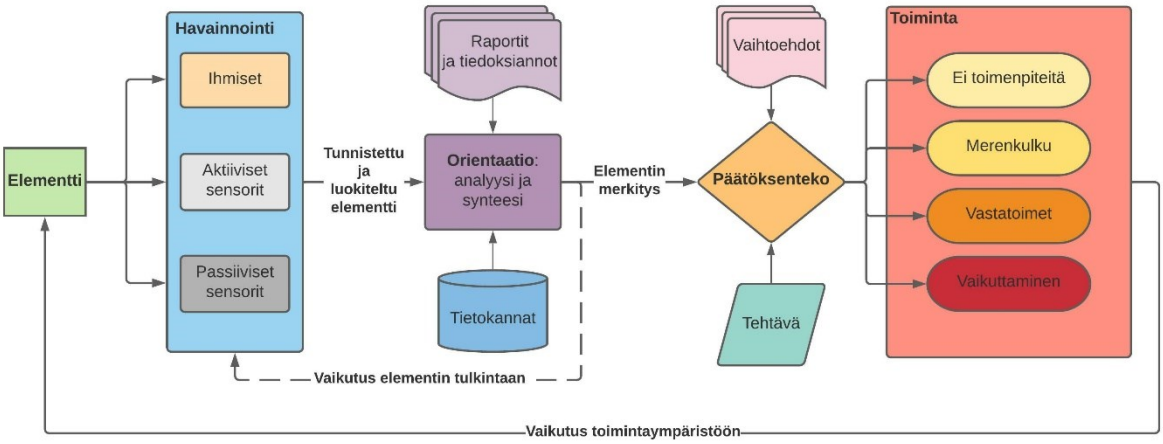
Etäohjausaseman ei tarvitse olla ranta-asema, vaan ohjausasema voi sijaita yhtä hyvin niin kutsutulla emäaluksella. Tästä seuraa suoraan, että sotilaallisen autonomisen järjestelmän tulee olla tiedonsiirtokyvyltään robusti sekä vikasietokykyinen myös potentiaalisen vastustajan tuottaman häirinnän osalta, jolloin ROC konseptissa visioitu satelliittiyhteys ei voi olla ainut käytettävissä oleva tiedonsiirtokanava. Merivoimien aluskaluston tiedonsiirtovaihtoehtojen kirjo on kattava ja saattaa vastata ROC-tyyppisten asemien käyttöönottoon, mutta asiaa ei tarkastella enempää tutkimuksen turvaluokituksen puitteissa. Elektronisen sodankäynnin ja kyberturvallisuuden näkökulmista miehittämätön, autonominen järjestelmä on altis paitsi komentoyhteyksien katkaisemiselle myös haittaohjelmille tai jopa ohjauksen kaappaamiselle vastustajan toimenpitein.

Tämän tutkimuksen näkökulmasta autonomisten järjestelmien tulee kyetä toteuttamaan tiettyjä, tekoälyllä toteutettuja osatoimintoja riittävällä tarkkuudella ja luotettavuudella etävalvonnan ja -ohjauksen mahdollistamiseksi siten, että järjestelmän käytettävyys lisää sotilaallista suorituskykyä. Käytännössä tämä tarkoittaa, että autonomisen järjestelmän tulee kyetä tekemään paikan päällä riittävää havainnointia, orientaatiota ja matalan tason päätöksentekoa kuten reitinvalintaa, ja välittämään näistä tiedonsiirrollisesti kevyttä dataa etäasemalle ihmisen kontrollin ja valvonnan mahdollistamiseksi, mutta operaattorin kuormitusta tarpeettomasti lisäämättä. Nämä toimenpiteet toteutetaan tekoälyn menetelmin. Seuraavassa luvussa tarkastellaan eri menetelmien soveltuvuutta Suomen laivaston toimintaympäristön vaatimuksiin ja toimijoiden tehtäviin muun muassa näissä kategorioissa.

2.6 Tekoäly sotilasympäristössä

Tekoälyn sotilaallisten mahdollisuuksien kartoittamiseksi tässä työssä segmentoidaan tekoälyn mahdolliset soveltamisalueet strategin ja Yhdysvaltojen ilmavoimien everstin John Boydin esittelemän ja nykyään yleisesti käytetyn OODA-silmukan teoriakehykseen. OODA tulee sanoista *Observation, Orientation, Decision* ja *Action*, kuvaten päätöksenteon silmukkaa, jossa havainnot prosessoidaan orientaatiossa synteesiksi tai analyysiksi, josta juontuu päätös tai hypoteesi, jonka mukaisesti tehdään toimenpiteitä ympäristöön vaikuttamiseksi. OODA-silmukka voidaan mieltää C2 *(command and control*) prosessiksi eli sodankäynnin johtamis- ja ohjausprosessiksi. Alun perin Boyd on kuvannut silmukalla ilmataistelun päätöksentekoa eräänlaisena metaforisena jatkeena biologiselle ärsykkeiden ja reaktioiden syklille. Toimiva OODA-silmukka mahdollistaa onnistuneen adaptaation toimintaympäristöön. [84 s. 383-385] Boydin mukaan taistelutilanteissa voiton ratkaiseva tehokkuus saavutetaan siten, että oman OODA- eli päätöksentekosilmukan on oltava vastustajan vastaavan silmukan ”sisällä” eli tätä tehokkaampi, mahdollistaen oman proaktiivisen toiminnan ja häiriten vastustajan orientaatiota tätä nopeammilla päätöksillä ja muutoksilla. [84, s. 240] Liitteessä 2 olevan kuvan mukaisesti havaintoihin (*observations*) lukeutuvat ulkopuolinen informaatio, tapahtumaolosuhteet ja ympäristön vuorovaikutus. Teknisesti havainnot koostuvat erilaisista sensorihavainnoista, jotka muodostavat tilannekuvan ja -ymmärryksen kokonaisuuden. Alun perin vaiheen nimi on ollutkin ”*sensing*”, viitaten paremmin erilaisiin sensorihavaintoihin, mutta muutettu muotoon ”*observations*”, jotta silmukan nimi ei ole SODA. Orientaatiossa muodostetaan havaintojen pohjalta analyysi ja synteesi näiden merkityksestä suhteessa uuteen informaatioon kuten havaintojen poikkeavuuteen aiemmista tapahtumista, aiempiin kokemuksiin, kulttuurisiin traditioihin kuten koulutustaustaan ja toimintatapoihin sekä geneettiseksi perimäksi nimettyyn tekijään, jolla viitataan yksilöllisiin ominaisuuksiin ja näiden aiheuttamiin piirteisiin ja rajoitteisiin. Analyysin ja synteesin perusteella muodostetaan hypoteesi halutusta proseduurista eli päätös toimintatavasta kyseisten havaintojen pohjalta tehtyyn synteesiin vastaamiseksi. Päätöksen hypoteesi testataan toiminnalla, joka toimeenpanee päätöksen ja käynnistää uuden iteraation silmukassa päätöksen vaikutusten havainnoimisesta muutoksina toimintaympäristössä. [84 s. 384-385]

Tohtori Leslie M. Blaha Yhdysvaltojen ilmavoimien tutkimuslaboratoriosta on tehnyt OODA:sta uuden, inhimillisen päätöksenteon sijaan teknisen tai ”koneellisen” konseptin. Tässä versiossa, joka on liitteessä 2 Boydin alkuperäisen silmukan yhteydessä, suurimmat erot syntyvät orientaation teknisistä ominaisuuksista: millä käyttöjärjestelmällä toimitaan, mikä on järjestelmän mikrosirurakenne, mitkä ihmismallit ohjaavat toimintaa ja miten koneellinen äly käsittelee sitä. Blahan lähtökohta on toteuttaa ihmisen päätöksenteon kaltainen prosessi, joka avustaa ihmistä päätöksenteossa. [15] Tässä tutkimuksessa tekoälyä tarkastellaan kuitenkin myös itsenäisenä toimijana, jota ihminen valvoo. Jotta tekoälyn käyttöönottoa voidaan hahmottaa OODA-silmukan koneellisessa konseptissa, tarkastellaan tässä työssä silmukan vaiheita SSM keinoin oleellisten systeemien hahmottamiseksi. Systeemeistä pyritään hahmottamaan ne vaiheet ja toiminnot, joihin voidaan hyödyntää tekoälyn menetelmiä, sekä kartoittaa hypoteesiavaruudet käytettävistä menetelmistä.



KUVA 15: OODA-silmukka laivaston toimintaympäristössä

Tutkimuksessa tarkastellaan mahdollisuuksia hyödyntää tekoälyä kuvassa 15 esitettyjen vaiheiden ja toimenpiteiden toteuttamisessa yksittäisinä kategorioina. Systeemi, joka kykenee toteuttamaan koko OODA-prosessin itsenäisesti, voidaan nähdä täysautonomisena systeeminä, riippumatta siitä, koostuuko järjestelmä sitten koneista ja ihmisistä tai pelkästään koneista. Näin ollen linkki tekoälymenetelmien kategorisen tarkastelun ja autonomisten merellisten järjestelmien välillä on siinä, että teknisesti autonomisen järjestelmän on koostuttava kaikissa kategorioissa tapahtuvien toimenpiteiden yhteistoiminnasta, muodostaen järjestelmä, joka kykenee paitsi havainnoimaan ympäristöään myös tuottamaan havainnoista analyysin ja tämän perusteella muodostamaan joko itsenäisesti tai avustetusti päätöksen toimintatavasta, jolla toimintaympäristöön vaikutetaan. Yhtä lailla yksittäisissä kategorioissa toimivat tekoälyn menetelmät voivat avustaa, tehostaa tai uudistaa työskentelyä muuallakin Merivoimien tehtävissä kuin taistelualuksen automatisoituvassa tehtäväkentässä. Tästä syystä tarkastelu on geneerinen, vaikka näkökulma on rajattu alustaisteluosaston tehtävien suorittamiseen.

2.7 SSM ja CRISP-DM

Tässä alaluvussa esitellään yksityiskohtaisemmin päätutkimusmenetelmä SSM sekä CRISP-DM prosessi, jota hyödynnetään liitteinä olevissa kokeellisissa tekoälymallinnuksissa ja johon peilataan kehitystarpeita työssä tarkasteltavissa kategorioissa.

Kuten johdannossa kuvattiin, tähän työhön mukailtu SSM metodologia sisältää seitsemän vaihetta kuvan 1 mukaisesti. Operaatioanalyysin ja kovan systeemimetodologian sijaan SSM valikoitui menetelmäksi yksinkertaisesti siitä syystä, ettei kaikista tarkasteltavista ongelmatilanteista löydy hyödynnettävää dataa, jolloin validien matemaattisten mallien luominen ja ratkaisu on käytännössä mahdotonta. Vaikkakin SSM on tarkoitettu pääasiallisesti inhimillisten systeemien mallintamiseen, konseptointiin ja parantamiseen, ovat sen juuret operaatioanalyysissa, systeemianalyysissa ja kovassa systeemimetodologiassa, jolloin teknisten systeemien tarkastelu on edelleen mahdollista hyödyntäen abstraktimpaa tarkkuutta tarvittavan datan puuttuessa. Lisäksi on huomioitava tekoälyn tarkoitus: aiemmin kuvatusti tekoälyn on tarkoitus tukea tai korvata ihmisen tekemää työtä. Näin ollen, SSM kartoittaessa ja soveltuessa nimenomaisesti inhimillisten systeemien toiminnan tarkasteluun ja tehostamiseen, voidaan argumentoida sen soveltuvan tekoälyn käyttötarkoitusten kartoittamiseen, vastaten nimenomaisesti tämän työn tutkimuskysymyksiin.

Peter Checklandin mukaan alkuperäinen SSM metodologian ensimmäinen osa, ilmaisu, koostuu vaiheista 1 ja 2: ongelmallisen tilanteen rakenteen tai systeemin osittamisesta epämuodolliseksi ja tämän epämuodollistetun tilanteen ilmaisemisesta. Tarkoitus on muodostaa kattavin mahdollinen käsitys ongelma-alueesta tai -tilanteesta, jotta seuraavissa vaiheissa voidaan tunnistaa ongelman ratkaisemiselle oleelliset näkökulmat, joista ongelman tarkastelua jatketaan. Yleensä käsitys muodostetaan niin sanottuna ”*rich picture*” kuvana, johon tarkastelija tai tarkastelijoiden ryhmä kokoaa käsitellen ongelmatilanteen rakenteellisia elementtejä, prosessien elementtejä sekä tilanteen ilmapiiriä. Tässä työssä tarkastelut ovat työn tekijän omia näkemyksiä kyseisistä systeemeistä. Ideaalitilanteessa ilmaisusta kyetään hahmottamaan oleelliset systeemit seuraavassa vaiheessa, jossa määritetään oleellisten systeemien perimmäiset määritelmät (”*root definitions of relevant systems*”). Checklandin mukaan perimmäisten määritelmien testaamisessa kannattaa hyödyntää CATWOE muistisääntöä, jossa vastataan kuuteen kysymykseen:

1. Ketkä tai mitkä ovat systeemin muutosten hyötyjät tai kärsijät? (*Customers*)
2. Ketkä tai mitkä ovat systeemin toimijat? (*Actors*)
3. Mitä täytyy tehdä muutostavoitteen saavuttamiseksi? (*Transformation process*)
4. Mitä muutoksella yritetään saavuttaa? (*Weltanschauung*, näkymä)
5. Kuka systeemin omistaa? (*Owners*)
6. Millaisia rajoituksia systeemin ympäristö tuottaa? (*Environmental constraints*)

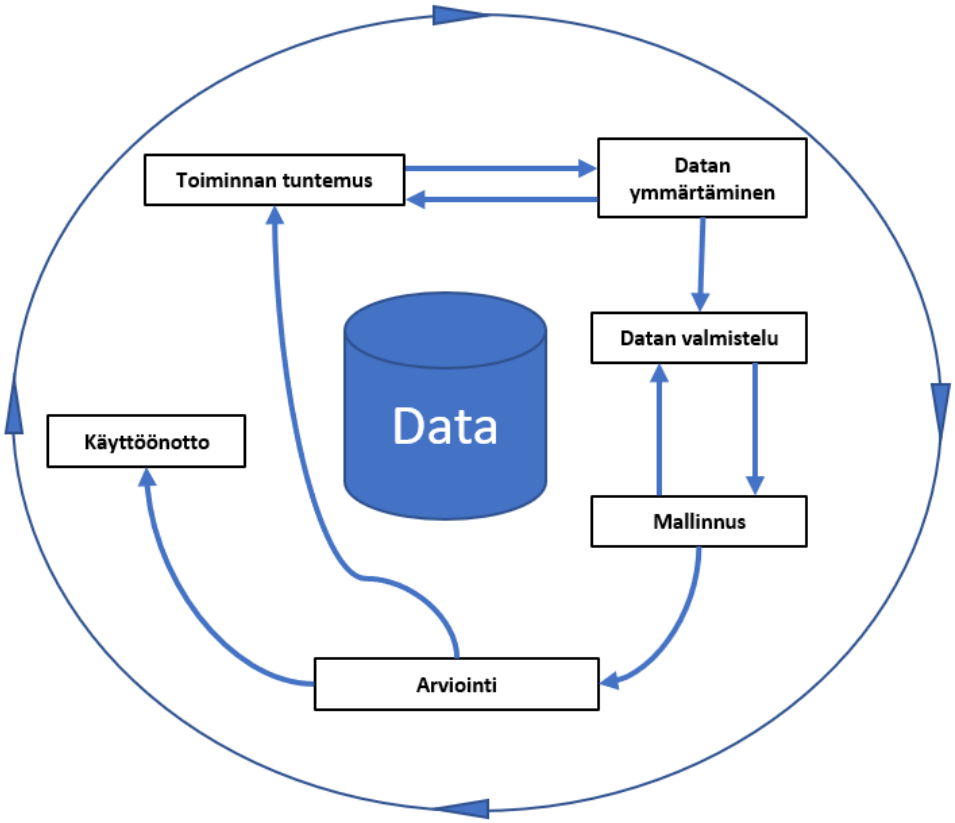
Kun oleellisten systeemien määritelmät on luonnosteltu, siirrytään muodostamaan konseptimalli tai -mallit. Konseptimallin luomiseksi tulee selvittää, mitä toimintoja tulee suorittaa missäkin järjestyksessä halutun toimenpiteen suorittamiseksi. Koska konseptimalli on eräänlainen aktiivisuusmalli, tulee se mallintaa käyttäen verbejä, jotka ovat oleellisia määritelmän kuvaaman systeemin toiminnalle. Seuraavassa vaiheessa konseptimallia verrataan todellisuuteen. Systeemin todellisuus eli ongelmatilanne on kuvattu vaiheessa kaksi, jolloin laadittua konseptimallia kuvaannollisesti verrataan kyseiseen ongelmatilanteeseen ja tarkastetaan, vastaako konsepti kyseiseen tilanteeseen

ja siinä olevaan ongelmaan. Lopulta kartoitetaan soveltuvat, toteuttamiskelpoiset muutokset systeemiin, sekä toimenpiteet ongelmatilanteen parantamiseksi. [13 s. 128-141, 165-181]

Tässä työssä ongelmatilanteet määritellään kategorioittain, jonka jälkeen tarkastellaan ongelmatilanteessa havaittavia mahdollisuuksia hyödyntää tekoälymenetelmiä, eli niin sanotusti oleelliset systeemit. Kun oleelliset systeemit on määritetty, kartoitetaan kyseisiin systeemeihin soveltuvat tekoälymenetelmät, joiden pohjalta luodaan konseptimalli ongelmatilanteen parantamiseksi. Tekoälyn osalta pyritään siis hahmottamaan kuhunkin alaongelman soveltuvia tekoälymenetelmiä, joita testataan kokeellisesti niiltä osin, kun saatavissa on käyttökelpoista dataa tekoälymallien tuottamiseen. Viimeisessä toteutettavassa vaiheessa kartoitetaan muutokset, joita hyödyntäminen tai toteutettujen mallien käyttö ja parantaminen edellyttävät. Itse toimenpidettä tilanteen parantamiseksi ei toteuteta tämän työn puitteissa.

Niiltä osin, kuin tämän työn puitteissa on mahdollista, demonstroidaan konseptimalleja luomalla tekoälymalleja CRISP-DM mukaisesti. CRISP-DM muodostuu kuvan 16 mukaisesti kuudesta vaiheesta. Ensimmäinen on nimetty alun perin muotoon ”*business understanding*”, mutta tähän ympäristöön se on käännetty toiminnan tuntemukseksi, joka pitää sisällään toimijoiden, toimintatapojen, toimintaympäristön ja toiminnan tavoitteen tuntemisen. Tämä vaihe keskittyy projektin päämäärien ja vaatimusten tunnistamiseen sekä muotoilemiseen datan louhinnan ongelmaksi, muodostaen alustavan suunnitelman päämäärän saavuttamiseksi. Toisena vaiheena on datan ymmärtäminen, jossa saatavilla olevaa dataa käsitellään ja tarkastellaan sen ominaisuuksien ymmärtämiseksi. Tarkoitus on muodostaa alustava käsitys datasta mahdollistaen mielenkiintoisten piilevien piirteiden tunnistamisen ja niistä johdettavien hypoteesien tekemisen. [12 s. 10] Käsitys ja datassa piilevät korrelaatiot eri piirteiden välillä voidaan havaita ja havainnollistaa esimerkiksi aiemmin esitetyillä regressiotyökaluilla tai rypästämisellä.

Datan valmistelu käsittää kaikki ne toimenpiteet, joita lopullisen eli mallinnustyökaluille syötettävän data-aineiston muodostaminen edellyttää. Nämä toimenpiteet voivat olla toistuvia eivätkä välttämättä noudata tiettyä järjestystä. Yleisiä toimenpiteitä ovat muun muassa taltiointi, komponenttianalyysi, datan siistiminen ja annotointi, Mallinnusvaiheessa valmisteltu data syötetään hypoteesiavaruudesta valitulle, yhdelle tai useammalle mallinnustekniikalle. Kun malli tai mallit ovat luotu arvioidaan niiden laatu ja oikeellisuus projektin päämäärien kannalta. Tärkeintä on tunnistaa, mikäli jokin oleellinen toiminnan ja päämäärien osa-alue on jäänyt huomioimatta. Vaiheen päätteeksi tulee muodostua päätös datanlouhinnan tulosten eli mallin käyttöönotosta tai vaihtoehtoisesti uusista iteraatioista prosessissa. [12 s. 10-11]



KUVA 16: CRISP-DM vaiheet. Kuva mukailtu Chapman ym. julkaisun pohjalta [12 s. 10]

CRISP-DM ja sitä noudattavan projektin viimeinen vaihe on käyttöönotto, jossa louhinnan tuloksena tuotettu malli hyödynnetään. Käyttöönotto voi esimerkiksi olla raportti, joka lisää ymmärrystä jostain ilmiöstä tai malli, joka muodostaa päätöksentekoa tukevan työkalun. [12 s. 11]

Aihealueeseen liittyen on huomionarvoista, että sekä SSM, että CRISP-DM vastaavat skemaattisesti operaatioanalyysin mallinnuksen prosessia. Sotilaallinen operaatioanalyysi määritellään tieteelliseksi metodiksi tuottaa asevoimille kvantitatiivinen päätöksenteon perusta suorittaa operaatioita. Operaatioanalyyttisellä sovellusalueella tarkoitetaan ongelmaa, johon voidaan hyödyntää operaatioanalyyttisiä menetelmiä ja työkaluja. Analyysin on tarkoitus parantaa jonkin järjestelmän toimintaa tuottamalla joko teknologisia parannuksia itse järjestelmään tai muuttamalla olemassa olevia toimintatapoja saman järjestelmän paremman tehokkuuden saavuttamiseksi. Parannusehdotusten löytämiseksi ja testaamiseksi operaatioanalyysin metodologiassa käydään läpi prosessi, jossa ensin formuloidaan ongelma-alue, johon kehitetään matemaattinen malli. Seuraavaksi kerätään dataa ja ratkaistaan kehitetty malli, jonka jälkeen malli validoidaan. Mikäli se todetaan validiksi, seuraa evaluaatio ja implementaatio. Jos malli ei ole validi, palataan johonkin aiempaan vaiheeseen. [85 s. 6-7; 86; 87 s. 7] Näin ollen prosessin vaiheet ovat varsin yhtenevät CRISP-DM prosessin kanssa. Pohjimmiltaan kaikki kolme metodologiaa ovat iteratiivisia ja dataa hyödyntäviä mallinnusprosesseja. Suurin suorituksellinen ero CRISP-DM ja operaatioanalyysin välillä on mallin luonnin ajoitus prosessin syklissä, joka on tulkinnallisesti samanlainen ero kuin tilastotieteen ja koneoppimisen välillä, sillä operaatioanalyysissä malli luodaan kuvaamaan ongelmaa, kuten jakauma oletetaan kuvaamaan jotain aineistoa, jotta voidaan ratkaista dataan liittyviä muita ongelmia.

Datan louhinnassa pyritään yleistetysti löytämään datassa piilevä, usein vektorietäisyyksiin pohjautuva, matemaattinen malli ja valjastamaan se käyttöön, kun taas operaatioanalyysissa malli on usein muuttujien määrältä yksinkertaisempi mutta kuvattu ongelma saattaa olla laajempi ja huomattavasti yleiskäsitteisempi. Molemmat prosessit pyrkivät kuitenkin tuottamaan datan pohjalta laskennallisen ratkaisun. Mallin matemaattisuus ja ratkaisun laskennallisuus ovat merkittävin ero SSM kanssa, vaikka sen historiassa on yhtymäkohtia operaatioanalyysin kanssa [13 s. 72, 134-147]. Sen sijaan SSM on yhtenevä operaatioanalyysin pyrkimyksessä esimerkiksi arvioimaan uusien teknologioiden vaikutusta ja mahdollisuuksia tai parantamaan vanhoja toimintatapoja, mutta pehmeämmästä, ihmiskeskeisestä näkökulmasta.

Vaikka metodologioissa itsessään on yhteneväisyyksiä, on ratkaistava ongelma-alue ja ratkaisusta saatava malli näiden välillä usein merkittävästi erilainen. CRISP-DM prosessia hyödynnetään siitä syystä, että se on varta vasten tekoälyn hyödyntämiseen kehitetty ja siinä vakiintunut. Yhteneväisyydet operaatioanalyysiin nähdään tässä opinnäytetyössä validoivina, menetelmien valintaa tukevina ja tekoälytutkimuksen sekä asevoimien operatiivisen toiminnan synergiaa korostavina.

CRISP-DM sovelletaan täysimääräisesti kahteen datan louhinnaksi miellettävään ongelmaan, jotka esitellään seuraavassa luvussa ja jotka on kuvattu kokonaisuuksina liitteissä 3 ja 6. Yleinen toiminnan tuntemus merisodankäynnistä ja rajoitetun merialueen kuten Itämeren muodostamista erityispiirteistä luodaan seuraavan pääluvun alussa. Ongelma-aluekohtainen toiminnan tuntemus käydään läpi aihealueittain tarkemmin, jolloin mahdollistetaan datan ymmärrys ja valmistelu.

3 Tekoäly Suomen laivastojoukoissa

Tässä luvussa sidotaan edellisissä luvuissa kartoitetut aihepiirit ja tietopohja Suomen merivoimien ja erityisesti laivastojoukkojen toimintaan, kartoittaen tekoälyn käyttöönottomahdollisuuksia laivaston taisteluosaston toimintaympäristössä. Tarkastelun lähtökohtana on kuvan 15 mukaillun OODA-silmukan teoriakehys. Luku vastaa tutkimuksen toiseen alatutkimuskysymykseen. Alaluvussa esitellään havaintojen pohjalta yhteenvedot ja konseptit erilaisista ratkaisumalleista. Konseptilla tarkoitetaan tässä yhteydessä viimeistelemätöntä luonnosta eli hahmotelmaa tekoälyn hyödynnettävyydestä kyseiseen ongelma-alueeseen tarkastelun abstraktiotasolla.

3.1 Suomen laivastojoukkojen tehtävät ja toimintaympäristö

Sir Julian Corbett, yksi merisodankäynnin vaikutusvaltaisimpia ja arvostetuimpia teoreetikoita, tulkitsee merisodan tärkeimmiksi piirteiksi meren miehittämättömyyden ja siitä juontuvan meriyhteyksien suojaamisen tärkeyden. Merialueen normaali tila on se, ettei se ole minkään osapuolen hallinnassa. [88; 89 s.54-56] Corbettin näkemyksessä merenherruus on suhteellinen käsite, jonka asteet riippuvat kyvystä saavuttaa merenherruus ja vastustajan kyvystä kiistää se. Näin ollen merenherruus voi olla täydellistä tai rajoitettua ajan ja paikan suhteen, joista täydellinen merenherruus on pääasiassa hypoteettinen ja äärimmäisen vaikea saavuttaa. Corbett ei myöskään uskonut suuriin ratkaisutaisteluihin merellä, vaan hänen näkemyksessään tärkeintä merisodassa on saada kontrolli vastustajan merellisistä yhteyksistä joko tuhoamalla tämän sota- ja kauppa-aluksia tai muodostamalla saartoja. [89 s. 57, 59] Corbett toteaa merellisten yhteyksien omaavan positiivisen ja merkittävyydeltään vaihtelevan mutta kiistattoman arvon kaikille meriyhteyksiä omaaville valtioille. [89 s. 57] Suomen tapauksessa kyseinen arvo on erittäin merkittävä, joka heijastuu Suomen merivoimien tehtäviin, joita ovat merialueiden valvonta ja alueloukkauksien torjuminen, meriyhteyksien turvaaminen sekä merellisten hyökkäysten torjunta. [7] Nämä tehtävät korreloivat Corbettin näkemyksiin merenherruuden merkityksestä, meriyhteyksien tärkeydestä ja niihin vaikuttamisen muodoista ja kyseisen vaikuttamisen torjumisesta.

Suomen laivastojoukkojen toiminta-alue painottuu edellä mainittujen tehtävien toteuttamiseksi Itämerelle ja pääasiassa Suomen merialueelle tai sen välittömään läheisyyteen. Itämeri rajoittuu lännessä Tanskan salmiin, muodostaen yhden maailman suurimmista sisämeristä ja murtovesialtaista. Itämeren pohjan topografia on vaihteleva, mutta pääasiassa merialue on matala. Meriyhteyksien merkitystä alueella korostavat muun muassa se, että 15% maailman merikonttiliikenteestä tapahtuu Itämerellä. Alueella on päivittäin noin 2000 suurta kauppa-alusta, jotka kuljettavat noin 40% Ruotsin hyödykkeistä, noin 70% Venäjän konttiliikenteestä ja yli 80% Suomen ulkomaankaupan kuljetuksista. [90 s. 226-228; 91] Itämeren keskisyvyys on vain 54 metriä ja suolaisuus vaihtelee Tanskan salmien 10-33‰ vaihteluvälistä pohjoisen Perämeren lähes makeaan veteen, ollen keskiarvioisesti 7‰. Sääolosuhteiden valossa merkitsevä aallonkorkeus voi teoriassa nousta 9.5 metriin, jolloin korkein yksittäinen aalto nousee yli 15 metrin korkuiseksi, korkeimman mitatun ollessa 2011 tehdyn tilaston perusteella 8.2 metriä. Keskimääräinen aallonkorkeus on kuitenkin $\leq$ 1.5 metriä. Keskisellä Itämerellä merkitsevä aallonkorkeus ylittää 4 metriä alle 10% ajasta. [92 s. 103-105; 93]

Itämeri on myös etäisyyksiltään pieni, esimerkiksi Suomenlahti on kapeimmillaan noin 50 kilometrin levyinen. Rannikkoalueet ovat rikkonaiset ja erityisesti Suomen ja Ruotsin rannikkoalueet ovat maailman tiheimpiin kuuluvan saariston kattamia [94 s. 9]. Veden lämpötila vaihtelee myös huomattavasti pintaveden lämpötilan nousten kesäisin jopa yli 20 celsiusasteen lukemiin ja talvella osittain jäätyen, keskimääräisen jääpeitteen ollessa noin kaksi viidesosaa Itämeren pinta-alasta. Ankarina jäätalvina Itämeren pinta-alasta jäällä peittyy yli puolet. [95]

Koska Itämeri on pieni merialue, joka on kuitenkin usean valtion elintärkeä logistinen solmukohta, on se rantavaltioidensa ja näiden liittolaisten toimesta merkittävän valvonnan ja sotilaallisen läsnäolon kohteena. [96] Suurille pinta-aluksille Itämeri tarjoaa huonot mahdollisuudet operoida tulematta havaituksi, sillä valvottava pinta-ala on varsin pieni ja rannikoilla olevat kiinteät valvonta-asemat pystyvät kattamaan suurimman osan alueesta. Pohjoisella ja eteläisellä Itämerellä on alueita, joiden valvominen kiinteillä rannikkoasemilla ei ole tutkakantamien puolesta mahdollista. Sukellusveneille Itämeri tarjoaa paitsi haasteita myös mahdollisuuksia [8 s. 129]. Vaihtelevat veden lämpötilat ja lämpötilaeroista muodostuvat harppauskerrokset, matalat vedet ja erilaiset pohjanmuodot tekevät sukellusveneiden etsinnän erittäin haastavaksi pinta-aluksille.

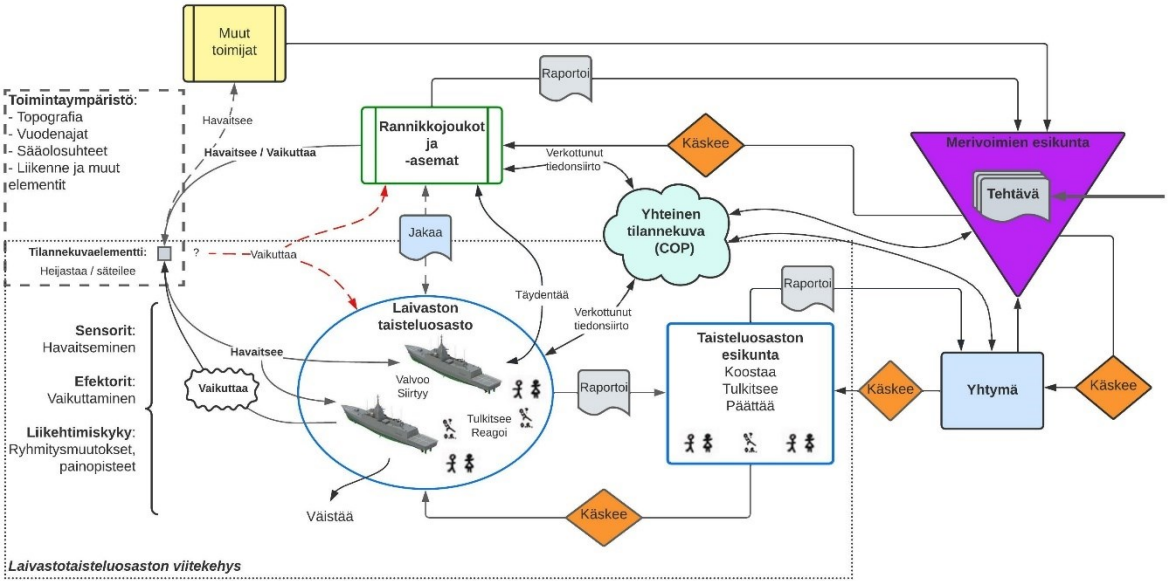
Toimintaympäristönä Itämeri muodostaa edellä kuvatusti tiiviin ja vilkkaasti liikennöidyn ympäristön, jonka maalitilannekuva on aktiivinen runsaan kauppameri- ja huviliikenteen vuoksi. Valtameriin verrattuna Itämeri on haastava sensoriympäristö, sillä jokaisessa dimensiossa on enemmän haittasignaaleja kuin kiinnostavia hyötysignaaleja. Lyhyiden etäisyyksien takia tilannekehitykset ovat nopeita ja äkillisiä. Näin ollen Itämerellä tapahtuva merisodankäynti edellyttää tehokasta OODA-silmukkaa, jossa kyetään havaitsemaan toimintaympäristön kriittiset elementit tehokkaasti, ennakoimaan tilannekehitys ja siihen liittyvät anomaliat mahdollistaen nopean ja dataan perustuvan päätöksenteon ja sen seurauksena oikea-aikaiset ja, onnistuessaan, oikeat toimenpiteet.

Seuraavaksi käsitellään SSM mukaisesti kukin kuvan 14 esittämä OODA-vaihe muodostamalla ensin käsitys ongelmatilanteesta, rajaamalla jokainen tarkastelu ongelmatilanteessa tunnistettuihin oleellisiin systeemeihin, luomalla hypoteesi soveltuvista tekoälymenetelmistä kyseisen ongelma-alueen tehostamiseksi tai kehittämiseksi ja tuottamalla siten syöte seuraavaan prosessin vaiheeseen ja sen systeemitarkasteluun. Näin ollen lopputuotteena on koko OODA-silmukan kattava, tekoälyä hyödyntävä systeemimalli ja toimintaehdotukset sen mahdollistamisesta.

Sotatieteelliset tarkastelut rajataan usein esimerkiksi rauhanajan tai sodanajan oloihin analysoitavan toimintaympäristön selkeyttämiseksi. Tämän tutkimuksen kannalta esimerkiksi kriisin vaiheella ei kaikkien kategorioiden osalta ole väliä, vaan mallinnetut systeemit ja niihin esitetyt parannukset ovat yleispäteviä. OODA-silmukan malli on validi riippumatta siitä, tarkastellaanko sitä sodanajan taisteluiden päätöksentekoprosessina vai rauhanajan aluevalvonnan prosessina. Lisäksi tarkastelun lähtökohtana olevat inhimillisen toiminnan systeemit, joihin tekoälyn menetelmiä sovitetaan, ovat laivaston osalta teknisesti muuttumattomia. Päätöksentekoprosessin käytössä olevat vaihtoehdot luonnollisesti riippuvat esimerkiksi siitä, ollaanko kriisissä ja sotatilan alaisia, mutta itse prosessit, joilla tilannekuva ja siihen liittyvät päätökset muodostuvat, toimivat samoin kuin rauhan aikana.

Käytettävien esimerkkien ja systeemien laajuuden vuoksi tutkimuksessa on kuitenkin valittu tarkastelun lähtökohdaksi laivaston pintataisteluosasto esikuntineen. Pintataisteluosaston tarkasteltava tehtävä eri kategorioissa on merioperaation suorittaminen Itämerellä. Päätöksenteon kategoriassa tarkastelu painottuu sodankäyntiin, sillä rauhanajan toimenpiteiden kirjo on suppeampi ja suoraviivaisempi. Myös toimenpiteiden tarkastelu painottuu sotilaalliseen voimankäyttöön.

Rajauksesta huolimatta tarkastellut ongelmatilanteet ovat yhteneviä myös muiden joukkojen osalta, sillä periaatteet pysyvät muuttumattomina näiden välillä. Alla olevassa kuvassa 16 on koostettu SSM mukainen rikas kuva Suomen merivoimien systeemien systeemin rakenteesta, toteuttamasta prosessista ja siihen liittyvistä ongelmatilanteista OODA-silmukan näkökulmasta.



KUVA 17: OODA-silmukan ongelmatilanne (”*rich picture*”) Merivoimien systeemien systeemissä

Kuvassa 17 on pyritty muodostamaan tiivis mutta kattava käsitys keskeisistä toimijoista ja näiden keskinäisistä vuorovaikutuksista merioperaatioiden suorittamisessa. Ihmissymbolit yhdistettynä toimintoja kuvaaviin verbeihin kuvaavat tämän opinnäytetyön keskiössä olevia inhimillisiä toimintoja, joiden tehostamiseen ja uudistamiseen tekoälyn menetelmiä sovitetaan. Kuvaan visualisoituja toimijoita ei ole rajattu pelkkiin laivaston yksiköihin, mutta jatkossa tarkastelun lähtökohta on alustaisteluosaston näkökulma tai siihen sisältyvä geneerinen viitekehys, kuten sensorijärjestelmä, joka on periaatteellisesti identtinen tarkasteltavasta joukosta riippumatta. Kuvasta puuttuvat myös toimintaa ohjaavat asiakirjat kuten lait, normit ja ohjeet, sillä ne nähdään tekoälyn näkökulmasta yhtenevinä niin inhimilliselle kuin koneelliselle toimijalle, muodostaen toimintamalleja ja reunaehtoja, joiden mukaisesti kuvan kuvaamat toiminnot on toteutettava. Kuvasta johdetaan oleelliset alasysteemikuvat OODA-kategorioiden tarkastelua varten seuraavissa alaluvuissa. Alasysteemikuvien perusteella on tarkoitus hahmottaa työtä ja toimintoja, joita voidaan tehostaa tai uudistaa tekoälyn menetelmin, sekä arvioida eri menetelmien soveltuvuutta ja haasteita.

Pehmeässä systeemimetodologiassa on aiemmin mainitusti oleellista hahmottaa tarkasteltavassa systeemissä tapahtuva tekeminen ja toiminta verbeinä. Tässä opinnäytetyössä tarkasteltava toiminta sisältyy ylätasoltaan tarkasteltaviin kategorioihin, jotka on jo jaoteltu tarkasteltavien toiminnallisuuksien mukaan. Näin ollen tekemisen hahmottamisen sijaan on oleellista hahmottaa kategorioissa ne toimenpiteet, joiden tehostamiseen voidaan hyödyntää tekoälyä, sekä kartoittaa mahdolliset muutokset eli vähintään teoreettisesti soveltuvat menetelmät joko kokeellisesti tai sisällönanalyysillä.

3.2 Havainnointi

Havainnoinnin ongelmatilanne liittyy tässä opinnäytetyössä tilannekuvan luomiseen. Tilannekuvalla tarkoitetaan esitettyä kuvaa vallitsevasta tilanteesta. Yhteinen tilannekuva (”*common operational picture”,* COP) voidaan määritellä jaetuksi yleiseksi representaatioksi operaatiota koskevasta tiedosta, joka on samalla identtinen esitys useamman kuin yhden johtoportaan jakamasta relevantista informaatiosta [97 s. 2]. Tilannekuvan esittämä informaatio on vallitseva tilannetietoisuus (”*situational awareness”*). Tilannetietoisuuden määritelmänä käytetään Yhdysvaltojen ilmavoimien entisen tutkimusjohtajan M. R. Endsleyn määritelmää: tilannetietoisuus on ympäristön elementtien havaitsemista tietyssä ajassa ja tilassa, tästä syntyvää ymmärrystä elementtien tarkoituksesta ja projektiota näiden elementtien statuksesta lähitulevaisuudessa. Endsleyn mukaan tilannetietoisuuden rakentuminen jakautuu näin ollen kolmeen vaiheeseen. [98 s. 36] Hollannin laivaston julkaisussa meritilannekuvaan ja tiedusteluun viitataan siten, että sodankäynnin johtamisjärjestelyn (”*command and control*”, C2) mahdollistaminen on puhtaasti vallitsevan tilanteen ymmärtämistä. Käsityksen mukaan tilannetietoisuus muodostuu kolmesta informaatioalueesta: omat joukot, luonnollinen ympäristö eli sää ja maasto, sekä muut toimijat kuten vastustajat, liittolaiset ynnä muut, jotka ovat osallisia tai läsnäolevia. Näihin liittyen tilannekuvan muodostaminen kuvataan datan hankinnan prosessina, jossa havainnot tuotetaan sensorein. Tiedustelu taas käsittää korkealaatuisen, analysoidun datan, jonka tuottaminen edellyttää usein erikoisasiantuntijoita, -yksiköitä tai -toimintaa. Tiedustelu voi liittyä nykytilanteeseen tai koostua taustatiedoista. [99] Näin ollen tiedustelu käsitetään osana tilannetietoisuuden muodostamista, mutta tiedustelu toimialana ei kuulu rajaukseen laivastojoukkojen tarkastelusta.

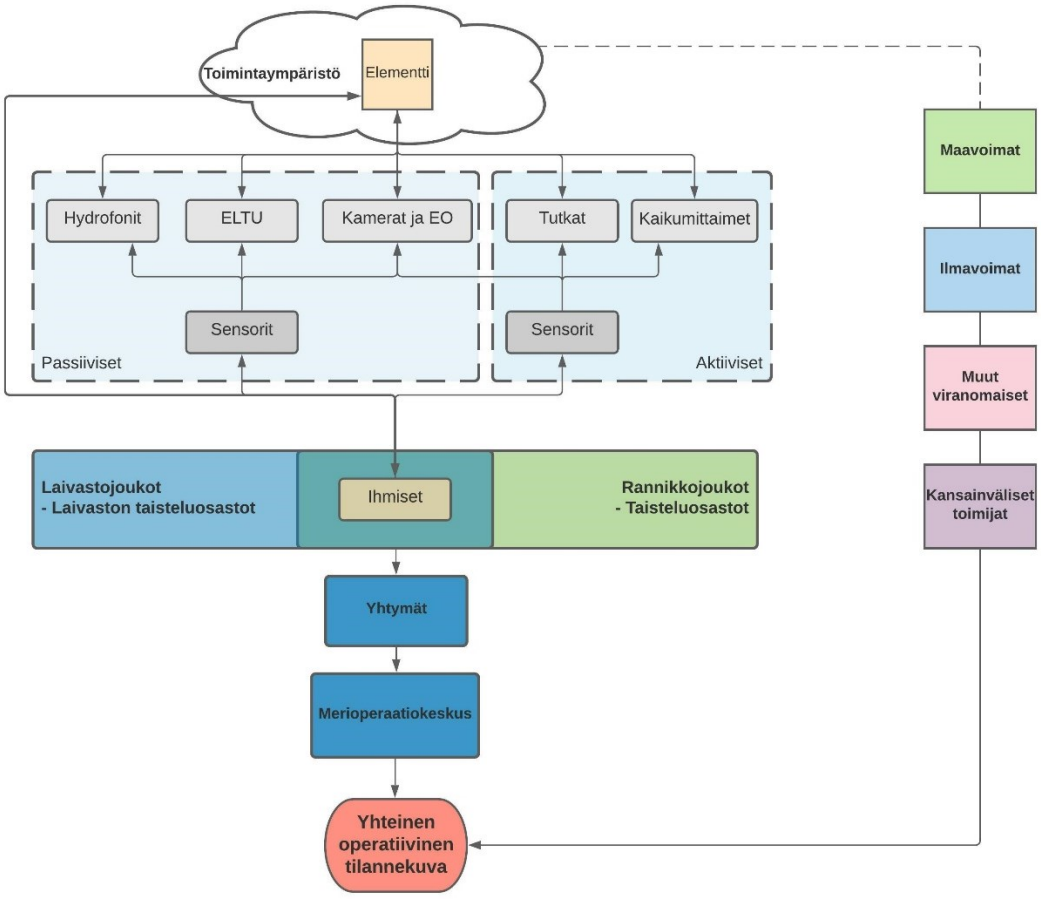
Yhdysvaltojen käsityksessä ISR-toiminnasta (”*intelligence, surveillance, reconnaissance*”) tiedustelutieto (”*intelligence*”) tuottaa kyvyn tulkita vastustajan aikomus. Valvonta (”*surveillance*”) on havainnoinnin prosessi, jossa tarkastellaan kiinnostuksen kohdetta toimintaa edellyttävien muutosten havaitsemiseksi. Tiedustelu (”*reconnaissance*”) kattaa ne keinot, joilla laivasto ja Merivoimien muut joukot tuottavat kovaa ja pehmeää dataa. Kovalla datalla tarkoitetaan tässä yhteydessä sensorien tuottamaa dataa kuten elektronista tiedustelua ja elektro-optisia sensoreita. Pehmeää dataa taas ovat esimerkiksi kirjoitetut raportit ja sanomat. ISR systeemi toteuttaa TCPED prosessia (*Task-Collect-Process-Exploit-Disseminate*), joka alkaa tiedonkeruun tehtävien antamisesta. Tämän jälkeen dataa kerätään ja se prosessoidaan informaatiotuotteeksi, jota hyödynnetään. Hyödyntämisen on tarkoitus muuttaa prosessoitu data käyttökelpoiseksi tiedustelutiedoksi ja -tuotteeksi, jotka jaetaan tarvitsijoille. [22 s. 170-171]

Tämän alaluvun tarkastelun keskiössä on Endsleyn näkemyksen ensimmäinen vaihe, ympäristön elementtien havaitseminen, sillä se vastaa päätössilmukan ensimmäistä vaihetta. Kaksi seuraavaa vaihetta kuuluvat määritelmällisesti orientaatiovaiheeseen.

Havainnon luokittelemisella tarkoitetaan tässä työssä elementin kuulumista johonkin tilannekuvaelementtiluokkaan. Tällaisia luokkia ovat esimerkiksi vedenalaisessa tilannekuvassa hylyt, kivet ja sukellusveneet, pintatilannekuvassa kauppa-alukset, huvialukset ja sota-alukset ja niin edelleen. Tunnistuksella tarkoitetaan havainnolle löydettyä, luokituksen sisäistä tunnistusta, kuten alustyyppiä tai -luokkaa. Tarkastelu vastaa myös ISR-toiminnan valvontaa ja tiedustelua, sekä TCPED-prosessin vaiheita tiedonkeruusta sen prosessointiin.

Endsleyn mukaan tilannetietoisuuden saavuttamisen ensimmäisessä vaiheessa havaitaan ympäristössä olevien, relevanttien elementtien status, attribuutit sekä dynamiikat [98 s. 37]. Sotilaallisessa kontekstissa ympäristö käsitetään operatiivisena toimintaympäristönä, jonka voidaan nähdä koostuvan maastosta, sääolosuhteista, siviilitekijöistä sekä vihollisesta. [100 s. 1-5] Käsitys maastosta ja sääolosuhteista muodostuu olemassa olevien aineistojen sekä sensoriverkoston havaintojen pohjalta, joista osa on orgaanisia ja osa muiden osapuolien hallinnoimia esimerkiksi sääolosuhteiden tarkkailussa. Käsitys vastustajasta luodaan maalitilanteen osalta sensorihavainnoin. Tämän lisäksi tilannekuvaan vaikuttavat muun muassa tiedustelutiedot ja muiden osapuolten välittämät havainnot. Tässä opinnäytetyössä rajataan ongelmatilanne kattamaan laivaston alusyksiköiden sensorihavainnot sekä näiden kanssa yhteensopivat maasijoitteisten järjestelmien sensorihavainnot niin sanotuksi tilannekuvasysteemiksi, jonka fokus on tuottaa eri sensorein signaalianalyysilla havaintoja vallitsevissa olosuhteissa. Sensorit jakautuvat aktiivisiin ja passiivisiin. Havainnoinnilla tarkoitetaan alusyksikön tai vastaavan järjestelmän kykyä kerätä ja tulkita sensoridataa ympäristön havainnoimiseksi ja maalitilannekuvan luomiseksi. Tilannetietoisuuden muodostamiseksi havainnointi on primääri vaatimus mille tahansa asejärjestelmälle sen toiminnan mahdollistamiseksi [101 s. 5].

Kuvassa 17 on esitetty ISR-toiminta Merivoimien toimintaympäristössä toteutettuna, sisältäen niin kovaa kuin pehmeää dataa tuottavia toimijoita. Tämän pohjalta on muodostettu kuvan 18 systeemi, jossa on havainnollistettu yhteisen operatiivisen tilannekuvan muodostamisen osatekijät. Tätä tarkastelua varten valitaan oleellisena alasysteeminä Merivoimien sensorijärjestelmien systeemi eli kova data, sulkien inhimilliset havainnot, viestit ja raportit pois tarkastelusta. Koska sensorit ovat toimintaperiaatteiltaan samoja käyttäjästä riippumatta, ei tarkastelua tarvitse erikseen rajata alusten sensorijärjestelmien systeemiin, vaan se voidaan tehdä yleisiä periaatteita tarkastellen.



KUVA 18: Yhteisen tilannekuvan luominen havaintojen muodostaman tilannetietoisuuden perusteella

CATWOE menetelmällä testatusti valitun sensorisysteemin hyötyjä on suoraan Merivoimat ja välillisesti, suuremmassa mittakaavassa, pääesikunta ja siten puolustusvoimien valvonta- ja tiedustelujärjestelmä. Merivoimien tilannekuvasysteemin ylemmän tason toimijoita ovat rannikko- ja laivastojoukot, kun toteuttavan tasan toimijoita ovat erilaiset yksiköt ja näiden sensorit sekä valvonta-asemien etäkäyttämät kiinteät tai siirrettävät sensorit. Tarkasteltava muutos järjestelmän toimintaan on tekoälymenetelmien sovittaminen tilannekuvaelementtien havaitsemiseen ja luokittelemiseen tunnistetun tilannekuvan luomiseksi. Kyseisen sensorijärjestelmän ja tilannekuvan muodostuksen prosessin omistaa Merivoimat, mutta Merivoimat ei kykene yksinään tuottamaan teknisiä muutoksia järjestelmiin. Merivoimien täytyy kuitenkin toimia muutoksen suunnittelijoina ja tilaajina. Ympäristön rajoitteita ovat muun muassa budjettikehys, toimintakulttuuri ja ohjesäännöt, tekoälyosaaminen ja järjestelmäintegraatio.

Kovan datan ja sensorien valitseminen oleelliseksi systeemiksi on luonnollinen rajaus esimerkiksi konenäön ja tekoälyavusteisen signaalikäsittelyn vuoksi, muodostaen yksiselitteisempiä tekoälyongelmia kuin pehmeän datan käsittely. Lähtökohtaisesti kaikki sensorityypit voivat olla niin alussijoitteisia kuin maa- tai ilmasijoitteisia, jolloin tarkasteltavat ratkaisut ovat lähtökohtaisesti yhteneviä kaikkiin vastaaviin ympäristöihin sijoitteisuudesta riippumatta, mutta tarkastelun lähtökohta on alussijoitteisissa sensoreissa niihin liittyvine nyansseineen. Merivoimien käyttämiä sensoreita ovat erilaiset tutkat, elektronisen tukemisen (ELTU, eng. ”*electronic support”*) sensorit, kamerat ja elektro-optiset sensorit sekä vedenalaiset kaikumittaimet ja kuuntelujärjestelmät. [101 s. 42, 83, 99, 196, 200]. Käytännössä kaikki sensorihavainnot perustuvat energian havaitsemiseen joko kohteen itsessään säteilemänä tai heijastamana. Energia etenee aaltoliikkeenä, ja haluttujen elementtien tunnistaminen niiden ympäristöstä edellyttää kykyä tunnistaa ja luokitella kyseinen energiamuoto sen taajuuden ja aallonpituuden mukaisesti. [101 s. 5] Tämä periaate pätee kaiken energian tulkitsemiseen, tehdään tulkinta sitten suorana signaalianalyysina tutka-antennin havaitsemasta paluusignaalista tai kameran tuottamasta kuvasta, josta tulkitaan aallonpituuksien erojen muodostamia kaavamaisuuksia.

Koska sensorijärjestelmät kykenevät tulkitsemaan elementtejä esimerkiksi tutkasignaaleista niihin kehitettyjen säännöstöjen perusteella, korostaen kiinnostavia elementtejä ja häivyttäen näiden ympäristöä, sekä seuraamaan näitä signaalilähteitä niin sanottuina maaleina erilaisin algoritmein, muodostavat ne automaattisia järjestelmäkokonaisuuksia, jotka esittävät yleensä ihmisoperaattorille prosessoidun näkemyksen vallitsevasta tilanteesta. [101 s. 76, 214] Tekoälyn menetelmin ei voida vaikuttaa olemassa olevan sensorin mekaaniseen kykyyn tuottaa signaalipulssi tai sen herkkyyteen havaita hyötysignaaleja. Sen sijaan tekoälyn menetelmin voidaan luoda tehokkaampia sekä uusia tapoja sensorijärjestelmän operointiin, datan prosessointiin ja tulkitsemiseen sekä eri sensorilähteiden tuottaman datan fuusiointiin, joka tuottaa maalitiedolle lisäarvoa laajemman piirreavaruuden muodossa.

TAULUKKO 3: Yleiset sensorijärjestelmät ja niiden yleiset ulostuloformaatit [102 s. 273-281]

Sensori	Ulostulon formaatti
Infrapuna	Kuva
Laser ja LIDAR	Pistepilvi
Kamera	Kuva
Optinen	Kuva
Tutka	Strukturoitu aikasarja
Sonar	Strukturoitu aikasarja
Spektroskopia	Taajuusspektri

Taulukossa 3 on katsaus yleisimpien sensorijärjestelmien tuottamista dataformaateista. Seuraavissa alaluvuissa käsitellään kyseisiä dataformaatteja ja niihin soveltuvia tekoälymenetelmiä sekä uusia innovaatioita.

3.2.1 Passiiviset sensorit: kamerat, EO, ELTU ja hydrofonit

Aiemmin kuvatusti tekoälyn menetelmät sopivat erityisen hyvin kaavamaisuuksien havaitsemiseen ja luokitteluun. Kun kyseessä on sensoridata ja kiinnostavien elementtien havaitseminen kyseisestä datasta, on tekoälyongelma lähtökohtaisesti piirreavaruuksien luominen ja erottelu erilaisten elementtien kaavamaisuuksien välillä. Passiiviset sensorit, jotka eivät lähetä omaa hyötysignaalia vaan pelkästään vastaanottavat jonkinlaista aaltoliikettä, usein piirtävät havaitsemansa aaltoliikkeen kuvaksi sensoria käyttävälle operaattorille. Kamera ja EO-järjestelmien kanssa tämä on itsestään selvää, sillä sähkömagneettisen spektrin näkyvän valon ja infrapunan aallonpituuksilla muodostuu sensorille suoraan kuvamateriaalia. Näin ollen piirreavaruus on kaksi- tai kolmiulotteinen matriisi, jossa on vähintään satoja, usein tuhansia pikseleitä. Kuva muodostaa laajan piirreavaruuden, josta kaavamaisuudet samanlaisten objektien välillä pyritään havaitsemaan.

Elektronisen tuen ympäristössä muodostetaan käsitys vallitsevasta, ympäröivästä sähkömagneettisesta säteilystä, kuten tutka- ja radiosignaaleista, jollain tarkasteltavalla taajuusvälillä. Kerätty data voidaan esittää esimerkiksi levittämällä ympäröivät havaintosuunnat X-akseliksi, kuvaten Y-akselilla havaittuja taajuuksia ja näiden voimakkuuksia pikselin kirkkaudella. Jokainen ajanhetki piirtyy tarvittaessa havaintopisteen suhteen ympärisektorin olevana kuvana. Kuitenkaan kyseisestä kuvasta ei ole saatavissa irti yksittäisten säteilijöiden parametreja, vaan nämä attribuutit tulee tarkastella erikseen. Vastaavalla tavalla voidaan analysoida hydrofonien havaintoja, esimerkiksi muodostamalla kuvankaltainen datapiste akustisen havainnon ominaisuuksista kuten aaltomuodosta, Mel-taajuudesta tai spektristä ja hyödyntää näin saatuun dataan kuvan luokitteluun soveltuvaa menetelmää kuten SVM tai CNN [103]. Vedenalaisten äänilähteiden tunnistamisessa syväoppimisen menetelmät vaikuttavat olevan pääsääntöisesti perinteisiä menetelmiä parempia [104]. Elektronisen tiedustelun kannalta radiosignaalien havaitseminen ja luokittelu on myös mahdollista tekoälyn menetelmin, tutkimusten osoittaessa koneoppimismenetelmien kuten SVM sekä erilaisten neuroverkkojen olevan toimivia ratkaisuja näihin tehtäviin. [105; 106]

Kuvia tuottavien sensorien, kuten tavallisten ja infrapunakameroiden, dataan soveltuvien tekoälymenetelmien määrittäminen on yksiselitteistä, sillä kuvien tunnistukseen ja luokitteluun on kehitetty huomattava määrä työkaluja, kuten aiemmin esitellyt CNN-verkot ja näiden mahdollistama konenäkö. Tämän opinnäytetyön liitteessä 6 on toteutettu koneoppimisprojekti, jossa on vertailtu erilaisten tekoälymenetelmien soveltuvuutta venäläisten ja suomalaisten sota-alusluokkien keskinäiseen luokitteluun. Jo vaatimattomalla data-aineistolla, hyödyntäen CNN-rakenteita ja siirto-oppimista, saadaan tuotettua kohtuullisia luokittelijoita.

Toinen passiivisiin, aikasarjaisiin sensorihavaintoihin kuten hydrofoneihin ja elektronisen tuen sensoreihin soveltuva tekoälymenetelmä on takaisinkytketty neuroverkko. Esimerkiksi pidemmän aikavälin riippuvuuksien havaitsemisessa erinomaiseksi osoittautunut LSTM (*Long-Short Term Memory*) algoritmi [36 s. 397-400] voisi tuottaa anomaliahavaintoja passiivisten sensorien datavirrasta, indikoiden esimerkiksi elektromagneettisen spektrin poikkeuksellista toimintaa tai vedenalaisen akustisen ympäristön poikkeavaa aktiivisuutta. Tällöin ei tunnisteta yksittäisen tilannekuvaelementin luokkaa tai tunnistetta, mutta tuotetaan lisää piirteitä taustaan, jota vasten tilannekuvaelementtejä tulkitaan. Aktiivisuuden muutokset suuntaan tai toiseen antavat indikaation anomaliasta, joka voi

laajemmassa analyysissa tuottaa lisäarvoa analyysille tai toimia indikaattorina tietyn tapahtumaketjun alkamisesta tai päättymisestä.

Muiden passiivisten sensorien datan tulkinnassa myös klassiset tekoälymenetelmät (GOFAI, ”*goold old fashioned AI*”, symbolinen tekoäly) ovat oletettavasti soveltuvia, sillä havaintojen verrattain suppea piirreavaruus mahdollistaa kirjoitetun säännöstön, joka voidaan tuottaa propositiologiikan muodossa osaksi järjestelmää. Kuitenkin symbolisen tekoälyn menetelmät edellyttäisivät huomattavaa määrää asiantuntijatyötä verrattuna koneoppimisen käyttämiseen. Esimerkiksi elektronisen tukemisen tapauksessa passiivisen sensorin tulee havaita signaalista sen modulaatio, sivukeilat, pulssin pituus ja pulssintoistotaajuus (PRF*, pulse repetition frequency*) [101 s. 84]. Näin muodostuu neliulotteinen piirreavaruus, joka liittyy jonkin laitteen havaittuihin parametreihin ja saa siten annotaation. Perinteisillä ohjatun oppimisen menetelmillä voidaan liittää kyseiset parametrit niiden annotaatioon esimerkiksi päätöspuuoppimisella tai multinomilla logistisella regressiolla. Myös ohjaamattomat koneoppimismenetelmät soveltuvat vähintään piirreavaruuksien ominaisuuksien hahmottamiseen, mahdollisesti myös suoraan luokittamiseen, esimerkiksi K-nearest neighbours algoritmilla rypästettynä. Koska elektromagneettisen spektrin dataa kerätään jatkuvasti, on olemassa kattavia data-aineistoja, joilla on myös suurilta osin vahvistetut annotaatiot. Kyseisen datan ominaisuuksia ei voida kuitenkaan kokeellisesti testata tämän työn puitteissa johtuen datan turvaluokituksista. Pienen piirreavaruuden ongelmissa perinteiset koneoppimismenetelmät tuottavat todennäköisesti hyviä luokittajia, mutta syväoppimismetodit vaikuttavat tarkasteltujen tutkimusten valossa tuottavan tästä huolimatta parhaita tuloksia suurimmassa osassa ongelmatilanteita. Rajoittavana tekijänä syväoppimisen hyödyntämiselle on taltioidun, soveltuvan datan määrä.

3.2.2 Aktiiviset sensorit: tutkat ja kaikumittaimet

Tutkadatan luokittelu on kuvia monimutkaisempi prosessi, sillä raakadata koostuu varsin yksinkertaisesta signaalihavainnosta: mikäli paluusignaali ylittää ilmaisukynnyksen, se esitetään operaattorille, jolloin havainto on tehty. Paluukaiun ainut merkittävä piirre tässä kohtaa on voimakkuus. Yksittäinen havainto eli paluukaiku on piste (”*plot*”), jolla on etäisyys ja suunta tutkan suhteen sekä mahdollisesti korkeus, mikäli käytetty tutka mahdollistaa sen havaitsemisen. Näitä pisteitä yhdistämällä aikasarjaksi muodostuu maalitieto liiketekijöineen, eli maalin liikesuunta ja nopeus edellisestä havaintopisteestä nykyiseen. Käytännössä tutkasignaali voidaan hahmottaa esimerkiksi monivaihetutkissa neliulotteisena datapisteenä, jonka piirreavaruuden dimensioita ovat suunta, etäisyys- ja korkeustieto sekä Doppler. Tällaisella 4D-datalla saadaan koulutettua neuroverkko tunnistamaan erilaisia kohteita suoraan tutkadatan kompleksilukuina esitettyjen piirteiden perusteella [107 s. 1-6]. Yksi vaihtoehto tutkasignaalien piirreavaruuden laajentamiseen on Fourier-muunnos sekä pilkottu Fourier-muunnos, jota hyödyntämällä on saatu tuotettua K-nearest neighbour luokittajalla yli 92% tarkkuus kuuden erilaisen tutkasignaalin välillä, joskin simuloidulla aineistolla [108 s. 583-587]. Kyseessä ei kuitenkaan ole erilaisten tutkasignaalihavaintojen luokittelu, eli toisin sanoen saman lähetinvastaanottimen tuottaman, samanmuotoisen signaalidatan luokittelu. Sen sijaan toisessa tutkimuksessa hyödynnettiin tekoälylle koulutettuina mittauksina yksittäisessä etäisyysportissa esiintyneen kohteen valaisuajan mikro-Doppler spektriä. [109] Mikro-Doppler ilmiöllä voidaan havaita varsinaisen kohteen liikkeen lisäksi kohteelle tyypillisiä piirteitä esimerkiksi siitä, onko sillä oman liikenopeutensa lisäksi muita dynaamisia piirteitä kuten pyöriviä siipiä. Näin signaalianalyysia saadaan vietyä pidemmälle ja pelkästä tutkasignaalista saadaan muodostettua piirreavaruus, jonka luokittelu kategorioihin on mahdollista. [110] Mikro-Doppler ilmiön hyödyntäminen kuitenkin edellyttää sen havaitsemista sensorin vastaanottamasta paluusignaalista suoraan.

Neumannin ja Broschin tutkimuksessa mikro-Doppler spektreillä koulutettu syvä neuroverkko saavutti 95.2% luokittelutarkkuuden kuuden eri luokan välille hyödyntäen huomattavaa data-augmentaatiota [109]. Koska itse sensorijärjestelmien raakadatan prosessoiminen ja hyödyntäminen edellyttää yhteistyötä laitteiston toimittajan kanssa, tulee tekoälyn menetelmien hyödyntämistä mahdollistavan operoinnin olla uusien hankintaprojektien keskiössä. Tämä on erityisen oleellista siksi, että laitteen valmistaja ei operoi laitetta samassa toimintaympäristössä, jolloin hyödylliset mikro-Doppler havainnot kertyvät vasta käyttäjän operoidessa sensoria. Näin ollen järjestelmähankinnan ei tulisi olla avaimet käteen -projekti, jossa valmistaja siirtää laitteiston kokonaisuudessaan käyttäjän ja käyttäjän sopimusorganisaation operoitavaksi ja huollettavaksi, vaan hankintasopimuksissa tulisi huomioida datan kertyminen operoinnin aikana ja tähdätä pitkäaikaiseen yhteistyöhön data-analyysin ja sen myötä signaalinkäsittelyyn kykenevien tekoälymallien kehittämisessä. Mikäli esimerkiksi mikro-Dopplerin analysoinnilla voidaan antaa signaalihavainnolle luokitus kuten ’helikopteri’, ’suihkukone’, ’potkurikone’, kyetään havainnointia tehostamaan ajallisesti sekä muodostamaan parempaa tilannekuvaa mahdollisesti kamerajärjestelmien havaintohorisonttia kauempaa. Aikaisempi luokitus mahdollistaisi pidemmän reagointiajan sekä muun muassa parantaisi mahdollisuuksia hyödyntää kaukovaikutteisia järjestelmiä.

Perinteisessä mallissa tutkamaalien luokittelu perustuu useista havainnoista koostuvan tutkamaalin piirteiden perusteella tehtyyn analyysiin, jolloin maalin ominaisuuksia arvioidaan esimerkiksi nopeuden ja liikesuunnan perusteella kuten aiemmin on demonstroitu tekoälymenetelmiä esiteltäessä. Näin ollen tunnistuslogiikasta voidaan jälleen muodostaa propositiologinen säännöstö etukäteen, kuten joissain taistelunjohtojärjestelmissä on tehtykin. Säännöstön ongelma on kuitenkin muuttumaton, sillä siitä muodostuu joko liian monimutkainen tai tarkkarajainen, jolloin esimerkiksi lentokone, joka lentää metrin alle 10 kilometrin korkeudessa olisi pommittaja, kun taas metrin korkeammalla se olisi reittikone. Tätä toimivampi menetelmä on hyödyntää koneoppimista operatiivisessa ympäristöstä kerättyyn tilannekuvadataan. Yksi vaihtoehto on käyttää sumeaa logiikkaa pehmentämään maalityypeille opittuja raja-arvoja. Sumealla logiikalla maalin kuuluvuutta johonkin luokkaan voidaan arvioida liukuvammin jokaisen piirteen kohdalla erikseen, jolloin päätösraja kuuluvuudesta tiettyyn luokkaan ei ole täysin tarkkarajainen. S. P. Noyesin tutkimuksessa esitettiin sumean logiikan säännöstö tutkamaalien luokittelemiseen esimerkiksi ilmaaluksiin, ohjuksiin, aluksiin, kulkuneuvoihin ja lintuihin pääasiassa niiden nopeuden ja signaalin voimakkuuden perusteella. Tutkimuksessa muodostetaan parametreista jatkuvat muuttujat osallisuudelle johonkin luokkaan, eli ”sumeutetaan” mitatut piirteet, jonka jälkeen eri parametrien luokkaan kuuluvuuden kombinaatioilla päädytään todennäköisimpään tulokseen, joka ilmaistaan diskreetisti. [111] Tässä työssä demonstroitu logistinen regressio muodostaa jatkuvan arvon eri piirteiden kuulumisesta johonkin luokkaan, samalla tavalla sumeuttaen tunnistuslogiikan tarkkarajaisuutta, mutta tuottaen diskreetin luokituksen.

Vedenalainen akustiikka ja siihen pohjautuvat aktiiviset kaikumittaimet tuottavat huomattavasti elektromagneettisen spektrin tutka-aalloista poikkeavaa signaalidataa. Kyseisen datan analysointiin ja havaintojen luokitteluun on toteutettu tätä työtä varten käytännöllistä testausta hyödyntäen erilaisia tekoälymenetelmiä. Testi on kuvattu tuloksineen liitteessä 3.

3.2.3 Sensorifuusio

Sensorifuusio tarkoittaa eri lähteistä tulevan datan yhdistämistä siten, että yhdistämällä muodostettu data on jollain tapaa parempaa kuin yksittäisten lähteiden data yksinään. Sensorifuusion saavutuksia ovat muun muassa jykevyys,

laajennettu spatiaalinen ja ajallinen kattavuus, korkeampi luotettavuus ja vähentynyt epävarmuus sekä parannettu erottelukyky. [112 s. i] Yleinen esitys sensorifuusion kategorisoimiseksi on jakaa fuusio kolmeen tasoon taulukon mukaisesti.

TAULUKKO 4: Sensorifuusion kategoriat [112 s. 14]

Fuusion taso	Toiminto
Matala: raakadatan fuusio	Yhdistää useiden lähteiden raakadataa uuden datan tuottamiseksi
Keskitaso: piirretason fuusio	Yhdistää useita piirteitä piirrekartoiksi, joita voidaan käyttää segmentointiin ja havaitsemiseen
Korkea: päätöksenteon fuusio	Yhdistää useiden solmujen päätökset esimerkiksi sumealla logiikalla tai tilastollisesti

Taulukon 5 kategorioissa fuusio luokitellaan sen mukaan, minkälaista dataa fuusioon käytetään. Raakadatan fuusio on laskennallisesti raskain, sillä käsittelemätön data on piirreavaruudeltaan suuri ja kompleksinen yhdistettävä. Keskitason fuusiossa voidaan yhdistää esimerkiksi kahden eri tutkan, valvonta- ja tulenjohtoseuraimen maali samaksi maaliksi näiden tulkittujen piirreavaruuksien osalta, sillä molemmilla maalitiedoilla on olemassa spatiaalinen tila tietyllä tarkkuudella ja sen muutoksesta mitattu nopeus ja korkeus. Päätöksenteon fuusiossa yhdistetään useilta solmuilta tai asiantuntijoilta tulevia päätöksiä yhden konsensuksen muodostamiseksi. [112 s. 14-15]

Viime vuosina monisensorinen fuusio on saanut korostetusti huomiota niin siviili- kuin sotilassovelluksissa. Sensorifuusio hyödyntää tekniikoita yhdistää usean sensorin dataa ja niihin liittyvää, relevanttia dataa tietokannoista saavuttaakseen parannetun tarkkuuden ja spesifimmin päättelyn kohteesta kuin yhdellä sensorilla on mahdollista. Esimerkiksi siinä, missä tutka on erinomainen sensori ilmaisemaan havainnon etäisyys, sen kyky tuottaa suuntatieto on rajoittuneempi, kun taas infrapunakamera kykenee tuottamaan erittäin tarkan suuntatiedon, muttei etäisyystietoa: sensorifuusiolla kahden sensorin välillä kyetään tuottamaan erittäin tarkka suunta- ja etäisyystieto maalista.[113 s. 537-540] Edellä kuvatusti tutka ei myöskään kykene luokittelemaan kyseistä havaintoa, ellei sitten paluukaiun koon ja voimakkuuden, maalipiirteiden tai mikro-Dopplerin perusteella, jolloin havainnon luokituksen vahvistamiseksi sensorifuusio on oleellinen keino laajentaa maalin piirreavaruutta ja tuottaa entistä parempia havaintoja.

Merellisen ympäristön sensorifuusiota on tutkittu kattavasti muun muassa Turun yliopistossa liittyen Rolls-Roycen autonomisten alusten kehitystyöhön. Yksi ehdotettu viitekehys suorittaa selektiivisenä hakuna mahdollisten kohteiden tunnistamisen RGB-kamerakuvasta, fuusioi datan LiDAR (*light detection and ranging*), IR-kameran, ja merenkulkututkan datan kanssa ja syöttää lopulliset ehdotukset CNN-arkkitehtuurille kohteiden tunnistamiseksi saavuttaen parhaimmillaan 88.8%, 100% ja 99% tarkkuuden luokille merimerkki, vene ja maa. Häiriöllisemmän datan ja LiDAR:n heijastusten tuottamien haamumaalien vaikeuttamassa luokittelussa viitekehyksen ehdottama malli saavutti vastaavissa luokissa 94.7%, 69.9% ja 74.2% tarkkuuden. [114] Merellisen ympäristön sensorifuusion hyödyntäminen olisi todennäköisesti hyödyllistä, paitsi alusten ja merivalvonta-asemien meritilannekuvan luomisessa, myös laivaston alusten ohjaamoryhmien työskentelyn tukemisessa. Hyödynnettävyyden mahdollistamiseksi aluksien valmiuksia tulisi parantaa hyödyntämällä nykyistä kamerakalustoa ja täydentämällä sitä lisäsensoreilla datan keräämiseksi. Kaupallinen ja siviilipuolen kehitystyö on niin pitkällä ja näkökulmaltaan yhtenevä

sotilasmerenkulun päämäärien kanssa, että Merivoimien kannalta olisi todennäköisesti kustannustehokkainta solmia kehitysyhteistyösopimuksia autonomisen sotilasmerenkulun tutkimiseksi ja kehittämiseksi.

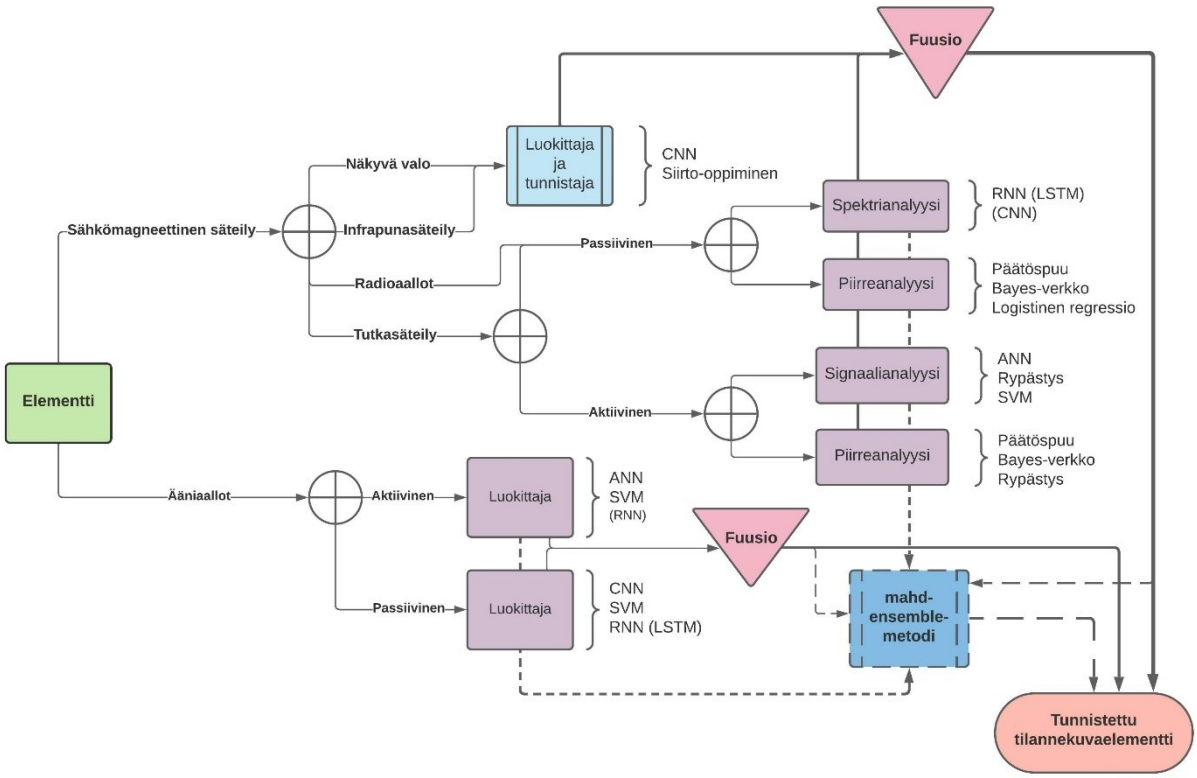
Edellä kuvatusti tekoälyn mahdollistama sensorifuusio kykenee yhdistämään eri sensorien havaintoja, muodostaen yhdistetyn piirreavaruuden ja sitä kautta mahdollisesti varmemman ja tarkemman luokituksen tai tunnistuksen kohteelle. Sensorifuusion prediktiivisen menestyksen lisäksi sensorifuusio saattaa auttaa välttämään aiemmin esitettyä vastakkaisten esimerkkien ongelmaa esimerkiksi kuvantunnistukseen liittyen. Sensorifuusion tapauksessa vastakkaisten esimerkkien tuottaminen korruptoi esimerkiksi yhden fuusioitavan piirrevektorin arvon, muttei muiden sensorien tuottamia piirrevektoreita. Tutkimusten valossa vastakkaiset esimerkit kykenevät estämään eri sensorilähteiden fuusiolla koulutetun mallin toiminnan tehokkaasti vain yhtä lähdettä häiritsemällä ainakin siinä tapauksessa, että kyseessä on samantyylinen data, kuten lämpökamerakuva ja RBG-kuva. Kuitenkin yhdistettäessä RGB-kuvaa ja kuvatun tilan syvyystietoa, saadaan huomattavasti luotettavampi malli vastakkaisista esimerkeistä huolimatta. [115]

Sensorifuusion ohella tekoälymenetelmiä voidaan käyttää niin sanotussa ”*ensemble learning*” mallissa, jossa mahdollisten ratkaisujen hypoteesiavaruudesta valitaan yhden mallin sijaan hypoteesien joukko (”*ensemble*”), jonka tuottamat prediktiot yhdistetään, esimerkiksi siten, että kappaleessa 2 kuvattujen, samalla datalla koulutettujen mallien tuottamat prediktiot lasketaan ääniksi tietyn luokituksen puolesta ja eniten tällaisia ”ääniä” saanut luokitus voittaa. Näin ollen väärinluokituksen riskin toivotaan pienenevän huomattavasti. [31 s. 748] Tässä mielessä ”*ensemble*” metodi vastaa korkean tason sensorifuusiota, jonka syötteet tulevat asiantuntijoiden sijaan raakadataa tai piirteitä käsitteleviltä älyiltä.

3.2.4 Yhteenveto ja konsepti

Havaittavien elementtien säteilemä tai heijastama aaltoliike jakautuu tilannekuvan tapauksessa sähkömagneettiseen säteilyyn ja ääniaaltoihin. Näistä ääniaalloilla viitataan nimenomaiseen vedenalaiseen akustiikkaan, kun taas sähkömagneettinen säteily pitää sisällään näkyvän valon, infrapunasäteilyn, radioaallot ja tutkasäteilyn. Piirreavaruudet edellä käsitellyissä sensorikategorioissa jakautuvat karkeasti aikasarjoihin ja läpileikkauksiin. Aikasarjoihin kuuluvat hydrofonien ja elektronisen tuen sensorien tallentama data, kun taas läpileikkauksia ovat tietyn ajanhetken havainnot, jotka eivät riipu muista ajanhetkistä. Myös useammasta havaintopisteestä tulkitut tutkamaalien piirreavaruudet lasketaan näihin, sillä maalin suunta- ja nopeustietojen määrittäminen vaatii vähimmillään vain nykyisen ja sitä edeltävän mittauksen, jolloin aikasarja on hyvin kapea.

Alaluvun havainnoista on luotu systeemikaavio, joka on esitetty kuvassa 19.



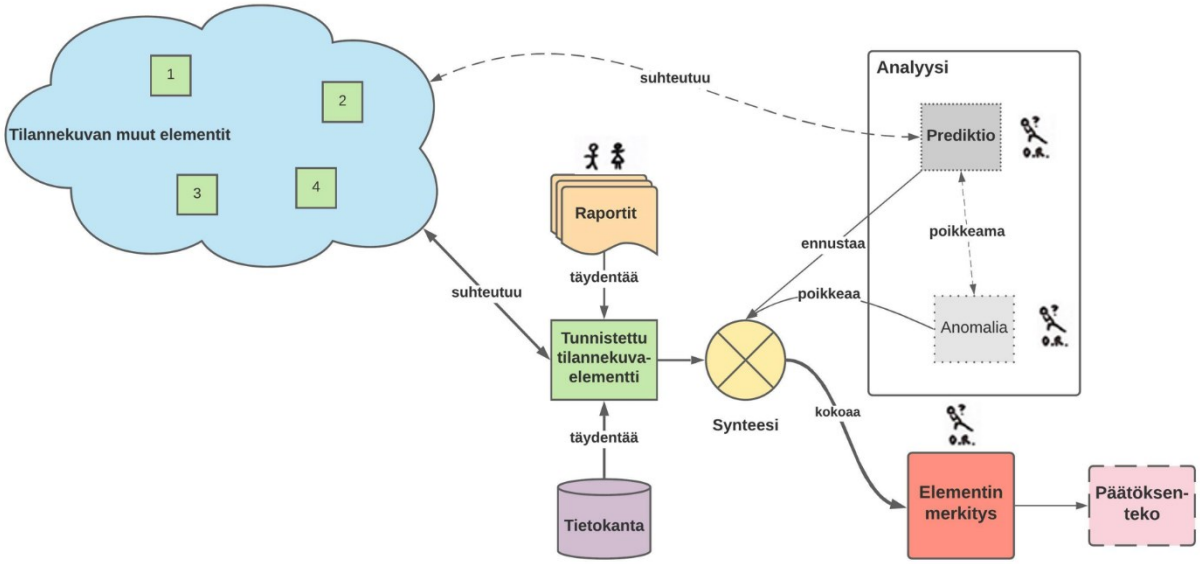
KUVA 19: Osaongelmien ratkaisuihin arvioidut tekoälymenetelmät

Havainnoinnin tehostamisen konseptissa läpileikkauksista koostuviin piirreavaruuksiin soveltuviksi menetelmiksi mielletään edellä mainitusti pääpainoisesti CNN ja siihen liittyvä siirto-oppiminen sekä SVM. Myös aikasarjaisessa datassa, josta kyetään tuottamaan läpileikkauskuva, kuten vedenalaisen äänen luokittelussa, toimivat lähtökohtaisesti samat menetelmät, kuten liitteessä 3 on arvioitu. Takaisinkytketyt neuroverkot, mukaan lukien LSTM, toimivat oletettavasti aikasarjaisen datan anomaliatarkastelussa, jossa kiinnostavat muutokset voivat tapahtua lyhyellä tai pitkällä aikajänteellä. Kapeiden piirreavaruuksien data, kuten ELTU ja aktiivisen tutkan maalien piirreavaruudet, on hypoteettisesti helpoin mallintaa vastaamaan nykyisten operaattorien tapaa luokitella havaintoja kouluttamalla luokittelumalleja perinteisin koneoppimismenetelmin. Oletettavasti niin ELTU kuin valvontatutka-operaattorien työskentelyn mallintamisessa päätöspuuoppiminen toimisi helpoiten validoitavana menetelmänä tuottaa operaattorin työskentelylogiikkaa vastaava äly tukemaan tai suorittamaan operaattorin toimenpiteitä.

Liitteiden 3 ja 6 tarkastelut vahvistavat osaltaan konseptin aktiivisen vedenalaisen äänen, valokuvien sekä IR-kuvien tulkinnan ja luokittelun osalta. Koska dataa muista tunnistetuista ongelma-alueista ei ollut saatavilla tämän työn puitteissa, on konseptin hypoteesien testaaminen toteutettava jatkotutkimuksena.

3.3 Orientaatio

Havainnointi tuottaa sensoridatasta tunnistetun tilannekuvaelementin tulkittavaksi seuraavalle prosessivaiheelle, orientaatiolle. Endsleyn näkemystä [98] mukaillen tässä työssä orientaatio on elementin analyysi, jossa syntyy ymmärrys elementin tarkoituksesta ja projektio elementin statuksesta lähitulevaisuudessa, sekä synteesinä elementin yhteys muihin elementteihin sekä näiden lähitulevaisuuden projektioihin. Aiemmin todetusti tekoäly on luonteeltaan prediktiivistä eli ennustavaa, joten tekoälyn termein kyseessä on prediktio elementin statuksesta ja toiminnasta. Olemassa olevan tiedon pohjalta luodaan malleja tunnistamaan ja luokittelemaan uusia, mallille tuntemattomia datapisteitä. Näin ollen tekoäly on määritelmällisesti erinomainen työkalu analyysiin ja esimerkiksi pörsseissä käytetty ennakoimaan osakemarkkinan liikkeitä ja hintakehitystä. Sotilaallisessa käyttötarkoituksessa voidaan tekoälyä käyttää vähintään kolmella tavalla, prediktioiden tekemiseen ja anomalioiden havaitsemiseen sekä epävarmuuksia sisältävien päättelyketjujen luomiseen. Havainnoinnissa anomaliat ovat lähtökohtaisesti kiinnostavia elementtejä, kun taas analyysissa anomalia on havaitun elementin poikkeama normaalista tai sille ennustetusta tilannekehityksestä. Lisäksi esimerkiksi bayesilaisella päättelyllä kyetään luomaan epävarmuuksia sietävä yhteys muuhun saatavilla olevaan dataan, kuten uhka-arvioihin ja tiedustelutietoon.



KUVA 20: Orientaation prosessi ja ongelmatilanteen visualisointi

Kuvassa 20 on visualisoitu orientaation ongelmatilanne siten kuin se tässä työssä hahmotetaan. Ihmissymbolit kuvaavat pehmeää dataa ja tällä hetkellä ihmisten toteuttamia osaprosesseja. Kuva toimii samalla sekä rikkaana visualisointina että systeemikuvana. Oleellisia systeemejä on erotettavissa kaksi, synteesin systeemi sekä analyysin systeemi. Synteesissä yhdistetään tunnistettu tilannekuvaelementti muuhun saatavissa olevaan tietoon, esimerkiksi tiedustelutietoon ja havaintoon liittyviin raportteihin. Validoivia ja täydentäviä raportteja voivat olla esimerkiksi muiden toimijoiden havainnot, jotka tukevat kyseisen havainnon luokittelua ja tunnistusta.

Analyysi tuottaa ennusteen tilannekuvaelementin lähitulevaisuuden statuksesta. Samalla analyysi vertaa sekä nykyistä että ennustettua statusta luokkakohtaiseen normaaliin. Mikäli jompikumpi tai molemmat statukset poikkeavat normaalista, on kyseessä anomalia. Statuksen anomalia lisää havaitun elementin todennäköisyyttä olla merkit-

tävä, sekä saattaa muokata koko havaintoympäristöä johonkin suuntaan. Esimerkiksi havaittaessa, että Suomenlahdella kulkee ulkovaltion sota-alus, jonka status on pääosin normaali: nopeus, sijainti, etäisyys muihin aluksiin tai mikään muukaan attribuutti ei ole poikkeava, paitsi kulkusuunta. Kulkusuunnan poikkeama on pieni ja tilastollisesti merkityksetön, mutta ennakoitu status 15 minuuttia tulevaisuuteen osoittaa sota-aluksen olevan tuolloin, mikäli attribuutit pysyvät samoina, Suomen aluevesillä. Tällöin kyseessä on anomalia, poikkeustilanne. Tilanne on myös tilastollinen poikkeama, sillä tilannekuva-aineistossa ulkomaiset sota-alukset pysyvät pääsääntöisesti aluevesien ulkopuolella ja niiden paikkaan ja aikaan sidotut liiketekijät eivät vastaa kyseisen havainnon attribuutteja samassa tilassa.

CATWOE tarkastelussa kuvan 20 systeemin omistaja ei näy kuvassa, vaan orientaation systeemi kuuluu mille tahansa Merivoimien toimijalle, joka kokoaa, esittää ja analysoi tilannekuvaelementtejä, kuten merioperaatiokeskukselle tai aluksen taistelukeskukselle. Tällöin systeemin omistaja on suoraan ylempi systeemi kuten Merivoimien esikunta tai taistelualuksen päällikkö, joka tuottaa systeemille tehtävän ja tavoitteen. Omistaja on samalla yksi systeemin asiakkaista. Kyseisen systeemin syöte on havainnoinnin systeemistä tuleva tunnistettu tilannekuvaelementti, jonka parissa toimivat analyytikot ja operaattorit pyrkivät suhteuttamaan muihin havaintoihin sekä vallitsevaan tilanteeseen ja tietopohjaan. Analyytikot, operaattorit ja tilannekuvajärjestelmät kokoavat, esittävät ja ennustavat kyseisen tilannekuvaelementin statusta, toteuttaen sekä synteesin että analyysin tehtävät. Systeemin asiakkaat eli hyötyjät löytyvät kolmesta suunnasta; sekä havaittu elementti, sen analyysi, että siitä luotu synteesi välitetään tietona niin ylös-, kuin alaspäin organisaatiossa, sekä samassa tasossa esimerkiksi taistelualusten välillä tai operaatiokeskusten välillä. Tekoälyn tuomat muutosmahdollisuudet liittyvät oleellisten alasysteemien, synteesin ja analyysin kuvan 20 mukaiseen toimintaan: anomaliahavaintojen tekemiseen, prediktiiviseen analyysiin sekä tiedon yhdistämiseen erilaisina prioreina oletettuun, epävarmaan posterioriin. Systeemin rajoitteita ovat synteesin laatu ja analyysin tarkkuus. Synteesin laatuun vaikuttaa ennen kaikkea tiedon saatavuus ja sen integroitavuus kuhunkin tilannekuvaelementtiin. Abstraktimpi tieto, kuten kiristynyt poliittinen tilanne, ei näy normaalissa maalitilannekuvassa tilannekuvaelementtien attribuutteina, vaan operaattorin ja systeemin päätöksentekijän prioritietona. Tämä vaikuttaa siihen, millä tavalla he tilannekuvaelementtejä tulkitsevat esimerkiksi uhkaavuuden osalta. Operaattorien ja analyytikoiden koulutuksella ja kokemuksella on merkittävä vaikutus systeemin toimintaa ja sen toteuttaman prosessin laatuun. Esimerkiksi sotilaskoneiden siirtolento rauhan aikana herättää normaalin seuraamisen mielenkiinnon, mutta tiedon yhdistäminen aiempaan alueloukkauksesta tai poliittisen tilanteen kiristymiseen muuttaa mielenkiinnon laatua ja intensiteettiä. Toimintaympäristön rajoite orientaation mallintamiselle tekoälyn menetelmin piilee oletettavasti siinä, ettei näiden priorien tuottaminen osaksi analyysijärjestelmää ole nykyisellään mahdollista, sillä kyseistä dataa ei ole taltioitu hyödynnettävään muotoon. Lisäksi kyseessä ei ole datanlouhinnan näkökulmasta rajattu systeemi kuten yhden tai useamman sensorin luoma datamassa tietystä sensorihavaintojen avaruudesta, vaan kompleksinen asiantuntijajärjestelmä, joka käsittelee useita erityyppisiä syötteitä useiden erilaisten toimijoiden työpanoksena. Siinä missä sensorihavaintojen datapisteet voidaan lähtökohtaisesti litistää yksiulotteiseksi piirrevektoriksi, ovat orientaatiossa käytettävät aineistot monimuotoisia syy-seuraussuhteita, totuusarvoja ja poikkeama-analyyseja.

3.3.1 Analyysi

Havaitun elementin merkitys riippuu sen aikomuksen tulkinnasta. Aikomus on tässä työssä kohteen kyvykkyyksien mahdollistama, havainnon tilaan perustuva todennäköisin toimintatapa ja samalla oletus kohteen tulevasta

tilasta. Sotilaallisessa kontekstissa tulkittu aikomus määrittää siihen kohdistuvat toimenpidevaihtoehdot. Synteesin rikastama elementin piirreavaruus yhdistettynä analyysin tuottamaan prediktioon tilannekuvaelementin ja mahdollisesti siihen liittyvän muun joukon aikomuksesta tuottaa lähtökohdan päätöksenteolle.

Merellisessä ympäristössä tekoälyä on käytetty muun muassa ennustamaan alusten liikkeitä Itämerellä saavuttaen alle 2 kilometrin tarkkuus alusten sijainnin ennustamisessa useita tunteja tulevaisuuteen. Kyseinen malli koulutettiin rypästämällä k-nearest neighbours menetelmällä AIS-historiatiedoista tehtävään soveltuva kapea tekoäly. [116 s. 304-309] Toisessa tutkimuksessa rypästäminen yhdistettiin neuroverkkoon, jolle koulutettiin dataa Jangtse-joen vesiliikenteestä. Tuloksena on malli, joka ennustaa alusten aikomuksen valita yksi kolmesta kulkureitistä keskimäärin yli 70% tarkkuudella. [117 s. 1-6] Vastaavien mallien kouluttaminen Merivoimien ja laivaston taltioimasta tilannekuvadatasta on paitsi mahdollista, myös oleellista, sillä sota-alukset eivät käytä AIS-lähetystä. Näin ollen kyseisten alusten liikkeet eivät sisälly muualta saatavissa olevaan koulutusdataan. Toiseksi sota-aluksia on kauppa-aluksiin verrattuna erittäin vähän. Tästä syystä havaintojen määrä on pieni, ja mielenkiintoisimpia havaintoja näiden liikkeissä on erittäin vähän, sillä tavanomainen siirtoliikenne Itämerellä ei poikkea profiililtaan erityisesti tavallisesta reittiliikenteestä. Luotaessa malli, joka ennakoi myös sota-alusten liikkeitä ja sijainteja useita tunteja tulevaisuuteen, kyettäisiin arvioimaan myös tämän prediktion toteutumasta anomalia eli poikkeama normaaliin. Mikäli anomalia on merkittävä, voidaan olettaa sota-aluksen tehneen tai olevan tekemässä jotain tavanomaisesta poikkeavaa, jolloin sen kiinnostavuus maalina kasvaa. Toinen hyöty meriliikenteen sijaintien ennakoimisella tunteja etukäteen liittyy meriliikenteen suojaamiseen ja tulenkäyttöön. Jotta laivasto ja Merivoimat voivat toteuttaa näitä ydintehtäviään, on päätöksenteon kannalta tärkeää kyetä arvioimaan suojattavan liikenteen status mahdollisimman pitkälle tulevaisuuteen parhaan toimintavaihtoehdon määrittämiseksi.

Edellä kuvatussa tilanteessa aluksen aikomus vaikuttaa olevan alueloukkaus, joka edellyttää tiettyjä vastatoimenpiteitä. Kyseiset toimenpiteet etenevät suoraviivaisesti tilannekohtaisen toimintamallin eli algoritmin mukaisesti, kunhan aikomus on tulkittu ja toimintavaihtoehto on valittu. Mikäli synteesi liittää kyseiseen havaintoon priorin siitä, että yleinen aktiivisuus on normaalia korkeammalla ja on olemassa ajallisesti kohonnut todennäköisyys provokaatiolle, muodostuu analyysin syötteestä synteesissä päätöksenteon kannalta luotettava posteriori. Havaittu anomalia eli alueloukkaus sen sijaan ei edellytä toimenpiteitä, jos synteesissä tilannekuvaelementtiin liitetään esimerkiksi raportin tai tietokannan tieto siitä, että kyseisellä alusyksiköllä on lupa tulla aluevesille. Tällöin anomaliatarkastelun merkittävyys nollataan synteesin tuottamalla lisäinformaatiolla. Tämän kaltainen, eristetty tapahtuma voidaan myös tulkita liittyvän isompaan kontekstiin ja toimivan priorina jollekin strategisen tasan posteriorille.

Yhteenvetona voidaan todeta, että analyysi tuottaa prediktioita tilannekuvaelementeille. Prediktioit ja niihin liittyvät anomaliat muodostavat yhden tekoälyavusteisen lisäsyötteen synteesille.

3.3.2 Synteesi

Synteesissä yhdistetään dataa syötteenä saatuun tilannekuvaelementtiin. Itse tilannekuvaelementin datapiste sisältää jo useanlaisia attribuutteja riippuen siitä, minkälainen elementti se on. Tässä käsittelyssä standardielementti on maalitieto, koska dimensiosta riippumatta maaleilla on lähtökohtaisesti vähintään kolme attribuuttia: sijainti, liikenopeus sekä nopeuteen liittyen kulkusuunta. Lisäksi maali kuuluu johonkin dimensioon kuten vedenalaisiin,

veden päällisiin eli pintamaaleihin tai ilmamaaleihin, ja kyseisessä dimensiossa johonkin luokkaan kuten sota-aluksiin ja luokassaan mahdollisesti johonkin alaluokkaan eli tunnisteeseen. Näin ollen tunnistettu tilannekuvaelementti on jo informatiivinen datapiste esitettäväksi tilannekuvassa. Esitetyltä elementiltä puuttuu kuitenkin merkitys, joka on oleellinen seuraavalle vaiheelle eli päätöksenteolle. Synteesi täydentää elementin piirreavaruutta liittämällä siihen erilaista dataa ja suhteuttaa elementin sen toimintaympäristön vallitsevaan yleistilanteeseen, muihin tilannekuvaelementteihin, uhkatilanteeseen ja omien joukkojen statukseen.

Osa datasta, joka elementtiin liitetään, opponoi ja validoi havainnoimisvaiheessa saatua tunnistetta. Esimerkiksi jos havainto luokitellaan sukellusveneeksi, voidaan synteesissä havaintoon liittää priorina muun muassa tiedustelutieto siitä, onko kyseisellä alueella sukellusvenettä ja jos, niin millä todennäköisyydellä. Kaavan 1 esittelemän Bayesin teoreeman mukaisesti kyseinen priori vaikuttaa suoraan ehdolliseen posterioriin ja tätä kautta muuttaa elementin merkitystä: mikäli tiedustelutieto tai -raportti tukee todennäköisyyttä, että kyseessä on sukellusvene, on tunnisteen oikeellisuuden todennäköisyys suurempi.

Tällä hetkellä synteesi tehdään asiantuntijatyönä, jossa operaattorit ja analyytikot liittävät tilannekuvassa esitetyt havainnot muuhun saatavalla olevaan tietoon tuottaen tulkinnan vallitsevasta tilanteesta. Jotta tätä työtä voidaan tehostaa tai korvata tekoälyn menetelmin, tulee synteesin käsittelemän tiedon olla yhdistettävissä tilannekuvan esittämään tietoon. Esimerkiksi tiedustelun raportit sisältävät validoivaa dataa, kuten tiedon siitä, että sukellusvenekalusto on liikkeellä, sekä prioritodennäköisyyksiä. Tiedusteluraportti voi todeta jonkun tapahtuman A olevan todennäköinen seuraavan vuorokauden sisään, jolloin kyseisen tapahtuman priori on tietyllä aikavälillä numeerisesti esimerkiksi $P(A) = 90\%$. Tähän prioriin voidaan liittää muita prioreja tai reunatodennäköisyyksiä. Yhdistämällä useita prioreja Bayes-teoreeman ehdollisiksi todennäköisyyksiksi saadaan tuotettua Bayes-verkko.

Bayes-oppiminen ja Bayes-verkkojen luominen ei kuitenkaan ole sotilasympäristössä yksiselitteistä, sillä todennäköisyyksien määrittäminen etenkin vastustajan kaluston ja toiminnan suhteen on hankalaa. Sopivan Bayes-verkon luominen on ylipäätään työläs ja monimutkainen prosessi. Bayes-verkko voidaan tehdä manuaalisesti tai oppia automaattisesti. Verkon rakentaminen koostuu kahdesta osasta, verkkorakenteen muodostamisesta ja verkon solmujen ehdollisten todennäköisyyksien muodostamisesta. Ensimmäinen on huomattavasti jälkimmäistä monimutkaisempi, sillä ehdolliset todennäköisyydet saadaan laskettua automaattisesti suhteellisen helpolla, mikäli käytössä on tilastotietoa, jonka frekvensseillä kyetään laskemaan todennäköisyyksiä. Sen sijaan verkon rakenteen eli solmujen keskinäisten riippuvaisuuksien muodostaminen ei ole yhtä yksinkertainen prosessi. Bayes-verkkojen oppimisen tutkimus on edelleen kehittyvää ja sisältää useita ehdotettuja algoritmeja [32 s. 215, 217]

Jotta Bayes-verkkojen hyödyntäminen tilannekuvaelementin aikomuksen tulkitsemisessa onnistuu, täytyy olla dataa, jonka avulla verkko luodaan. Bayes-oppiminen edellyttää dataa, josta on tulkittavissa tilastollisina osuuksina todennäköisyydet tietyn tapahtuman ilmaantuvuudelle ja tapahtumien keskinäisille suhteille [32 s. 160-161]. Liitteessä 8 on havainnollistettu konkreettisesti Bayes-verkon käytettävyyttä priorien yhdistämisessä havaintoihin konkreettisella esimerkillä, hyödyntäen valmista ohjelmistoa kaavan 5 mukaisten posteriorien laskemiseksi erilaisten priorien totuusarvojen yhdistelmien mukaisesti. Havainnollistavassa esimerkissä maalitietoon yhdistetään tunnisteen lisäksi muun muassa tietoa sen liiketekijöistä, joiden uhkaavuus määrittyy analyysin perusteella. Lisäksi havaintoon liitetään pehmeän datan tulkinta yleistilanteen ennakkoarviosta ja tiedustelutietoa.

Synteesi yhdistää maalitietoon myös muuta dataa kuin ennalta määrättyjä prioreja ja näistä muodostuvia posterioreja. Luotaessa tunnistettua tilannekuvaa sensorihavainnon piirreavaruuden rikastaminen voidaan toteuttaa myös esimerkiksi rypästämällä maali sen muun ympäristön kanssa, kuten kuvassa 19 on kuvattu tarkasteltavan elementin suhteutumisella muihin elementteihin. Ohjaamattomalla oppimisella voidaan kouluttaa malli, joka osaa luokitella maaleja esimerkiksi niiden muodostamien muodostelmien mukaan. Siinä missä kauppameriliikenne ja lentoliikenne eivät normaalisti etene muodostelmissa, sotilasalukset niin ilmassa kuin pinnalla etenevät jonkinlaisissa muodostelmissa. Kohteista voidaan muodostaa ryppäitä niiden keskinäisten etäisyyksien pohjalta sekä laskemalla näissä ryppäissä olevien maalien kulkusuuntien erotusten itseisarvot ja käyttämällä näitä laajentamaan yksittäisten maalien piirreavaruuksia. Mikäli luokittelijaa käytetään diskreetisti, se muodostaa tunnistuksia jollekin tarkoitteelle, sillä esimerkiksi suojausmuodostelma poikkeaa oletettavasti tavanomaisen siirtoliikenteen muodostelmasta. Mikäli tällaista luokittelijaa käytetään muodostamaan jatkuva kuuluvuusarvo, saadaan jälleen priori kuulumisesta johonkin muodostelmaan ja tämä muodostelman tarkoitteeseen. Tämä priori voidaan liittää edellä mainitusti ehdolliseen posterioriin sille, että kyseinen tilannekuvaelementti on osa tiedustelun ennakoimaa tilannekehitystä. Tällainen laajennettu, ympäristön huomioiva piirreavaruus mahdollistaa paremmin sotilaskohteiden ja poikkeavasti käyttäytyvien kohteiden luokittelun sekä niiden merkitysten analysoinnin kokonaisuuksina.

3.3.3 Yhteenveto ja konsepti

Orientaatiovaiheessa kova ISR-data yhdistetään pehmeään dataan, jotta tilannekuvaelementille muodostuu päätöksentekijän näkökulmasta sen tulkittuun aikomukseen tai aikomuksiin perustuva merkitys. Kovan datan fuusiointi havainnointivaiheessa on verrattain suoraviivainen prosessi pehmeään dataan verrattuna. Pehmeän datan etu on muun muassa ilmaisuvoima, sillä vapaamuotoisempi viesti tai raportti kykenee välittämään tietoa joustavammin ja monimuotoisemmin kuin formaali, standardoitu sanoma. Näin ollen abstraktien vaikuttajien ilmaisu sekä syy-seuraussuhteiden esittäminen onnistuvat parhaiten pehmeällä datalla. Tekoälyn on kuitenkin vaikea tulkita eiformaalista pehmeää dataa, sillä formaali, standardimuotoinen data on käsittelyn kannalta yksinkertaisinta hyödyntää. On yksinkertaista linkittää standardimuotoisen sanoman sisällöstä oleelliset tiedot tarkasteltavaan havaintoon, jos sanomamuoto tukee tätä prosessia. Esimerkiksi sukellusvenehavainnon varmuuteen liittyen tiedustelun raportissa voi olla totuusarvo tai todennäköisyys siitä, onko läheisistä tukikohdista lähtenyt sukellusveneitä. Ohjelmistoautomaatio voi poimia tämän totuusarvon talteen parametrina ja yhdistää sen havaintoon, muodostaen paremman luotettavuuden havainnon todenmukaisuudesta, mikäli sanoma on laadittu tukemaan tätä ominaisuutta.

Pehmeän datan vaikeasta hyödynnettävyydestä on todettava, että tekoälyä voidaan käyttää myös luonnollisen kielen prosessointiin ja siten tuottaa pehmeästä datasta helpommin hyödynnettävää muihin tarkoituksiin. Luonnollisen kielen prosessointi (*natural language processing*, NLP) on kattavasti tutkittu ja kehittyvä tekoälyn ala, jonka tarkoitus on kyetä prosessoimaan ja tiivistämään hyödyllistä tietoa tulkinnanvaraisesta ja usein moniselitteisestä luonnollisesta kielestä, josta pehmeä data koostuu. [31 s. 860-861, 865, 867] Yksi tapa tekoälyn hyödyntämiseen orientaation prosessissa on liittää NLP ominaisuuksia informaation käsittelyyn, tuottaen vähimmillään ihmisoperaattoreille mahdollisuuden käsitellä laajoja pehmeän datan aineistoja tehokkaasti ja parhaimmillaan tuottamaan kovaa dataa synteesiä varten pehmeän datan pohjalta automaattisesti.

Konsepti orientaation systeemin parantamiseksi edellyttää käyttökelpoisen data-aineiston keräämistä. Tämä haaste toimivien tekoälysovellusten toteuttamisessa on universaali. Myös Yhdysvaltojen laivaston tärkeimpiä kehitysehdotuksia on yhtenäisten, poikkitoimialallisten tietokantojen muodostaminen, joka edellyttää uusia tiedonkeruuprosesseja ja käyttökelpoiseksi formalisoituja tietokantoja [22 s. 63, 164]. Ilman poikkitoimialallista tietokantaa riittävän laajojen data-aineistojen ja eri toimialojen tuottaman datan yhdistäminen käyttökelpoiseksi datamassaksi ei onnistu. Datan merkitys ja käytettävyys eivät liity pelkästään tekoälyn kouluttamiseen, vaan myös reaaliaikaiseen analyysiin ja synteesiin. Käytettäessä esimerkiksi Bayes-verkkoja posteriorit riippuvat saatavilla olevasta prioridatasta, jolloin tulisi olla mahdollista synkronoida muiden toimialojen tuottamien priorien tila analyysin aikana. Tällaisen tietokannan käytettävyyden problematiikkaa ei käsitellä erikseen tässä työssä.

Olettaen yllä mainitun data-aineistojen muodostamisen olevan tulevaisuudessa mahdollista, ja näistä toteutettavan käyttökelpoisia tekoälysovelluksia prediktioiden ja riippuvuussuhteiden määrittelyyn, kyetään aiemmin esitellysti tuottamaan tekoälyn menetelmin lisäarvoa tilanneymmärryksen saavuttamiseksi. Soveltuviksi menetelmiksi analyysin tuottamiin prediktioihin käyvät tutkimuksen tulkinnan valossa koneoppimisen menetelmät. Synteesissä probabilistinen ajattelu ja Bayes-verkot kykenevät tuomaan poikkitoimialallisen tulkinnan havaitun tilannekuvaelementin merkityksestä taisteluosaston päätöksenteolle nopeammin kuin nykyinen prosessi, jossa toimijat ja data on hajautettu eri toimialoille ja suurelta osin operaatiota toteuttavan taisteluosaston ulkopuolelle.

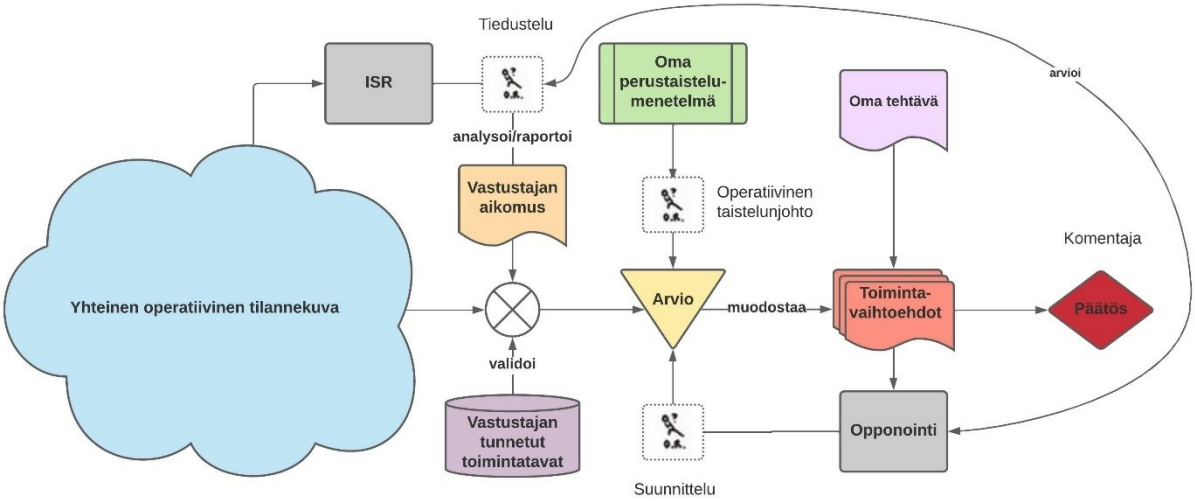
Data-aineistoilla ei tarkoiteta tässä yhteydessä pelkästään kovaa dataa ja sen taltioimista, vaan myös asiantuntijoiden tietopohjan taltioimista ja mallintamista käyttökelpoiseen muotoon. Organisaation sisällä olisi kyettävä tunnistamaan yksittäisten asiantuntijoiden ja asiantuntijaryhmien toteuttamien prosessien rakenteita ja niiden toistuvuutta sekä sitä kautta mallinnettavuutta, jotta asiantuntijoiden kokemus voitaisiin muodostaa formaaliksi asiantuntijajärjestelmäksi tai Bayes-verkoksi.

3.4 Päätöksenteko

Edelliset OODA-silmukan vaiheet ovat kuvanneet yksittäisten tilannekuvaelementtien havaitsemisen ja koostamisen tilannekuvaksi, jonka perusteella on tehty analyysi ja synteesi niiden oletusta tulevasta tilasta sekä aikomuksesta. Näiden tietojen perusteella tulee kyetä muodostamaan päätös siitä, miten havaittuun tilanteeseen reagoidaan.

Sotilaallinen päätöksenteko perustuu oman tehtävän täyttämiseen ja vastustajan toimenpiteiden ennakoimiseen. Yksinkertaisessa tapauksessa havainto ja orientaatio indikoivat tarvetta tietylle taistelutekniselle toimenpiteelle. Tällainen voi olla esimerkiksi havainto tutkalukosta ja kohti tulevasta ohjuksesta, joka edellyttää harhamaaliheitteitä tai muuta torjuntatoimenpidettä uhan välttämiseksi. Päätöksenteon ongelma on tällöin suoraviivainen ja yhdistää edelliset kategoriat suoraan taistelutekniseen toimenpidevalikoimaan eräänlaisena primitiivisenä ärsyke-reaktio-syklinä. Esimerkiksi AEGIS- ja Patriot-järjestelmät suosittavat sensorihavaintojen pohjalta torjuntoja ja mahdollistavat torjunnan suorittamisen myös täysin automaattisesti [43 s. 140-141, 163-164].

Tässä kappaleessa tarkastellaan päätöksentekoa taisteluosaston komentajan tasalla, jossa OODA-kategoria näyttäytyy kompleksisempana ja ominaisuuksiltaan vähemmän automaattisena kuin taistelutekniikka. Päätöksenteko sisältyy sodan- tai taistelun johtamisjärjestelyn eli C2 toimintaan. C2 määritellään yleisesti käsittävän tavoitteen asettamisen, tavoitteeseen johtavien toimintavaihtoehtojen arvioinnin ja niiden toimenpiteiden johtamisen ja valvomisen, jotka johtavat asetettuun tavoitteeseen [22 s. 203]. OODA-silmukan kaksi ensimmäistä vaihetta toteuttavat ISR toimintoja, tuottaen dataa kolmannelle vaiheelle eli päätöksenteolle. C2 toiminnot sijoittuvat OODA-ajattelussa pääasiassa orientaation ja päätöksenteon vaiheisiin, painottuen tässä työssä päätöksentekoon [22 s. 207]. Tekoälyn pääasialliseksi käyttötarkoitukseksi kyseiseen ongelmaan on nähty analyysin helpottaminen eli edellisten vaiheiden tehokkaampi toteuttaminen, jotta analyytikot voivat tehdä toimintavaihtoehtosuosituksia riittävän nopeasti. [22 s. 169]



KUVA 21: Päätöksenteon prosessi, systeemitoimijat ja ongelmatilanteen visualisointi

Kuvassa 21 on yksi käsitys taisteluosaston esikunnan päätöksenteon prosessista ja sitä toteuttavasta systeemistä. Prosessin toteuttavan systeemin muodostavat asiantuntijat, jotka toteuttavat prosessivaiheet. Kyseisen systeemin omistaja on taisteluosaston komentaja. Komentajan päätöksenteko pohjautuu lähtökohtaisesti hänen kokemus- ja

koulutustaustaansa ja tavalle, jolla hän näiden pohjalta käsittelee itselleen esitettyä dataa tilanneymmärryksestä ja toimintavaihtoehdoista. Systeemin asiakkaita ovat komentajan joukot, jotka odottavat tuotetta eli käskyä, joka sisältää komentajan tahtotilan ja päätöksen sekä arvion vastustajan toiminnasta. Systeemin toimijat muodostuvat komentajan esikunnasta. Kuvassa on visualisoitu kolme toimialaa, jotka tuottavat komentajalle arvion toimintavaihtoehdoista. Systeemin välituotteita ovat toimenpidevaihtoehdot, joista komentajan tulee valita vallitsevan käsityksen valossa optimaalisin. Päätuote on komentajan päätös toimintatavasta ylemmän tason kokonaisuutta tukevan lopputuloksen saavuttamiseksi alempien tasojen konkreettisilla toimenpiteillä. Systeemin toimintaympäristön rajoitteita ovat tilanneymmärrys, eli se, onko tilannekuva kattava ja vastustajan aikomus tulkittu oikein ja ovatko tunnetut toimintatavat todenmukaisia. Toinen rajoite on toimintavaihtoehtojen arviointi. Molempiin voidaan nähdä vaikuttavan muinainen periaate, että kaikki sodankäynti perustuu harhauttamiseen [118 s. 17], jolloin aikomuksen tulkinta on edelleen vaikeampaa. Kolmas rajoite on annetun tehtävän reunaehdot sekä lait, normit ja ohjeistukset, joiden puitteissa komentajan päätöksen tulee olla toteutettavissa. Systeemissä piilevä, potentiaalisin mahdollisuus muutokselle tekoälyn näkökulmasta sijaitsee aiemmin mainitussa käsityksessä ISR-prosessin tehostamisessa. On kuitenkin mahdollista hyödyntää tekoälyä toimintavaihtoehtojen kartoittamiseen ja simulointiin. Koska yhteisen operatiivisen tilannekuvan muodostaminen sekä sen muuttaminen tietopohjaksi on jo käsitelty aiemmissa kappaleissa, tässä kappaleessa tarkastellaan erilaisten toimintavaihtoehtojen tuottamista ja näiden keskinäistä arvottamista.

Päätöksentekoon liittyen on huomioitavaa, että jokaiseen päätökseen liittyy riski ja kustannus. Tämä liittyy myös edellisessä kategoriassa tarkasteltuihin tapahtumien todennäköisyyksiin. Esimerkiksi tulkittaessa, että vastustajan sukellusveneen voimankäyttö tietyllä alueella on epätodennäköistä, muttei mahdotonta, jää riski sille, että sukellusvene käyttää voimaa vastoin odotuksia. Realisoituessaan epätodennäköisellä vaihtoehdolla voi olla suuri kustannus operoinnille. Tästä syystä kustannuksen paino tulee ottaa huomioon vaihtoehtojen tarkastelussa.

3.4.1 Pelipuut

Taistelun voittaminen edellyttää, että komentaja tekee optimaalisia päätöksiä suhteessa vastustajan tekemiin päätöksiin. Tämä vastavuoroinen prosessi voidaan nähdä yksinkertaistetusti vuoropohjaisena lautapelinä, jossa pyritään ennakoimaan vastustajan siirto ja valitsemaan oma siirto vastaamaan parhaimmalla tavalla kyseiseen tilannekehitykseen. Vuoropohjaisissa peleissä, joissa kaikki pelielementit ovat molempien osapuolien nähtävissä, voidaan optimaalisia ratkaisuja etsiä tavanomaisena hakuongelmana hakupuusta, jossa optimaalinen ratkaisujen sarja on polku, joka johtaa oman osapuolen voittoon. Pelitilanteet muodostavat pelipuun, jolla mallinnetaan juuresta alkaen mahdollisia siirtoja seuraaviin solmuihin kombinatoriikan keinoin. Tunnetuimpia tällaisia hakualgoritmeja on minimax. [31 s. 163-164] Koko pelipuun läpikäynti on kuitenkin laskennallisesti vaativaa, sillä esimerkiksi minimax-algoritmin tarkastelemien pelitilojen määrä on eksponentiaalinen pelipuun syvyyden funktiona, laskennallisella aikavaativuudella $O(b^m)$, jossa b on sallittujen siirtojen määrä syvyys ja m puun syvyys. Pelipuuta pystytään karsimaan laskennan helpottamiseksi esimerkiksi alpha-beta karsinnalla, jossa algoritmi karsii haarat, jotka eivät voi enää vaikuttaa pelin lopputulokseen, puolittaen aikavaatimuksen eksponentin muotoon $O(b^{\frac{m}{2}})$. Suurissa tila-avaruuksissa tämäkään ei kuitenkaan ole käytännöllinen lähestymistapa. [31 s. 165-169] Esimerkiksi sääntöjensä puolesta yksinkertaisessa Go-pelissä on laskettu olevan arviolta $2{,}1 \cdot 10^{170}$ sääntöjen mukaan laillista siirtoa 19x19 peliruudukossa [119], jolloin karsittukaan pelipuu ei ole mahdollinen keino optimoida jokaista siirtoa. Todellisuudessa myös ei-säännönmukaiset ulkoiset tekijät muokkaavat päätöstilaa ennakoimattomasti.

Monissa peleissä on ennalta-arvaamattomuutta mallintava satunnainen elementti, kuten noppien heittäminen. Tällöin puhutaan stokastisista peleistä, jotka ovat periaatteeltaan hieman lähempänä taistelukentän päätöksenteon ongelmia. Kun satunnaisuus vaikuttaa siirtojen arviointiin, ei voida karsia optimaalisessa pelissä mahdottomaksi tulkittuja siirtoja, sillä optimaalinen pelaaminen olettaa molempien osapuolten tekevän pelin sääntöjen puitteissa optimaalisia siirtoja. [31 s. 177-180] Rajoitteena suhteessa todellisuuteen on edelleen havaittavuus, sillä peleissä vastustajan elementit ja niiden siirrot ovat usein havaittavissa ja tiedossa, toisin kuin sodankäynnissä. Osittain havaittavat pelit, kuten shakin pohjalta toteutettu Kriegspiel, tuovat jonkin verran sodan kitkaa pelaamiseen. Kyseisessä pelimuodossa pelaaja näkee vain omat nappulansa. Kriegspielin tapauksessa voittava strategia on kartoittaa kaikki loogiset peliasetelmat saatujen havaintojen perusteella. Jokaisen siirron on oltava osa sarjaa, joka johtaa voittoon, siitä tilanteesta, joka on havaittu. Näin vastustajan käsitys tilanteesta on irrelevantti, sillä strategian on toimittava myös siinä tapauksessa, että vastustaja näkee kaikki toisen puolen elementit. Vaihtoehtona on probabilistinen lähestyminen suhteessa voittavan puolen siirtojen satunnaisuuteen. Perusedellytys on kuitenkin sama, sillä siirtojen on toimittava jokaisessa mahdollisessa pelilaudan tilanteessa kyseisessä tilanneymmärryksessä. [31 s. 180-181]. Kriegspiel, nimensä mukaisesti, vastaa näiltä ominaisuuksiltaan sodankäynnin päätösavaruuden ongelmiin ja kompleksisuuteen. Kuitenkin shakki on säännöstöltään ja muuttujiltaan todellisuutta huomattavasti yksinkertaisempi ongelma Kriegspielin käyttämästä osittaisesta havaittavuudesta huolimatta.

Pelipuiden tarkastelu auttaa ymmärtämään sodankäynnin päätöksenteon monimutkaisuutta. Komentajan päätöksen perustana tulee olla arvio vastustajan toiminnasta, samalla tavalla kuin erilaiset algoritmit arvioivat omia siirtoja vastustajan siirtojen suhteen tai toteuttavat omat siirrot siten, ettei vastustajan siirrolla ole väliä, vaan se oletetaan omalta kannalta pahimmaksi mahdolliseksi vaihtoehdoksi. Kuitenkin suhteellisen yksinkertaiset pelit, joissa edistyneet algoritmit toimivat tehokkaasti, ovat osapuolten välillä yksinkertaisia ja symmetrisiä, toisin kuin sodankäynnin todellisuus.

Vastustajan toimintavaihtoehtojen todennäköisyyksien mallintaminen esimerkiksi Bayes – verkkona taas edellyttää joko asiantuntijoita, jotka kykenevät mallintamaan kyseisen verkon riittävällä tarkkuudella, tai varsinaisen sodankäynnin dataa, jolla voidaan kouluttaa kyseinen verkko. Koska vastustaja pyrkii lähtökohtaisesti salaamaan omat toimintamallinsa ja suorituskykynsä, tällaisen Bayes-verkon rakentaminen tai kouluttaminen riittävällä tarkkuudella on haastavaa. Mikäli vastustajasta on olemassa riittävästi suorituskykyjen ja toimintamallien dataa voidaan sodankäynnin taktiikoita tarkastella simulaatioilla.

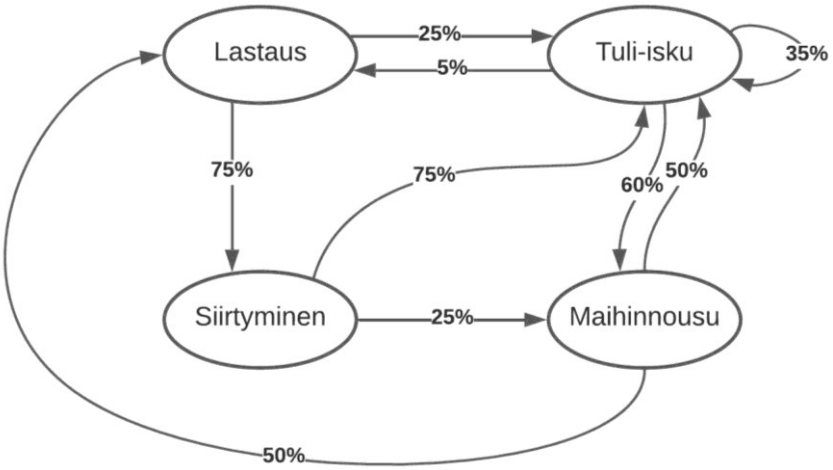
3.4.2 Simulaatiot

Yksinkertaisia simulaatioita voidaan toteuttaa esimerkiksi satunnaistettujen näytteiden algoritmeilla, jotka tunnetaan myös Monte Carlo -algoritmeina. Monte Carlo -algoritmeja voidaan käyttää muun muassa approksimoimaan päättelyä Bayes-verkoissa generoimalla satunnaista, jonkin todennäköisyysjakauman mukaista, dataa kyseiseen verkkoon ja laskemalla saaduista arvioista niiden esiintymistodennäköisyys. Mitä suuremmalla määrällä simuloituja näytteitä verkkoa simuloidaan, sitä luotettavampia ennusteita siitä saadaan tulokseksi. [31 s. 530-531]. Toinen vaihtoehto on Markovin ketju, jossa seuraava tila päätellään pelkästään nykyisen tilan perusteella, ja tätä ketjua simuloidaan Monte Carlo -algoritmilla. Syöttämällä satunnaisia arvoja nykyiseen tilaan päätellään mahdolliset seuraavat tilat päätyen jälleen todennäköisimpiin lopputulemiin sitä paremmin, mitä suurempi määrä satunnaisia

simulointeja suoritetaan. [31 s. 535-536] Näiden menetelmien etu on laskennan yksinkertainen toteutus. Ongelma taas on käsitys vastustajan toimintavaihtoehdoista Bayes-verkon tai Markovin ketjun laatimiseksi. Jotta voidaan luoda simuloimalla käyttökelpoisia toimintamalleja, pitäisi kyetä luomaan verkko vastustajan toiminnan vaiheista ja näiden keskinäisistä riippuvuuksista. Suoraviivainen taktinen skenaario on toteutettavissa käyttäen tunnettuja taktisia periaatteita. Esimerkiksi maihinnousun aloittaminen edellyttää oletettavasti ainakin seuraavat vaiheet:

1. Lastaus
2. Osaston siirtyminen kohti mahdollista maihinnousupaikkaa
3. Tuli-isku
4. Maihinnousu

Olettaen, että minkä tahansa kohdan osalta pitää ennakoida seuraava tapahtuma 24 tunnin sisään ja näiden välinen säännöstö eli tilojen välisten siirtymien todennäköisyydet tunnetaan, voidaan muodostaa Markovin ketju esimerkiksi seuraavasti:



KUVA 22: Markovin ketju maihinnousun todennäköisyydestä seuraavan vuorokauden aikana

Kuvan 22 esimerkissä Markovin ketjusta kyetään päättelemään yhden havaitun tilan pohjalta seuraavien mahdollisten tilojen todennäköisyydet. Esimerkiksi jos on havaittu lastaus, on 75 % todennäköisyys, että osasto aloittaa siirtymisen seuraavan vuorokauden aikana, ja 25 % todennäköisyys, että vastustaja suorittaa tuli-iskun. Ketjusta voidaan laskea tietyn tilan saavutettavuus lähtötilaan nähden jollakin määrällä askelia, jotka kuvaavat yleensä aikaa. Kyseisessä esimerkissä oletetaan vuorokauden tapahtuma-aika siirtymille. Havaittaessa lastaus, voidaan laskea eri tilojen todennäköisyys esimerkiksi kolme vuorokautta eteenpäin, jolloin saadaan posterioritodennäköisyys eri tapahtumille yhden solmun lähtöasetelmasta alkaen.

Näin Bayes-verkoilla, Markovin ketjuilla ja satunnaismuuttujia hyödyntävillä simulaatioilla pystytään päättelemään todennäköisimpiä vaihtoehtoja tilannekehitykselle. Datan saatavuus ja mallien tuottaminen sekä todentaminen pysyvät ongelmallisina edellä mainituista syistä vastustajan suorituskykyihin ja toimintamalleihin liittyen. Lisäksi muodostettuihin malleihin liittyy ongelma päätöksen kustannuksen arvioinnista suhteessa riskiin. Noudattaen aiempaa esimerkkiä, jos tuli-iskun todennäköisyys todetaan alhaiseksi ja näin ollen jätetään huomiotta, on määritettävä kustannus sille, että epätodennäköinen vaihtoehto toteutuukin, sillä riski on arvotettava paitsi todennäköisyyden myös vaikuttavuuden suhteen.

3.4.3 Kompleksiset pelit, simulaatiot ja tekoäly

Siinä missä tekoäly on voittanut ihmisen shakissa, Jeopardy!- ja Go-peleissä sekä oppinut pelaamaan Atari-pelejä [32 s. 7; 120], ei mikään näistä läpimurroista vastaa suoraan sodankäynnin ongelma-alueeseen päätöksenteon osalta. Havaittavat ja osittain havaittavat peliympäristöt muodostavat monimutkaisia mutta todellisuutta yksinkertaisempia tila-avaruuksia. Lisäksi vuoropohjaisuus ei vastaa todellista sodankäyntiä, jonka tapahtumat ovat päällekkäisiä ja reaaliaikaisia.

Googlen DeepMind tytäryhtiön luotua Go-pelin maailmanmestarin voittanut AlphaGo-tekoäly alalla vallitsi käsitys, että reaaliaikaiset strategiapelit kuten suosittu StarCraft II olisivat edelleen äärimmäisen vaikea haaste tekoälylle, sillä tekoälyn pitäisi oppia reagoimaan reaaliaikaisesti suunnattoman tila-avaruuden haasteisiin. AlphaStar-tekoäly voitti maailman rankingin parhaat pelaajat kuitenkin jo kolme vuotta myöhemmin. Vaikka edelleen kyseessä on huomattavasti reaalimaailmaa yksinkertaisempi peli, on AlphaStarin suoritus vakuuttava. Lisäksi tekoälyä oli rajoitettu vastaamaan ihmisten rajoitteita pelaamisen suhteen, muun muassa sallimalla sille vähemmän päätöksiä per aikayksikkö kuin ihmiset tekevät keskimäärin ja rajoittamalla yksiköiden kontrolli siten, että tekoäly joutui siirtämään kuvakulman alueelle, jolla se teki toimenpiteitä. Kyseinen tekoäly koulutettiin hyödyntämällä neuroverkkoja, imitaatio-oppimista, vahvistusoppimista ja moniagenttioppimista. Ohjatulla oppimisella tekoäly koulutettiin imitoimaan ihmisten pelityyliä laittamalla se oppimaan pelattujen pelien taltioinneista. Kyseisiä taltioita oli lähes miljoona kappaletta. Opetettua mallia hiottiin opettamalla sille vain tiettyjä voittoon johtaneita taltioita. Tämän jälkeen toteutettiin vahvistusoppiminen laittamalla tekoälyagentteja pelaamaan toisiaan vastaan. Useiden mahdollisten strategioiden mahdollistamiseksi käytettiin moniagenttioppimista, jotta tekoäly ei keskity vain johonkin kapeaan otantaan voittavia lähestymistapoja. [121]

Kriittisesti voidaan todeta, että vaikuttavuudestaan huolimatta AlphaStar ei vastaa ongelmanratkaisukyvyltään suoraan todellisen sodankäynnin ongelmiin, joiden kombinatorinen avaruus on vielä kompleksisempi. On kuitenkin huomioitava, että AlphaStar on käyttänyt pelissä ennen näkemättömiä strategioita ja yllättänyt innovatiivisuudellaan parhaat ihmispelaajat. Koska sotilaallisen taktiikan ja strategian konventionaalinen kehitys ovat hitaan inkrementaalisia prosesseja, jotka edellyttävät lähes väistämättä todellista konfliktia tullakseen testatuksi, näitä tuloksia AlphaStarin kehityksessä voidaan pitää merkittävinä. Sotilaallinen hyödynnettävyys edellyttäisi riittävän tarkan ja todenmukaisen simulaattorin toteuttamista. Simulaattoria voitaisiin käyttää niin taktiikan ja strategian tutkimuksessa ja koulutuksessa kuin tekoälyn koulutuksessa.

Olettaen, että simulaattori rakennettaisiin peliksi, johon on mallinnettu todellisen kaltainen toimintaympäristö vaihteluineen, omat joukot todellisin suorituskyvyin ja vastustajan joukot parhaan olemassa olevan tiedon mukaisiksi, kyettäisiin peliympäristöön teoriassa kouluttamaan samantyylinen tekoäly kuin AlphaStar. Tällaisella tekoälyllä kyettäisiin mallintamaan erilaisiin tilannekehityksiin soveltuvia, mahdollisesti täysin uudenlaisia taktiikoita ja muodostamaan tilastollista tietoa siitä, millaisiin lopputuloksiin erilaiset tilannekehitykset johtavat. Näin voitaisiin muodostaa joko monimutkaisia Bayes-verkkoja tai Markovin ketjuja erilaisten tilannekehitysten todennäköisistä posterioreista, tai antaa simulaattorilla koulutetun älyn tehdä suoraan ennusteita ja suosittaa toimenpidevaihtoehtoja sen kokemuspohjan perusteella tunnistetun tilannekuvan pohjalta.

3.4.4 Hakualgoritmit ja optimointi

Aiemmin esitellyt vaihtoehdot toimenpidevaihtoehtojen kartoittamiseksi ovat joko liian monimutkaisia tai yksinkertaisia suhteessa siihen, miten paljon ja miten luotettavaa dataa oletetun vastustajan todellisista kyvyistä, toiminnasta ja tavoitteista on hyödynnettävissä. Yksi, suoraviivaisempi ratkaisumalli on tulkita päätöksentekotilanne siten, että havainnointi ja orientaatio tuottavat komentajalle käsityksen vallitsevasta tilanteesta ja sen ennustetusta kehityksestä aikavälillä $0 + t$, johon komentaja muodostaa geneerisen tahtotilan. Geneerinen tahtotila voi olla esimerkiksi vedenalaisen valvonnan painopisteen muodostaminen tiettyjen syvyyskäyrien alueelle tai valvonnan ulottaminen kattamaan maksimaalinen pinta-ala pohjoisella Itämerellä. Tällöin ratkaistava ongelma on yksinkertaisempi optimointiongelma, jossa pyritään maksimoimaan tai minimoimaan tietty muuttuja tai muuttujat sijoittamalla alusyksiköitä kykyineen tietyille alueille. Esimerkiksi hill-climb on ahne hakualgoritmi, joka pyrkii kohti tavoitetta eli minimiä tai maksimia valitsemalla lähimmän tavoitteen suuntaan johtavan naapurin tila-avaruudesta. Ahneena algoritmina hill-climb ei välttämättä löydä parasta ratkaisua eli globaalia maksimia tai minimiä, mutta se löytää joka tapauksessa hyvän ratkaisun nopeasti. [31 s. 121-122]. Algoritmia voidaan myös iteroida stokastisesti eri vaihtoehtojen tuottamiseksi.

Esimerkiksi mallinnettaessa Itämeri kaksi- tai kolmiulotteiseksi matriisiksi saadaan muodostettua taso, jonne yksiköt ovat sijoitettavissa suhteessa maaston muotoihin ja käytössä oleviin kulkuväyliin. Mallintaessa omat joukot niiden valvontakykyjen mukaan saadaan jokaisella yksikölle attribuutti valvonnan osalta, esimerkiksi pintavalvontakyky. Mikäli komentaja haluaa tehostaa pintavalvontaa levittämällä sen mahdollisimman laajalle alueelle, hill-climb - tai vastaavan optimointialgoritmin tehtävä on maksimoida valvottava pinta-ala käytettävissä olevilla yksiköillä tietyllä alueella. Heuristinen algoritmi löytää nopeasti ehdotuksen alusten sijoittelusta alueelle tehtävän täyttämiseksi ja on siten muodostanut toimenpidevaihtoehdon hyödynnettäväksi. Sama toimintalogiikka toimii jokaisen dimension osalta sekä vaikutuskykyjen osalta.

Jotta malli olisi edelleen käyttökelpoisempi, voidaan algoritmin optimoitavaksi ottaa useampi kuin yksi attribuutti ja siten yhdistää haluttuja tavoitteita, tai yhdistää useiden algoritmien tuloksia. Ihmisenkin on helppo luonnostella alusten sijoittelu siten, että näiden sensorit kattavat mahdollisimman suuren alueen, mutta yhdistettäessä esimerkiksi kolme eri attribuuttia kuten vedenalainen valvonta, pintavalvonta ja ilmatorjunnan kantamat, saattaa hahmottaminen olla haastavaa. Hakualgoritmeja voidaan käyttää kompleksisempien ongelmien ratkaisuun hyvien toimenpidevaihtoehtojen tuottamiseksi. Liitteessä 9 on havainnollistettu yksinkertainen esimerkki symbolisen tekoälyn, hakualgoritmien ja optimoinnin käytöstä alustaisteluosaston ryhmittymisessä.

3.4.5 Yhteenveto

Tekoälyn hyödyntäminen päätöksenteon tukena taktisella tasalla on haastavaa. Siinä missä havaitseminen ja orientaatio tuottavat operaattoreille ja taisteluosaston komentajille käyttökelpoista dataa yksittäisten elementtien ja näiden aikomusten tulkitsemiseksi ja taisteluteknisten toimintojen ohjaamiseksi, on taktisen mittakaavan päätöksenteon tukeminen kompleksisempi ongelma.

Esitellyistä vaihtoehdoista voidaan tulkita kaksi vaihtoehtoa. Ensimmäinen on yksinkertaistetut, asiantuntijatietoon ja kerättyyn dataan perustuvat mallit kuten Bayes-verkot ja hakualgoritmit. Toinen vaihtoehto on simulaattorilla tai pelillä koulutettu tekoäly, jolla kyettäisiin tuottamaan kompleksisesta datasta tarkempia toimenpidesuosituksia. Ensimmäinen vaihtoehto auttaa arvioimaan vastustajan toteuttamien toimenpiteiden todennäköisyyksiä ja

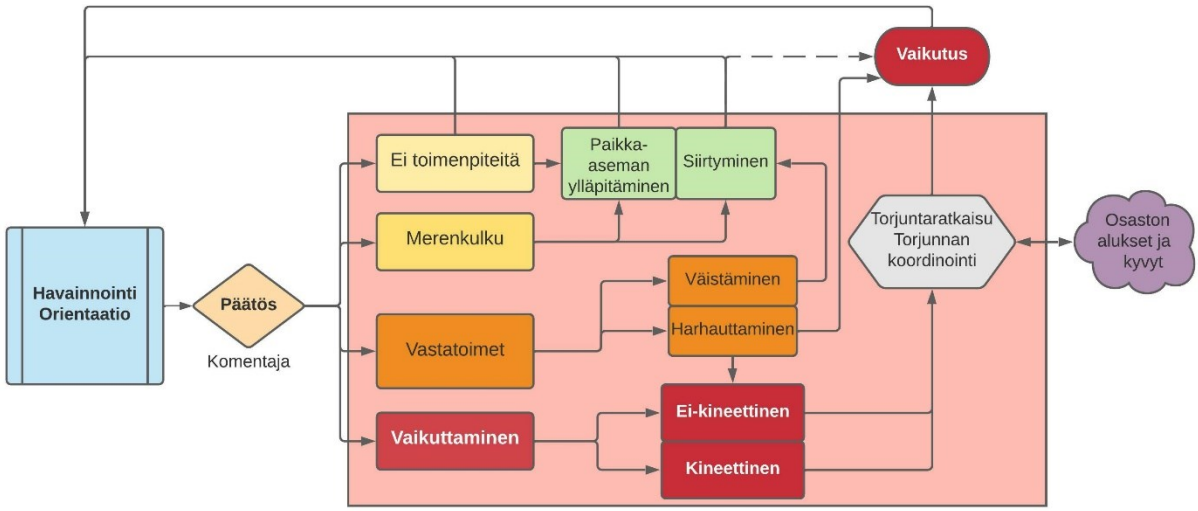
valitsemaan oman toimenpiteen niihin vastaamiseksi, kun taas jälkimmäinen voi teoriassa suosittaa esimerkiksi ryhmityksen optimoimista tulkittuun uhkaan vastaavaksi täysin uudella tavalla, ennustaen tilannekehitystä havaittua nykyhetken tila-avaruutta pidemmälle. Hyödynnettävyyden kannalta ensimmäinen on helpompi toteuttaa ja korreloi paremmin nykyisen päätöksentekoprosessin kanssa. Mallissa muutettaisiin asiantuntijoiden työpanos välittömästi hyödynnettäväksi järjestelmäksi, joka tarjoaa esimerkiksi tiedustelun asiantuntijoiden tietopohjaan nojaavan tulkinnan tilannekehityksestä nopeammin kuin nykymalli, jossa havainnot kiertävät erikseen analysoitavaksi ja tulkittavaksi ylempien johtoportaiden tiedusteluosastoille. Näin ollen kyseisen toteutustavan riskiprofiili voidaan myös nähdä matalana ja järjestelmän luotettavuus korkeana, olettaen, että säännöstö on rakennettu huolellisesti ja sitä päivitetään jatkuvasi uuden datan perusteella. Lisäksi hakualgoritmeilla kyettäisiin tukemaan taisteluosaston esikunnan suunnitteluosaston tehtäviä tuottamaan ehdotuksia esimerkiksi alusten ryhmitysmuutoksiin ja huoltoikkunoiden ennakointiin.

Jälkimmäinen vaihtoehto on hypoteettinen ja monimutkaisempi toteuttaa. Työnä vaihtoehto edellyttää simulaattorin rakentamiseksi saman asiantuntijatyön todenmukaisen virtuaalisen ympäristön mallintamiseksi kuin yksinkertaisempikin vaihtoehto, sekä itse simulaattorin kehittämisen työn tämän lisäksi. Koulutusdatan muodostaminen edellyttää mahdollisesti satoja tuhansia pelattuja skenaarioita, jotta tuotetaan riittävä data tekoälyn imitaatio-oppimista varten. Näiden skenaarioiden laatiminen ja pelaaminen edellyttää jälleen kriittistä asiantuntijatyötä sekä ymmärrystä vastustajan toimintamalleista, kyvyistä ja tahtotilasta. Imitaatio-oppiminen voidaan myös sivuuttaa ja käyttää suoraan vahvistusoppimista, jolloin tekoälyagentit saavat käytännössä itse simulaattorin rajoitusten puitteissa löytää voittavat taktiikat. Kyseisellä tavalla voitaisiin muodostaa uudenlaisia toimintatapoja, mutta todennäköisimmin paljastettaisiin pääasiassa simulaattorissa olevia bugeja, vinoumia ja muita ongelmia. Lisäksi on huomionarvoista, että esimerkiksi mainittu Googlen DeepMind on edistynyt yhtiö, jolla on käytössään huippuasiantuntijat tekoälyn kehitykseen sekä kokemus aiemmista vastaavista projekteista. Voidaan argumentoida, ettei kyseinen vaihtoehto ole realistinen kansallisin resurssein ainakaan lähitulevaisuudessa, vaan se edellyttäisi kansainvälistä yhteistyötä ja huomattavia panostuksia kehitystyöhön. Toisaalta tavoitetilan voidaan argumentoida olevan todennäköisesti yhtenevä ainakin lähialueen yhteistoimintaosapuolten kuten Ruotsin merivoimien kanssa, mikä puoltaisi yhteistyötä.

Omien kykyjen suhteen toimenpidevaihtoehtojen optimointi hakualgoritmein on suoraviivaista ja verrattain yksinkertaista. Näin ollen esitetyistä malleista parhaiten soveltuvana nähdään hakualgoritmit, joilla luodaan toimenpidevaihtoehtoja vastaamaan komentajan tahtotilaa, sen sijaan, että pyrittäisiin tuottamaan itse tahtotilaa tekoälyn menetelmin. Tämä lähestymistapa ei poissulje vastustajan toimenpiteiden ennakoimista, mutta vähentää niiden tarkkuuden kriittisyyttä. Mikäli ei kyetä luomaan mallia, joka ennakoi riittävällä tarkkuudella tietystä tilasta muodostuvat mahdollisuudet vastustajan toimenpiteille, voidaan silti optimoida omien joukkojen käyttö uhkaavimman skenaarion mukaisiksi tai minimoimaan päätöksen kustannus suhteessa riskiin.

3.5 Toimenpiteet ja toiminta

Päätöksenteossa valitun toimintavaihtoehdon toteuttaminen edellyttää konkreettisia toimenpiteitä. Tässä työssä taisteluosaston toiminnot erinäisten päätösten täyttämiseksi on jaettu neljään kategoriaan mukaillen RAND tutkimuslaitoksen USV algoritmien tehtäväkenttää taulukosta 2. Vastustajan suuntaan passiiviset vaihtoehdot ovat reagoimattomuus eli se, ettei havainto edellytä toimenpiteitä, sekä merenkululliset toimenpiteet kuten siirtymä tai nykyisen paikan ylläpito. Aktiivisia toimenpiteitä ovat vastatoimet ja vaikuttaminen, sillä näillä pyritään vaikuttamaan joko vastustajan kykyyn toteuttaa omaa tehtäväänsä tai suoraan vastustajan yksiköihin. Passiivisilla toimenpiteillä on tarkoitus ylläpitää tilannetietoisuus ja varautua toteuttamaan aktiivisia toimenpiteitä. Jako ei ole tarkkarajainen, sillä vaikuttamisen toimenpiteet voivat edellyttää merenkulullisia toimenpiteitä, samoin kuin vastatoimet, jolloin toimintojen välille muodostuu keskinäisiä riippuvuuksia.



KUVA 23: Toimenpiteiden systeemi ja toteutuksen prosessi

Kuvassa 23 on esitetty toimenpiteiden systeemi aiempien systeemien ketjussa. Systeemin omistaja on taisteluosaston komentaja ja sen toimijoita ovat alusyksiköt, jotka henkilöityvät alusten päälliköihin, sekä taisteluosaston esikunta. Systeemin asiakas on tehtävän antanut ylempi johtoporras, jonka antaman tehtävän täyttämiseksi toimintaympäristöön vaikutetaan. Rajoitteita ovat paitsi materiaaliset resurssit myös lait ja normit, kuten voimankäytön säädökset. Systeemissä olevat muutostekijät, joihin tekoälyn menetelmiä voidaan hyödyntää, liittyvät pääasiassa toimintojen koordinointiin ja suorittamisen optimointiin. Ympäristön rajoitteita systeemin toiminnalle ovat tekniset ratkaisut, sillä osaston toimenpiteiden optimointi edellyttää hyvin verkottunutta ja tiedonsiirtokapasiteetiltaan robustia johtamis- ja tiedonsiirtojärjestelmää.

Toimenpiteiden tarkastelu on tämän tutkimuksen näkökulmasta kaikista kategorioista yksinkertaisin. Yksinkertaisuus johtuu siitä, että tulkitut toimenpidevaihtoehdot ovat joko yksinkertaisia toteuttaa tai niiden automatisointiin ja toteutuksen optimointiin on jo panostettu merkittävästi, kuten merenkulun ja automaattisen maalittamisen tapauksissa [23; 122 s. 115]. Yksinkertaisin ja yleisin tilanne on se, että uuden tilannekuvaelementin havaitseminen ja havainnon tulkitseminen eivät aiheuta mitään konkreettisia toimenpiteitä. Tällöin tilannekuvaelementti pidetään mahdollisuuksien ja uhka-arvion perusteella tilannekuvassa havainnoimalla sitä jatkuvasti eli muodostamalla seuranta. Jatkuvalla havainnoinnilla pidetään yllä tilannekuva ja -ymmärrys ilman muita tavoitteita kuin tilannekuva

itsessään. Mahdollisimman reaaliaikainen tilannekuva luo pohjan myös muille vaihtoehtoisille toimenpiteille tilanteiden kehittyessä. Toiseksi vaihtoehdoksi alustaisteluosaston yksiköiden toimenpiteille määritetään merenkulullisesti toteutettavat tehtävät, jotka käsittävät siirtymiset sekä nykyisen paikan ylläpitämisen. Taistelualusten perimmäinen tarkoitus on toimia liikkuvana lavettina ja siirtää haluttu suorituskyky tai -kyvyt haluttuun paikkaan tai ylläpitää aiemmin määritetty paikka-asema. Kolmas vaihtoehto on vastatoimet, jonka nähdään pitävän sisällään väistämisen ja harhauttamisen toimenpiteet. Väistäminen on merenkulullinen toimenpide, jossa alus siirretään maaston tai etäisyyden suhteen suojaan joltain uhalta. Harhauttaminen pitää tässä tarkastelussa sisällään taktisen tason harhautukset ryhmitysmuutosten osalta, jotka ovat käytännössä merenkulullisia tehtäviä, sekä säteilynhallinnan toimenpiteet. Säteilynhallinnalla ja herätteitä manipuloimalla kyetään vähentämään tai lisäämään omien joukkojen havaittavuutta ja sitä kautta vaikeuttamaan tilannekuvan muodostusta [101 s. 97]. Harhauttaminen pitää sisällään myös muun muassa harhamaaliheitteet ja herätteiden muokkaamisen esimerkiksi suojasuihkuin, joilla jäähdytetään aluksen pintaa lämpöherätteen minimoimiseksi. Pyrkimys on vaikeuttaa, estää tai harhauttaa vastustajan vaikutuksen kohdistuminen omiin yksiköihin.

Neljäs vaihtoehto, vaikuttaminen, jaetaan kineettiseen ja ei-kineettiseen vaikuttamiseen. Kineettisellä vaikuttamisella tarkoitetaan fyysisen vaurion ja vahingon tuottamista vaikutuksen kohteeseen konventionaalisin keinoin kuten ammuksin ja räjähtein. Ei-kineettisellä vaikuttamisella tarkoitetaan kohteeseen vaikuttamista tuottamatta sille fyysisiä vaurioita tai vahinkoa, esimerkiksi kyberhyökkäyksellä tai elektronisen sodankäynnin vaikuttamisella kuten aktiivisella häirinnällä.

Merenkulullisten tehtävien tehostamista ja toteuttamista tekoälyn menetelmin ei tarkastella osana tätä tutkimusta. Autonomisia merenkulkujärjestelmiä, niiden algoritmeja ja älykkäitä järjestelmiä on tutkittu muihin kategorioihin nähden huomattavasti ja kyseisiä järjestelmiä on kaupallisesti saatavilla. Näin ollen tässä osiossa tarkastellaan seuranta, vastatoimet ja vaikuttaminen, sillä kategoriat ovat keskinäisesti riippuvaisia ja näyttäytyvät pääasiassa optimointiongelmina taisteluosaston kyvykkyyksien allokaatiolle.

Johtuen vaikuttamisen ja vastatoimien todellisen datan puutteesta, ei kyseistä ongelma-aluetta voida lähestyä suoraviivaisesti koneoppimisen keinoin. Mikäli vastatoimia ja vaikuttamista halutaan tehostaa datalähtöisesti, on suurin haaste todenmukaisen datan simulointi, kuten aiemmassa päätöksenteon kategoriassa. Mikäli esimerkiksi ohjustulenkäyttöä mallinnetaan ja simuloidaan oletetun vastustajan yksiköitä vastaan, voidaan vaikuttamisesta tuottaa dataa hyödynnettäväksi koneoppimisen menetelmin. Tämä ratkaisu on kuitenkin yhtenevä aiemmin kuvatun simulaattoriympäristön kanssa, eikä sitä tarkastella tässä yhteydessä uudelleen.

3.5.1 Seuranta, vastatoimet ja vaikuttaminen

Jos tilannekuvaelementti ei edellytä toimenpiteitä, mutta pysyy jonkinasteisen mielenkiinnon kohteena havainnointialueella, toteutetaan sen suhteen seuranta. Tilanteen ja tilannekuvaelementtien seuranta on passiivinen toimenpide, sillä mahdollisten aktiivisten sensorien lisäksi elementtiin ei kohdistu aktiivisia toimenpiteitä eikä se aiheuta omissa joukoissa muita toimenpiteitä. Riippumatta suoritettavasta operaatiosta ja kriisinvaiheesta, seuranta on suora jatkumo tilannekuvaelementin havaitsemiselle tilanneymmärryksen ylläpitämiseksi. Toimenpiteen tarkoitus on mahdollistaa tilannekehityksen ennakointi ja toimenpidevalinnan muutoksen tehokas toteuttaminen.

Seuranta ei tekniseltä toteutukseltaan poikkea havainnoinnista, sillä seuranta on määritelmällisesti jatkuvaa havainnointia. Sen tarkoitus on ylläpitää tilannekuva, mahdollistaa OODA-silmukan uudet iteraatiot sekä tarvittaessa maalinosoitus. Verrattuna havainnoinnin kategoriaan, tässä kohtaa tekoälyn menetelmät tuovat lisäarvoa verkottuneen taisteluosaston kykyyn muodostaa tiiviisti kytketty sensoriverkko. Yksi ehdotus tehokkaan sensoriverkoston tuottamiseksi on muodostaa sensorifuusiolla raakadatasta piirredataa alusyksiköillä ja jakaa se taisteluosaston kesken. Taisteluosastotasalla toteutettaisiin keskitason fuusio, jossa eri yksiköiltä kerätty piirredata yhdistetään. Tähän kyetään jollain tasolla myös nykyisillä datalinkeillä ja taistelunjohtojärjestelmien mahdollistamalla fuusiolla. Seurantaa suorittavat ihmisoperaattorit, allokoimalla sensoreita tiettyihin maaleihin oman harkintansa mukaan ja jakamalla niitä tilannekuvaan tunnistustensa perusteella.

Mikäli tekoälyavustettu havainnointi kykenee tuottamaan ihmisoperaattorien tulkintoja vastaavia luokituksia havaittujen tilannekuvaelementtien tyypille, aikomukselle ja uhkaavuudelle, voidaan tunnistettu data jakaa muille yksiköille automaattisesti. Alukset muodostavat tällöin linkittyneen sensoriverkon, jonka jokaisella solmulla on käytössään tietyt sensorijärjestelmät. Korkeammalla tasolla, esimerkiksi taisteluosaston esikunnan solmussa, toimiva tekoäly voi jakaa sensoriallokaatiosuosituksia tai -komentoja kokonaistilannekuvaan perustuvan arvion mukaisesti. Älykäs agentti kykenisi reagoimaan symbolisen säännöstön perusteella muun muassa siihen, milloin maali on siirtymässä tietyn yksikön sensorikantaman ulkopuolelle, onko se mahdollista allokoida alusosaston toiselle yksikölle ja edellyttääkö sen uhkaavuus parempaa seurantaa eri sensorijärjestelmän tai useamman yksikön toimesta. Älykäs agentti voisi myös huomioida säteilynhallinnan vuorottelemalla yksiköitä ja minimoimalla säteilyn tietystä pisteestä vastustajan havainnoinnin häiritsemiseksi. Näin verkottuneen alustaisteluosaston sensorijärjestelmien kokonaisuuta voitaisiin hallita tehokkaan seurannan mahdollistamiseksi ja linkittää seuranta vastatoimien toteuttamiseen.

Osaston sensorijärjestelmiä hallinnoivan älyn tuottaminen edellyttää asiantuntijoita ja asiantuntijatyötä, jolla aselajiosaaminen ja aselajikäskyjen reunaehdot muutetaan asiantuntijajärjestelmäksi. Seurannan tehostamisen kannalta symbolisen tekoälyn asiantuntijajärjestelmä ja ohjelmistoautomaatio vaikuttavat käyttökelpoisimmalta tavalta tehostaa taisteluosaston toimintaa tilannekuvan ylläpitämisessä ja maalinosoituksen mahdollistamisessa, siinä missä koneoppimista ja syväoppimista hyödynnetään itse tilannekuvaelementtien havaitsemisessa ja tunnistamisessa.

Sensoriverkostojen optimointiin voidaan hyödyntää muun muassa geneettisiä algoritmeja [123 s. 109-116; 124 s. 349-354]. Geneettisillä algoritmeilla on kyetty muun muassa optimoimaan sensoriallokaatiota ja sensoriverkoston seurantaa energiatehokkaaksi. Tulevaisuudessa alusyksiköiden muodostaessa edelleen tiiviimmin kytketyn sensoriverkoston on verkon toiminta niin havainnoinnin kuin seurannan optimoimiseksi mahdollista esimerkiksi yhdistämällä keskitason sensorifuusio ja hakualgoritmien käyttö taisteluosaston toiminnan tehokkaaksi koordinoimiseksi.

Sensoriverkoston seurannan optimoinnin yhteydessä kyetään optimoimaan samanaikaisesti säteilynhallinta kunkin tehtävän tai riskiprofiilin mukaiseksi. Annettaessa sensoriverkostolle rajoitteita esimerkiksi yhtäjaksoisen säteilyn ajalliselle pituudelle korkean uhkaprofiilin tehtävässä voidaan optimoinnilla laskea dynaamisesti sensoriverkostolle paras mahdollinen sekvenssi säteilyjaksoille yksikköjen välillä ylläpitäen paras mahdollinen tilannekuva pai-

nopistealueella. Rajoitteena optimoinnille on sensorijärjestelmien kontrolli ja tiedonsiirtoyhteydet. Mikäli osa taisteluosaston yksiköistä on heikon tiedonsiirtoyhteyden päässä taisteluosaston johtoportaasta, voidaan kyseiselle yksikköryppäälle muodostaa oma solmu, joka prosessoi lyhyempien etäisyyksien ja siitä juontuvien korkeampien tiedonsiirtotaajuuksien ja -kapasiteettien ryppäässä alueensa havainnot yksinkertaisemmaksi piirreavaruudeksi. Prosessoitu piirreavaruus taas voidaan siirtää optimaalisesta yhteyssolmusta johtoportaaseen liitettäväksi yhteiseen tilannekuvaan. Tiedonsiirron optimointi on toteutettavissa esimerkiksi muodostamalla taisteluosaston kokoonpanosta verkkorakenne ja käyttämällä reaaliaikaisesti mitattavia, käytettävissä olevia tiedonsiirtonopeuksia eri solmujen välisinä kaarina. Tällöin voidaan valita optimaaliset yhteyssolmut eri yksiköiden tai yksikköryppäiden välillä maksimivirtausalgoritmilla, joka hakee verkkorakenteesta suurimman tiedonsiirtokapasiteetin lähtösolmusta maalisolmuun [49 s. 153]. Yhdistämällä erilaisia hakualgoritmeja kyetään teoriassa muodostamaan dynaamisesti optimoituva hallintajärjestelmä taisteluosaston yksiköiden valvontakykyjen koordinoimiseksi tehtävän täyttämiseksi ja uhkien väistämiseksi. Modernit datalinkkijärjestelmät optimoivat jo tiedonsiirtokapasiteettien allokoimista ja tiedonsiirtoverkon ylläpitoa automaattisesti. Uutena mahdollisuutena on siirrettävän datan keräämisen ja jakamisen optimointi sekä kollektiivinen säteilynhallinta.

Uhkaperusteinen järjestelmäallokaatio maaleihin vaikuttamiseksi on ollut aiemmin todetusti jo pitkään osa laivastojen taistelunjohtojärjestelmien kehitystä alustasalla. Näin ollen tarkasteltavaksi valittu mahdollisuus tekoälyn hyödyntämiselle on osaston vaikuttimien koordinoitu, optimaalinen hyödyntäminen. Koska harhamaaliheitteet ja muut toimenpiteet vastustajan kineettisen voiman väistämiseksi liittyvät kiinteästi samaan havainnon ja toimenpiteen sykliin käsitellään ne samassa yhteydessä. Vaikuttamisen järjestelmät on lähtökohtaisesti suunniteltu tiettyyn tai tiettyihin torjuntatyyppeihin ja vaikutuskohteisiin, mikä näkyy jo järjestelmien nimissä. Oikean vaikuttimen allokointi tiettyyn uhkaan on suoraviivainen propositiologinen yhtälö, esimerkiksi uhkaavaan ilmamaaliin vaikutetaan ilmatorjuntajärjestelmällä ja lukittuneeseen hakupäähän vaikutetaan elektronisella vaikuttamisella tai harhamaaliheitteiden avulla.

Vaikuttamisen optimointi ei poikkea ongelmatilanteena sensoriverkoston optimoinnista, sillä kyseessä on uhkaan suhteutuvien kyvykkyyksien saatavuuden, vaikuttavuuden ja kapasiteetin optimointi. Tällä tarkoitetaan käytettävissä olevia suorituskykyjä, tulivoimaa ja määrällistä sekä etäisyydellistä saatavuutta. Optimaalisessa torjunnassa kohteeseen vaikutetaan pienimmällä riittävällä määrällä resursseja riittävän vaikutuksen tuottamiseksi ennen kuin kohde ehtii vaikuttaa omiin yksiköihin omilla kyvyillään. Näin ollen hakualgoritmin tulee arvioida, mikä yksikkö tai yksiköt ovat tehokkaan kantaman etäisyydellä kohteesta ja mitkä näiden suorituskyvyistä ovat soveltuvia ja käytettävissä kyseiseen torjuntaratkaisuun. Lopputuloksen tulisi olla optimaalinen torjuntaratkaisu, joka allokoi soveltuvimmat kyvyt yksiköiden välillä suoritukseen toteuttamatta merkittävää yli- tai alivaikuttamista. Oleelliseksi muodostuu kohteen utiliteetin arviointi orientaatiossa. Korkean utiliteetin kohdetta vastaan on kyettävä käyttämään enemmän resursseja. Edellytyksinä optimoinnille on tiivis verkottuneisuus ja reaaliaikainen tilannetieto kunkin yksikön statuksesta niin sijainnin kuin sensorien sekä vaikuttimien tilan ja ampumatarvikemäärien suhteen. Liitettäessä optimoitavaan rakenteeseen täydennyspaikkojen sijainnit ja täydennyskapasiteetit kyetään myös optimoimaan lyhyet siirtymät ampumatarvikkeiden täydentämiseksi.

Vastaavat periaatteet toimivat myös esimerkiksi miinoitusoperaatioiden koordinoinnissa. Miinoitustehtävät kohdistetaan tiettyjen, kriittisiksi tulkittujen alueiden suluttamiseksi. Pääperiaate on toteuttaa ensimmäiset miinoitteet

tietyn suunnitelman mukaan heti kriisin alkuvaiheilla ja täydentää näitä kriisin kehittyessä vastustajan suunnitelmia ennakoiden. Tällöin optimoitavaan verkkorakenteeseen tulee liittää logistiset solmut, joista miinat lastataan, ja tieto saatavilla olevista miinamääristä ja -tyypeistä. Näiden tietojen perusteella voidaan optimoida reitit täydennyspaikoista kriittisiksi tulkituille miinoitusalueille ja allokoida taisteluosaston yksiköitä dynaamisesti tehtävien toteuttamiseksi ajallisesti lyhimpiä polkuja pitkin.

Koska edellä kuvatuista kategorioista ei tämänhetkisen tiedon perusteella ole olemassa dataa koneoppimisen hyödyntämiseksi, jäävät hakualgoritmit lähtökohtaisesti parhaaksi vaihtoehdoksi ongelmatilanteen ratkaisulle. Tiettyjen prosessien suunnitteluun kuten miinoitusoperaatioiden suorittamisen optimointiin voidaan hyödyntää myös simulaatioita. Mallintaessa tavoitetila, yksiköiden miinoituskapasiteetit, siirtymisnopeudet ja täydennyspaikat voidaan Monte Carlo -simulaatiolla arvioida erilaisia yksikkökombinaatioita ajallisesti tehokkaimman suorituksen löytämiseksi ja tehdä miinoitussuunnitelmien toteutus nojaten näin löydettyihin ratkaisuihin.

3.5.2 Yhteenveto ja konsepti

Tässä tarkastelussa toimenpiteiden systeemin tehostamiseen soveltuviksi tekoälymenetelmiksi muodostuvat asiantuntijajärjestelmät sekä hakualgoritmit. Aiemmissa kategorioissa tehdyt toimenpiteet suorittavat monimutkaisempaa data-analyysia ja mahdollistavat prosessoidun datan tuottamisen toimenpiteiden valinnan perustaksi. Toimenpiteiden tehostamiseksi on ylläpidettävä tilannetieto suorituskykyjen keinovalikoiman saatavuudesta taisteluosaston sisällä ja kyettävä allokoimaan tilanteeseen nähden optimaalisin toimenpide soveltuville yksiköille. Käytännössä soveltuvimman toimenpiteen valitseminen voi tapahtua symbolisen tekoälyn toimenpitein propositioloogisesti yhdistämällä ehtoja ja valitsemalla näiden pohjalta tilanteeseen käytettävä toimenpide.

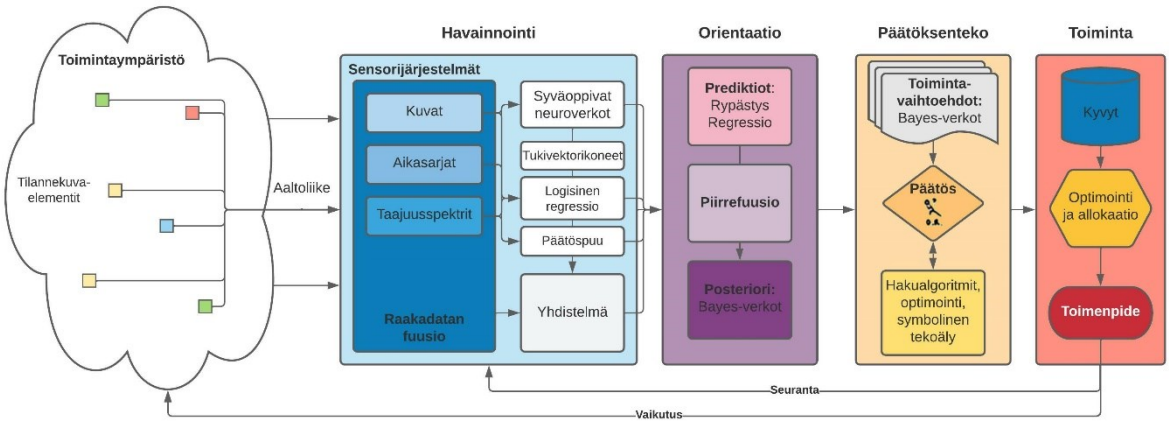
Toimenpiteiden toteuttamissa voidaan hyödyntää hakualgoritmeja kuten geneettisiä algoritmeja optimaalisen ratkaisun laskemiseen verkottuneen taisteluosaston yksiköiden ominaisuuksista suhteessa uhkaavaksi tulkittuun kohteeseen. Hakualgoritmeja voidaan hyödyntää osana asiantuntijajärjestelmää, jolloin asiantuntijajärjestelmän valittavaksi tuotetaan hakualgoritmeilla laskettuja vaihtoehtoja.

Vaikka kompleksisen simulaattorin rakentaminen ja kategoriaan soveltuvan datan tuottaminen sen avulla rajattiin tarkastelun ulkopuolelle, on vaihtoehto sille olemassa. Päätöksenteon tueksi luotu simulaattori edellyttää yksiköiden suorituskykyjen mallintamisen, jolloin sillä voidaan tuottaa dataa myös toimenpiteiden vaikuttavuudesta ja siten koneoppimismalleja toiminnan tehostamiseksi. Tämän kaltaisen simulaattorin ongelmat ovat kuitenkin samat kuin edellisessä alaluvussa. Tästä syystä yksinkertaisin keino tehostaa toimenpiteiden toteuttamista tekoälyn menetelmin on muodostaa asiantuntijajärjestelmä vastaamaan nykyistä, taistelukeskushenkilöstön tuottamaa työpanosta, ja optimoimaan sen vaikuttavuus hyödyntämällä hakualgoritmeja.

Ratkaisun rajoitteita ovat asiantuntijajärjestelmän monimutkaisuus ja testaamisen haastavuus. Lisäksi optimointi edellyttää taisteluosaston mallintamista laskennan mahdollistavaan muotoon. Tilannekuvajärjestelmien representaatiosta voidaan tuottaa verkkorakenne, jossa taisteluosaston omat yksiköt muodostavat suorituskykyjen solmut suhtautuen havaittuun tilannekuvaan. Olemassa olevia johtamisjärjestelmiä ja näiden ominaisuuksia hyödyntäen voidaan teoriassa luoda nykytilanteeseen vastaava, optimointia tukeva tietorakenne, jonka pohjalle asiantuntijajärjestelmä voidaan rakentaa.

4 Johtopäätökset

4.1 Tutkimustulokset ja systeemimallien yhteenveto



KUVA 24: Systeemimallien yhteenveto

Tarkastelujen pohjalta on kerätty kuvan 24 systeemien systeemi, jossa oleellisimmat tekoälymenetelmät on koottu kategorioittain osaksi päätöksentekosilmukan kokonaisuutta. Yhteenveto kuvaa Suomen laivastojoukkojen mahdollisuuksia tekoälyn käyttöönotolle taisteluosaston näkökulmasta vastaten päätutkimuskysymykseen.

Kuvassa data tilannekuvaelementeistä ja niiden toimintaympäristöstä kulkee systeemin läpi. Systeemi tulkitsee tilannekuvaelementeistä vastaanotetun aaltoliikkeen ja antaa sinne annotaation koneoppimisen menetelmin. Annotoidulle datalle tehdään prediktiot ja siihen liitetään muu saatavilla oleva data. Kun tilannekuvaelementtien merkitys on muodostettu, systeemi laskee todennäköisyydet eri toimintavaihtoehtojen tarpeelle ja suosittaa optimoidun ratkaisun alusosaston toimenpiteille valittua tilannekehitystä varten, jonka päätöksentekijä eli ihminen vahvistaa. Tämän jälkeen systeemin asiantuntijajärjestelmä valitsee tai suosittaa toimenpiteen tilanteeseen vastaamiseksi optimoimalla käytettävissä olevien kykyjen käytön, pyrkimyksenä minimoida resurssien käyttö ja maksimoida vaikutus sekä sen utiliteetti suhteessa riskiin. Tulosten valossa tekoäly on hyödynnettävissä matalalla kynnyksellä tilannekuvaa muodostavan sensoridatan käsittelyyn koneoppimisen ja etenkin syväoppimisen menetelmin, tehostaen olemassa olevia toimintamalleja. Sama voidaan todeta toiminnan tehostamisesta optimointialgoritmeilla. Vastaavasti tilanneymmärrystä ja sen muodostumisen nopeutta kyetään parantamaan koneoppimismallien prediktioilla ja anomaliatarkastelulla sekä probabilistisella lähestymisellä jo lähitulevaisuudessa. Valmius päätöksenteon tukemiseen tekoälyn menetelmin on olemassa, mutta hyödyntäminen edellyttää laajempia toimenpiteitä ja tarkastelua. Johtopäätökset korreloivat muun muassa Lyytisen diplomityön johtopäätösten kanssa [26 s. 90-91].

Autonomisten järjestelmien näkökulmasta voidaan todeta, että järjestelmä, joka kykenee toteuttamaan kyseisen systeemien systeemin toimenpiteet, on määritelmällisesti autonominen. Hankalimpien alasysteemien toteuttaminen voidaan tuottaa ihmisten työpanoksella tai sen tukemana, jotta järjestelmä on robustimpi esimerkiksi päätöksenteon osalta kuin täysautonominen järjestelmä. Etävalvottuna järjestelmä saavuttaa taulukon 1 mukaisesti Lloyds Registerin autonomialuokituksen AL 3 tai AL 4, riippuen päätöksenteon systeemin tasosta. Näin ollen kyky kehittää kuvan 24 alasysteemejä edistää myös autonomisten lavettien kehitystä ja käyttöönottoa.

4.2 Toimenpiteet tekoälyn käyttöönoton mahdollistamiseksi

Tässä luvussa käydään läpi tekoälyn käyttöönoton haasteet Suomen laivastojoukoissa niihin esitettyjen ratkaisujen näkökulmista, vastaten tutkimuksen kolmanteen ja neljänteen alatutkimuskysymykseen. Aiemmissa luvuissa esitettyihin ratkaisuihin on löydetty neljä korkeamman tason toimenpidettä mahdollistamaan näiden toteuttaminen.

Ensisijainen toimenpide tekoälyn käyttöönoton mahdollistamiseksi ja hyödyntämiseksi on asiantuntijuuden lisääminen Merivoimien organisaation sisällä. Yhdysvaltojen periaatepäätös on seurata siviilipuolen kehitystyötä, tutkimusta ja teknologian käyttöönottoa, mikä on tutkimusresurssien näkökulmasta Suomessakin paras lähtökohta. Tekoälyn hyödyntämiselle tulee kuitenkin olla sisäsyntyinen tarve ja ymmärrys, sillä siviilipuolen tutkimus keskittyy niiden ongelmien ympärille, joissa on kaupallista potentiaalia tai yhteiskunnallista merkitystä. Nämä eivät välttämättä korreloi sotilaallisen intressin kanssa. Lisäksi Yhdysvaltojenkin lähestymistapa edellyttää kykyä ja resursseja seurata kehitystä. Tekoälyn ja erityisesti sen mahdollisuuksien ja rajoitteiden hahmottaminen on kriittinen asiantuntijaosaamisen vaatimus organisaation sisällä, sillä missään muualla ei kohdata Suomen laivaston operatiivisen todellisuuden ongelmatilanteita ja kerätä niistä vastaavaa, käyttökelpoista dataa. Tämä osaamisvaatimus liittyy niin nykyisten toimintatapojen tehostamiseen kuin uusien toimintatapojen luomiseen ja suorituskykyjen hankkimiseen, eikä ole havaintona ainutlaatuinen tälle tutkimukselle [122 s. 70-71; 125 s. 18]. Tekoälyosaamisen lisäksi on tärkeää lisätä tekoälyn ymmärrystä, jotta organisaatiolla on realistinen käsitys sen käyttöönotosta ja mahdollisuuksista, eikä muodostu organisatorista vastarintaa esimerkiksi tilannekuvaa muodostavalle tunnistusjärjestelmälle tai autonomisille merellisille järjestelmille.

Toinen kriittinen toimenpide on datamassojen kerääminen ja taltiointi hyödynnettävään muotoon. Tämän tutkimuksen kokeellisten osioiden lähtökohtana on ollut tutkijan näkemys kapeista ongelmista, joita voitaisiin ratkaista tekoälyn avulla. Kokeiden tuottamisessa yhdistyi visio tekoälyn hyödynnettävyydestä sekä tuntemus operatiivisesta taistelualustoiminnasta ja sen tuotteena syntyvistä datamassoista. Datan hyödynnettävyysaste on silti ollut erittäin matala ja datan kerääminen on ollut haastavaa. Tämä ei ole johtunut organisaation vastustuksesta, sillä vastaanotto kaikille tietopyynnöille on ollut erittäin positiivinen ja tutkimuksen tuloksia on ilmaistu odotettavan mielenkiinnolla. Ongelmia tuottavat tiedon taltiointimenetelmät, tietokantojen hajautetut sijainnit ja tiedon sirpaloituminen sekä turvaluokat. Tämä on yhtenevä ongelma Yhdysvaltojen laivaston kanssa, jonka osalta todetaan kriittiseksi tarpeeksi suurien datamassojen koostaminen hyödynnettävään muotoon tutkimus- ja kehitystyön mahdollistamiseksi [22]. Laivastojoukot tuottavat teknisenä aselajina operointinsa sivutuotteena suuria datamassoja, jotka jäävät tekoälyn kannalta toistaiseksi hyödyntämättä. Jatkona asiantuntijuuden lisäämiselle tekoälyn hyödyntämisen saralla on vaatimus muodostaa yhtenäinen tietokantajärjestelmä, jonne voidaan taltioida operatiivista dataa hyödynnettäväksi erilaisiin tutkimus- ja kehityshankkeisiin. Avaintekijöitä tietokantajärjestelmän koostamisessa ovat datan kattavuus, saatavuus ja käytettävyys. Tämä edellyttää järjestelmän lisäksi ohjausta datan kerääämiseen ja taltiointiin. Saatavuudella tarkoitetaan tässä tietokannan yhteyksiä ja verkottuneisuutta, käytettävyydellä taas datan esikäsittelyä ennen taltiointia. Esikäsittely on helpoin toteuttaa dataa kerättäessä esimerkiksi kuva-aineistojen osalta. Kattavuudella tarkoitetaan sitä, että mahdollisimman paljon hyödynnettäviä datamassoja on koottu saman järjestelmän piiriin. Lisäksi datan laajempi hyödynnettävyys edellyttää asiantuntijoiden analyysityön taltiointia osana maalitietoja ja raportteja hyödynnettävissä muodossa. Esimerkiksi nykyisin tilannekuvan maali

luokitellaan johonkin tunnistuskategoriaan, joka ei indikoi suoraan operaattorin tai asiantuntijan maalille tulkitsemaa uhkaavuutta, vaan se tulkitaan erikseen tilanteen kehittyessä. Näin ollen piirreavaruudet jäävät vajaaksi ja tulkinnanvaraisiksi tuotettaessa esimerkiksi todennäköisyyspohjaisia järjestelmiä.

Kolmas toimenpide liittyy turvaluokitellun datan käsittelyyn, tekoälymallien kouluttamiseen ja sen edellyttämiin laitehankintoihin. Kun operatiiviselle datalle on muodostettu tietokanta, on omattava riittävä sisäinen laskentakapasiteetti erilaisten hypoteesien testaamiseksi järkevässä ajassa. Rahallisesti vaatimattomalla panostuksella on hankittavissa nimenomaisesti koneoppimiseen ja syväoppimiseen optimoituja tietokoneita, joissa on useita laskentaan käytettäviä grafiikkakortteja. Tällöin kyettäisiin tuottamaan tehokkaasti erilaisia malleja tietokannan datan pohjalta irrallaan verkkoyhteyksistä ja muualla yleistyneestä pilvilaskennasta, jonka ei voi katsoa soveltuvan operatiivisen datan ja operatiivisten tekoälysovellusten testaamiseen ja tuottamiseen. Vaihtoehtoisesti tämänkaltainen käytännön kehitystyö ja testaaminen voidaan toteuttaa yhteistyönä muiden osapuolten kanssa, mutta tällöin on varmistuttava siitä, että Merivoimien data, sillä tehdyt mallit ja niiden suorituskyky ovat yksinomaan Merivoimien omaisuutta.

Neljäs toimenpide on yhteistyö puolustusteollisuuden ja siviiliyliopistojen kanssa. Inkrementaalinen kehittäminen ja käyttöönotto edellyttää välttämättä yhteistyötä järjestelmien toimittajien sekä uutta teknologiaa kehittävien ja tutkivien tahojen kanssa. Samassa yhteydessä tulisi muodostaa kehitysyhteistyösopimuksia kahden- ja monenvälisin sopimuksin ulkomaisten asevoimien kanssa, kuten Ruotsin ja Yhdysvaltojen kanssa. Perimmäinen edellytys onnistuneelle yhteistyölle muiden osapuolten kanssa on ensin mainittu asiantuntijuus ja kyky hahmottaa oman organisaation kehityskohteita ja mahdollisuuksia niin datamassojen kuin systeemien ja prosessien osalta, sekä kyky artikuloida nämä mahdollisuudet, tavoitteet ja rajoitteet yhteistyöosapuolille. Tekoälylle luonteenomainen, inkrementaalinen kehitystyö, edellyttää pitkäaikaista yhteistyötä, jossa tekoäly tuodaan osaksi järjestelmää ja testataan käytännössä, kehittäen sitä paremmaksi sitä mukaa, kun sen operointi todellisessa käyttöympäristössä osoittaa kehitystarpeita tai ennakoimattomia vinoumia.

Kaikkien neljän toimenpiteen ohjaaminen ja toteuttaminen edellyttää tekoälyn hyödyntämisen pyrkivää konseptia. Toimintaa ohjaavan konseptin laatiminen edellyttää siitä päävastuullista tahoa, joka paitsi laatii ja päivittää konseptin myös johtaa sen tahtotilan jalkauttamisen käytännön toimenpiteiksi ja sitä kautta kehitykseksi. Konseptin tulee määrittää merivoimallinen tahtotila ja painopisteet tekoälyn kehittämiseksi sekä ohjata muun muassa tiedonhallintaa määrittävien aselaji- ja toimialakäskyjen laatimista tekoälyn kehittämiseen tarkoitettujen datamassojen ja tietopohjien koostamiseksi. Ottamatta työn rajauksen puitteissa kantaa puolustusvoimalliseen koordinointiin ja ohjaukseen, konseptissa tulisi myös tunnistaa synergiaedut muiden puolustushaarojen ja yhteistoimintaosapuolten kanssa, lisäten ymmärrystä mahdollisuuksille ja data-aineistojen tarpeille myös Merivoimien ulkopuolella. Esimerkiksi orientaation ja päätöksenteon avustaminen tekoälyn menetelmin sisältää samoja elementtejä pehmeän ja kovan datan yhdistämisestä erilaisiin epävarmuustekijöihin puolustushaarasta riippumatta. Toisaalta esimerkiksi kohteen tunnistamiseen kykenevät mallit voivat hyödyttää niin laivastoyksiköitä kuin ilmavoimien maalinosoitukseen kykeneviä yksiköitä, tukien niin Merivoimien tilannekuvaoperaattorien kuin Ilmavoimien lentäjien tehtävien toteutusta.

4.3 Johtopäätökset ja pohdinta

Tutkimuksessa on pääasiassa tunnistettu prosesseja, joihin on sovitettu niin kvalitatiivisesti kuin kvantitatiivisesti tekoälyn menetelmiä näiden prosessien toteuttamisen tehostamiseksi. On kuitenkin huomionarvoista, että tekoälyn hyödyntämisen toinen lähtökohta on luoda kokonaan uusia toimintatapoja. Tutkimuksen tärkein havainto uusien toimintatapojen luomisesta liittyy kompleksisiin peleihin ja simulaatioihin ja niillä mahdollisesti luotaviin täysin uudenlaisiin taktisiin toimenpiteisiin ja päätöksentekoon. Tämän ohella autonomiset järjestelmät mahdollistavat toisen kokonaisuuden osin tai täysin uusille toimintatavoille nykyisten toimintatapojen tehostamisen ohella.

Tutkimuksen tulokset erilaisten tekoälymallien soveltuvuudesta eri tehtäväkategorioihin ovat lähtökohtia jatkotutkimukselle ja esitettyjen hypoteesien testaukselle niiden todellisen toimivuuden varmistamiseksi ja kokeelliseksi opponoinniksi. Tutkimus ei ole tuottanut eksplisiittisiä ratkaisuja esitettyihin ongelmiin, vaan esittänyt yhden tavan kartoittaa tekoälyn hyödynnettävyyttä ja kehitystä laivastojoukkojen kontekstissa. Käytetty pehmeä systeemimetodologia vaikuttaa tämän tutkimuksen valossa soveltuvan tekoälyn hyödyntämismahdollisuuksien kartoittamiseen Suomen laivastojoukkojen kontekstissa. Tutkimustuloksia ei kuitenkaan tule käsitellä käyttöönottomalleina tai muina konkreettisina ratkaisuina, vaan tapana jäsennellä Merivoimien mahdollisuuksia tekoälyn kehityksen seuraamiselle ja hyödyntämiselle. Tutkimuksen tulokset voivat toimia yhtenä lähtökohtana merivoimallisen tekoälykonseptin luomiselle. Liitteissä raportoidut kokeet validoivat osan tutkimuksen tuloksista, sitoen ne konkreettisesti Merivoimien nykytilaan muun muassa data-aineistojen ja niiden haasteiden kautta. Tutkimuksen havainnot tekoälyn käyttöönoton haasteista ja niiden asettamista vaatimuksista ovat universaaleja ja korreloivat muiden tutkimusraporttien ja alan kirjallisuuden havaintoihin saman aihepiirin ongelmista.

Tekoälyn käyttöönoton haasteet liittyvät pääasiassa datan puutteeseen, sen laatuun tai käytettävyyteen, sekä asiantuntijuuteen tekoälyn hyödyntämisessä. Lisäksi käyttöönoton tekninen haaste on siinä, ettei tekoälymallia voida ottaa käyttöön operatiivisessa järjestelmässä ilman järjestelmäintegraatiota järjestelmätoimittajan puolelta, vaikka malli voitaisiinkin toteuttaa organisaation sisäisin toimenpitein. Mikäli Merivoimat toteuttavat edellä mainitut toimenpiteet tekoälyn hyödyntämisen mahdollistamiseksi, tullaan samalla luomaan pohja autonomisten järjestelmien konfiguroimiseksi Itämeren olosuhteisiin. Vaikka Suomessa ei kehitettäisi autonomisia merellisiä järjestelmiä, edellyttäisi hankittavien järjestelmien käyttöönotto samojen systeemien toimintaympäristöön sovitettuja tekoälyn menetelmiä ja kykyjä kuin kuvassa 24 on esitetty. Näin ollen yksittäisten, operatiiviseen dataan perustuvien tietokantojen ja mallien kehittämisen voidaan nähdä valmistavan Suomen laivastojoukkoja autonomisten järjestelmien käyttöönottoon nykyisen toiminnan tehostamisen ohella.

Suomen laivastojoukot muodostavat systeemien systeemin, joka kykenee edistämään tekoälyn käyttöönottoa organisaation eri tasoilla nopeallakin aikataululla, edellyttäen, että nykyisen operoinnin ohella huomioidaan datakeskeisyys ja panostetaan hyödyllisten datamassojen taltiointiin, asiantuntijajärjestelmien muodostamiseen ja näiden hyödynnettävyyden kehittämiseen. Poikkitieteellisyytensä takia tekoälyn ei tulisi olla jonkin tietyn toimialan koordinoitavissa, vaan omata kyky toimialat ja aselajit läpäisevään koordinointiin mahdollisimman monipuolisen ja laaja-alaisen hyödyntämisen mahdollistamiseksi. Näin ollen tämän tutkimuksen tärkein johtopäätös on, että Merivoimat tarvitsevat tekoälyn ja autonomisten järjestelmien konseptin, joka määrittää vastuualueet edellisessä alaluvussa listattujen toimenpiteiden toteuttamiseksi sekä ohjaa tekoälyn hyödyntämiseen tähtäävien projektien ja hankkeiden käynnistämistä.

LÄHTEET

[1] Valtioneuvoston julkaisuja 2021:78, *Valtioneuvoston puolustusselonteko*, Valtioneuvosto, Helsinki, 2021, saatavilla: julkaisut.valtioneuvosto.fi

[2] Ministry of Defence, *Strategic Guidelines for AI Solutions*, Finland 2020, saatavilla: julkaisut.valtioneuvosto.fi

[3] Suojanen M., Miettinen M., Helle S., Kalliohaka T., Kallinen K., *AI-ROADMAP Tutkimuskokonaisuuden Mitä AI on? loppuraportti,* Puolustusvoimien Tutkimuslaitos, 18.9.2018, Riihimäki, KÄYTTÖ RAJOITETTU STIV

[4] Pääesikunnan johto, *Tekoälyn tiekartta (Artificial Intelligence Roadmap) – Versio 1*, Suunnitelma, diaari AO16025, 20.9.2018, Helsinki, AO16025, KÄYTTÖ RAJOITETTU STIV

[5] Laivue 2020 Puolustusvoimien strateginen hanke, Puolustusministeriö, Lönnberg 2017, ISBN 978-951-25-2879-0

[6] L 11.5.2007/551 Laki Puolustusvoimista 2§, 3§

[7] *Merivoimat – turvanamme 24/7*, Merivoimat, viitattu 24.1.2021, saatavilla: https://merivoimat.fi/tietoa-meista

[8] Rantapelkonen J. (toim), Brunberg J., "*Tuleva Sota"*, Maanpuolustuskorkeakoulu, Sotataidon laitos, julkaisusarja 2: Tutkimusselosteita nro 5, Otavan Kirjapaino Oy, Keuruu 2018, 270 s., ISSN 2342-5283

[9] *Finferries' Falco world's first fully autonomous ferry*, Finferries Press releases, 03.12.2018, viitattu 16.9.2020, saatavilla: https://www.finferries.fi/en/news/press-releases/finferries-falco-worlds-first-fully-autonomous-ferry

[10] Puolustusministeriö, *Puolustusvoimille sata uutta virkaa*, puolustusministeriön tiedote, 29.8.2018, viitattu 5.12.2021, saatavilla: https://valtioneuvosto.fi/-/puolustusvoimille-sata-uutta-virkaa

[11]*SKM-MERIVE Puolustusvoimien alusten merenkulku- ja konepäällystön vähimmäismäärä- ja vähimmäispätevyysvaatimukset*, Merivoimien esikunta, operatiivinen osasto, sotilaskäsky HL834, 10.8.2015, KÄYTTÖ RAJOITETTU STIV

[12] Chapman P., Clinton J., Kerber R., Khabaza T., Reinartz T., Shearer C., Rüdiger W., *CRISP-DM 1.0 Step-by-step data mining guide*, CRISP-DM consortium, 2000, viitattu 14.2.2021, saatavilla: ftp://ftp.software.ibm.com/software/analytics/spss/Modeler/Documentation/14/UserManual/CRISP-DM.pdf

[13] Checkland P. "*Systems Thinking, Systems Practice"*, John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ Englanti UK, 1999, 424 s., ISBN-13 978-0-471-98606-5

[14] Wilson B., *Soft Systems Methodology: Conceptual Model Building and its Contribution*, John Wiley & Sons Ltd, Baffins Lane, Chichester, West Sussex PO19 1UD UK, 2001, 250 s., ISBN 0-471-89489-3

[15] Blaha L.M., *Interactive OODA Processes for Operational Joint Human-Machine-Intelligence*, Pacific Northwest National Laboratory, Richland, Washington, USA, 18.7.2018, STO-MP-IST-160, DOI 10.14339/STO-MP-IST-160-PP-3P-PDF

[16] Jokioinen E., Poikonen J., Hyvönen M., Kolu A., Jokela T., Tissari J., Paasio A., Ringbom H., Collin F., Viljanen M., Jalonen R., Tuominen R., Wahström M., Saarni J., Nordberg-Davies S., Makkonen H.,*Remote and Autonomous Ships: The next steps*, AAWA Position Paper, Rolls-Royce plc, 2016, 62 Buckingham Gate, Lontoo, Englanti, viitattu 2.4.2021, saatavilla: https://research.utu.fi/converis/portal/Publication/18550958?auxfun=&lang=en_GB

[17] Heiskanen T., *Aalto-yliopisto ja Saab laajentavat strategista tutkimusyhteistyötä*, 6.9.2021, Aalto-yliopisto, viitattu 9.9.2021, saatavilla: https://www.aalto.fi/uutiset/aalto-yliopisto-ja-saab-laajentavat-strategista-tutkimusyhteistyota

[18] Webster G., Creemers R., Triolo P., Kania E., *China's Plan to 'Lead' in AI: Purpose, Prospects, and Problems*, 1.8.2018, New America, 740 15th Street NW, Suite 900, Washington, DC 20005, viitattu 16.4.2021, saatavilla: https://www.newamerica.org/cybersecurity-initiative/blog/chinas-plan-lead-ai-purpose-prospects-and-problems/

[19] O'Rourke R., *Navy Large Unmanned Surface and Undersea Vehicles: Background and Issues for Congress*, R45757, 8.8.2020, Congressional Research Service, viitattu 17.4.2021, saatavilla: https://crsreports.congress.gov

[20] Martin B., Tarraf D.C., Whitmore T.C., DeWeese J., Kenney C., Schmid J., DeLuca P., *Advancing Autonomous Systems: An Analysis of Current and Future Technology for Unmanned Maritime Vehicles*, RAND Corporation, Santa Monica, 2019

[21] *Navy Center for Applied Research in Artificial Intelligence (NCARAI)*, viitattu 27.1.2022, saatavilla: https://www.nrl.navy.mil/itd/aic/

[22] Tangredi J.S., Galdorisi G., *AI at War: how Big Data, artificial intelligence, and machine learning are changing naval warfare*, 2021, Naval Institute Press, Annapolis, Maryland, 464 s., ISBN 978-1-682-476-062

[23] Sharma S., Subudhi B., *Navigation and Control of Autonomous Marine Vehicle*s, the Institution of Engineering and Technology, Croydon UK, 2019, 333 s., ISBN 978-1-78561-338-8

[24] Eerola L., *Autonomisten maataistelurobottien torjunta pataljoonan taistelussa 2030-luvulla*, pro gradu tutkielma, Helsinki, huhtikuu 2020, Maanpuolustuskorkeakoulu, Sotataidon laitos, KÄYTTÖ RAJOITETTU TLIV, 129 sivua

[25] Havu E., *Tekoälyn vaikutus Venäjän sotataitoon,* yleisesikuntaupseerikurssin diplomityö, Helsinki, elokuu 2021, Maanpuolustuskorkeakoulu, Sotataidon laitos, strategia, 91 sivua

[26] Lyytinen T., *Koneoppimisen hyödyntäminen ilmaoperaation suunnittelussa*, yleisesikuntaupseerikurssin diplomityö, Helsinki, elokuu 2021, Maanpuolustuskorkeakoulu, Sotatekniikan laitos, KÄYTTÖ RAJOITETTU TLIV, 107 sivua

[27] Kilpeläinen L., *Tekoälyavustettu taistelunjohto*, pro gradu -tutkielma, Helsinki, huhtikuu 2020, Maanpuolustuskorkeakoulu, Sotataidon laitos, KÄYTTÖ RAJOITETTU TLIV, 69 sivua

[28] Timoskainen M., *Vedenalaisen autonomisen järjestelmän vaikutus aluksen turvallisuuteen*, pro gradu tutkielma, Helsinki, huhtikuu 2017, Maanpuolustuskorkeakoulu, Sotataidon laitos, STIV, 101 sivua

[29] Röberg B., *Sodan oikeussäännöt ja merelliset miehittämättömät järjestelmät*, yleisesikuntaupseerikurssin diplomityö, Helsinki 2019, Maanpuolustuskorkeakoulu, Johtamisen ja sotilaspedagogiikan laitos

[30] Kuusisto R., Metsola T., *Selvitysraportti tekoälyn maturiteetista*, Puolustusvoimien tutkimuslaitos, Informaatiotekniikkaosasto, 29.5.2018, Riihimäki, AO10213, KÄYTTÖ RAJOITETTU TLIV

[31] Russel S., Norvig P., *Artificial Intelligence – A Modern Approach, Third Edition,* Pearson Education Limited, Harlow, Englanti, 2016, 1132 s., ISBN: 978-1292-1453-96-4

[32] Ertel W., *Introduction to Artificial Intelligence*, Second Edition, Springer, Ravensburgh, Saksa, maaliskuu 2017, 356 s., ISBN 978-3-319-58487-4

[33] Neapolitan R.E., Jiang X., *Artificial Intelligence with an Introduction to Machine Learning Second Edition*, Taylor & Francis Group LLC, Boca Raton USA, 2018, 480 s., ISBN 978-1-138-50238-3

[34] Jung A., *Machine Learning: Basic Principles*, Aalto yliopisto, Espoo, arXiv:1805.05052v12 [cs.LG] 21.9.2020

[35] Mohri M., Rostamizadesh A., Talwalkar A., *Foundations of Machine Learning, second edition*, 2018, the MIT Press, Cambridge, Massaschusetts, London, England, 505 s., ISBN 978-0-2620-3940-6

[36] Goodfellow I., Bengio Y., Courville A., *Deep Learning*, The MIT Press, Cambridge, Massachusetts USA, 2016, 775 s., www.deeplearning.org, ISBN 978-0-262-03561-3

[37] Poole D., Mackworth A., Goebel R., *Computational Intelligence: A Logical Approach*, New York: Oxford University Press, 1998, ISBN 978-0-19-510270-3

[38] Kielitoimiston sanakirja, "*keino*", Kotimaisten kielten keskus ja Kielikone Oy, viitattu 18.4.2021, saatavilla: https://www.kielitoimistonsanakirja.fi/#/keino

[39] Kielitoimiston sanakirja, "*teko*", Kotimaisten kielten keskus ja Kielikone Oy, viitattu 18.4.2021, saatavilla: https://www.kielitoimistonsanakirja.fi/#/teko

[40] McCarthy J., *What is Artificial Intelligence?,* Computer Science Department, Stanford University, Stanford, CA 94305, 12.11.2007, viitattu 7.10.2020, saatavilla: http://www-formal.stanford.edu/jmc/whatisai.pdf

[41] Rich E., *Artificial Intelligence*, McGraw-Hill Series in Artificial Intelligence, McGraw-Hill Inc, New York USA, 1983, 436 s., ISBN 0-07-052261-8

[42] Tegmark M., *LIFE 3.0: Being Human in the Age of Artificial Intelligence,* Vintage Books, New York 2017, 364 s., ISBN 978-1-101-94660-2

[43] Scharre P., *Army of None*, W.W. Norton & Company, New York USA, 2018, 446 s., ISBN 978-0-393-35658-8 pbk

[44] Newell A., Simon H.A., *Computer Science as Empirical Inquiry: Symbols and Search,* ACM Turing Award Lecture, Communications of the ACM, volume 19, number 3, maaliskuu 1976

[45] Harnad S., *The Symbol Grounding Problem,* Department of Psychology, Princeton University, Princeton, NJ 08544 USA, Physica D 42 (1990) pp. 335-346, 1990, viitattu 23.2.2021, saatavilla: https//eprints.soton.ac.uk/250382/1/symgro.pdf

[46] Nilsson N.J., *The Physical Symbol System Hypothesis: Status and Prospects,* Stanford Artificial Intelligence Laboratory, Stanford University, julkaistu M. Lungarella ym, "*50 Years of AI, Festschrift",* LNAI 4850, s. 9-17, Springer 2007, viitattu 5.4.2021, saatavilla: ai.stanford.edu/users/nilsson/OnlinePubs-Nils/PublishedPapers/pssh.pdf

[47] Ben-Gal I., *Bayesian Networks,* Ruggeri F., Faltin F. & Kennelt R. Encyclopedia of Statistics in Quality and Reliability, Wiley & Sons 2007, 15.3.2008

[48] Griffiths T.L., Yuille A., *Technical Introduction: A primer on probabilistic inference*, UCLA Department of Statistics Papers, 25.10.2011, viitattu 8.1.2020, saatavilla: https//scholarship.org/uc/item/96w8s0rg

[49] Laaksonen A., *Tietorakenteet ja algoritmit*, Helsingin Yliopisto, Helsinki, 2.1.2020, saatavilla: https://www.cs.helsinki.fi/u/ahslaaks/tirakirja/

[50] Lighthill J., *Part I Artificial Intelligence: A general survey*, Cambridgen yliopisto, heinäkuu 1972. Viitattu 15.4.2021, saatavilla: www.chilton-computing.org.uk/inf/literature/reports/lighthill_report/p001.htm

[51] Simon A., Deo M. S., Venkatesan S., Babu R., *An Overview of Machine Learning and its Applications*, IJESE, Volume 1: 2015, Dayananda Sagar College of Engineering, Bengaluru-78

[52] Ross S.M., *Introductory Statistics*, Academic Press, 125 London Wall, Lontoo, 2017, 828 s., ISBN 978-0-12-804317-2

[53] Winstead P.J., *Implementation of Unmanned Surface Vehicles in the Distributed Maritime Operations Concept*, 01.12.2018, Technical Report, Naval Postgraduate School, Monterey USA, viitattu 7.2022, saatavilla: https://apps.dtic.mil/sti/citations/AD1069834

[54] Python Scikit learn dokumentointi 1.10. Decision Trees, viitattu 1.8.2021, saatavilla: https://scikit-learn.org/stable/modules/tree.html

[55] Åström K.J., Murray R.M., *Feedback Systems*, versio v2.11b, Princeton University Press, Princeton and Oxford, 28.8.2012, viitattu 27.2.2021, saatavilla: http://press.princeton.edu/titles/8701.html

[56] Hornik K., Stinchcombe M., White H., *Multilayer Feedforward Networks are Universal Approximators,* Neural Networks Vol. 2 pp. 359-366, Pergamon Press plc, 9.3.1989

[57] Karras Tero, Laine Samuli, Aila Timo, *A Style-Based Generator Architecture for Generative Adversarial Networks*, 6 Feb 2019, arXiv:1812.04948v2 [cs.NE]

[58] Duignan B., "*Occam's razor"*, Encyclopedia Britannica, saatavilla: https://www.britannica.com/topic/Occams-razor, viitattu 8.3.2021.

[59] Masters D., Luschi C., *Revisiting Small Batch Training for Deep Neural Networks*, 20.4.2018, saatavilla: https://arxiv.org/abs/1804.07612

[60] Hayes B., *Cloud Computing*, Communications of the ACM, Volume 51, Number 7 (2008), p. 9-11, Durham NC, DOI: http://doi.acm.org/10.1145/1364782.1364786

[61] Biamonte J., Wittek P., Pancotti N. ym., *Quantum machine learning*, Nature 549, 195-202 (2017), viitattu 12.1.2022, saatavilla: https://doi.org/10.1038/nature23474

[62] Xuelong L., Kai K., Bin Z., *Weather GAN: Multi-Domain Weather Translation Using Generative Adversarial Networks*, Computer Vision and Pattern Recognition (cs.CV), 9.3.2021, cited as arXiv.2103.05422, available at: https://arxiv.org/abs/2103.05422

[63] Loyola-González O., *Black-Box vs. White-Box: Understanding Their Advantages and Weaknesses from a Practical Point of View*, IEEE Access, vol. 7, pp. 154096-154113, 2019, doi: 10.1109/ACCESS.2019.2949286.

[64] Kainulainen J., *Kädet irti ratista: Honda julkisti auton, joka nostaa autonomisen ajamisen uudelle tasolle*, Kauppalehti 8.3.2021, viitattu 10.3.2021, saatavilla: https://www.kauppalehti.fi/uutiset/kadet-irti-ratista-honda-julkisti-auton-joka-nostaa-autonomisen-ajamisen-uudelle-tasolle/6b4c8294-1a32-4d42-b27c-fbf0eb1ffe90

[65] Goodfellow I. J., Shlens J., Szegedy C., *Explaining and Harnessing Adversarial Examples*, Google Inc, Mountain View California, ICLR 20.3.2015, viitattu 1.3.2021, saatavilla: https://arxiv.org/pdf/1412.6572.pdf

[66] H. Kwon, Y. Kim, H. Yoon and D. Choi, *Fooling a Neural Network in Military Environments: Random Untargeted Adversarial Example,* MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM), 2018, pp. 456-461, doi: 10.1109/MILCOM.2018.8599707.

[67] Kurakin A., Goodfellow I.J., Bengio S., *Adversarial Examples in the Physical World,* Workshop track - ICLR 11.2.2017, viitattu 22.4.2021, saatavilla: https://arxiv.org/pdf/1607.02533.pdf

[68] Shukla M., *Organizations Must Address These Five Challenges When Adopting AI and Automation*, Forbes, 8.7.2019, viitattu 22.4.2021, saatavilla: https://www.forbes.com/sites/forbestechcouncil/2019/07/08/organizations-must-address-these-five-challenges-when-adopting-ai-and-automation/

[69] Hemminki P., *New Brains For the Defence System*, pro gradu – tutkielma, Turun yliopisto, luonnontieteiden tiedekunta, tietojenkäsittelytieteiden laitos, 2020, 54 sivua

[70] Merriam-Webster Dictionary, "*automation*", viitattu 22.2.2021, saatavilla: https://www.merriam-webster.com/dictionary/automation

[71] IEEE Corporate Advisory Group (CAG), *IEEE Guide for Terms and Concepts in Intelligent Process Automation*, IEEE Standards Association, IEEE Std 2755-2017, 3 Park Avenue, 10016-5997, New York USA, 2017

[72] Merriam-Webster Dictionary, *"robotics",* viitattu 22.2.2021, saatavilla: https://www.merriam-webster.com/dictionary/robotics#other-words

[73] Merriam-Webster Dictionary, "robot", viitattu 22.2.2021, saatavilla: https://www.merriam-webster.com/dictionary/robots

[74] Kurfess, T.R., *Robotics and automation handbook*, Library of Congress Cataloging-in-Publication Data, CRC Press LLC, 2005, 602 s., ISBN 0-8493-1804-1, TJ211.R5573 2000

[75] Kauhanen A, Maliranta M, Rouvinen P., Vihriälä V, *TYÖN MURROS – Riittääkö dynamiikka?*, Elinkeinoelämän tutkimuslaitos ETLA, Taloustieto Oy, Helsinki 2015, ISBN 978-951-628-646-7

[76] P. Dario, E. Guglielmelli, B. Allotta and M. C. Carrozza, *Robotics for medical applications*, in IEEE Robotics & Automation Magazine, vol. 3, no. 3, pp. 44-56, Sept. 1996, doi: 10.1109/100.540149

[77] Yenne B., *Attack of the Drones: A History of Unmanned Aerial Combat*, Zenith Press, 380 Jackson Street, St. Paul, MN, USA, 2004, 127 s., ISBN 0-7603-1825-5

[78] Appelqvist P., *Järjestelmien autonomisista piirteistä sotilassovelluksissa*, tutkimusseminaari 18.11.2014, MATINE, Puolustusministeriö, viitattu 5.1.2021, saatavilla: https://www.defmin.fi/files/2978/Appelqvist_MATINE_TS2014-web.pdf

[79] Cares J.R., Dickmann J.Q. Jr, *Operations Research for Unmanned Systems*, 2016, First Edition, John Wiley & Sons Ltd, ISBN: 978-1-118-91894-4, 328 sivua

[80] Lloyd's Register, LR Code for Unmanned Marine Systems, Lloyd's Register Group Limited, 71 Fenchurch Street, Lontoo, helmikuu 2017

[81] Jokiranta V., *The Shipmaster in Remote & Autonomous Operations*, 505914, Kansainvälisen oikeuden merkitys globalisaation aikakaudella, pro gradu tutkielma, Turun yliopisto, oikeustieteellinen tiedekunta, 6.5.2019, 84 sivua

[82] Burmeister H-C, *Final Report Summary – MUNIN (Maritime Unmanned Navigation through Intelligence in Networks)*, European Commission, Fraunhofer Center for Maritime Logistics and Services, Am Schwarzenberg-Campus 4 D, Hampuri, Saksa, viitattu 13.5.2021, saatavilla: https://cordis.europa.eu/project/id/314286/reporting

[83] Department of Defense, *DIRECTIVE NUMBER 3000.09: Autonomy in Weapon Systems*, United States of America Department of Defense, 21.11.2012, viitattu 13.5.2021, saatavilla: https://www.esd.whs.mil/Directives/issuances/dodd/

[84] Boyd J.R., Hammond G.T., *A Discourse on Winning and Losing*, Air University Press, Curtis E. LeMay Center for Doctrine Development and Education, Maxwell AFB, Alabama, USA, maaliskuu 2018, ISBN 978-585-566-279-1, 400 sivua

[85] Jaiswal N.K., *Military Operations Research: Quantitative Decision Making,* Springer, 1997, https://doi.org/10.1007/978-1-4615-6275-7, 388 sivua

[86] Ikonen I., *Operaatioanalyysin käsite sotilaallisessa kontekstissa,* diplomityö, Helsinki, elokuu 2017, Maanpuolustuskorkeakoulu, Sotataidon laitos, tekstisivuja 108

[87] Hiller F.S., Lieberman G.J., *Introduction to Operations Research,* McGraw-Hill, New York, 2001, ISBN 9781259872990

[88] Smith J.W.E., *Sir Julian Corbett's Relevance to the Modern Strategic Thinker*, 21.8.2018, The Strategy Bridge, viitattu 24.1.2021, saatavilla: https://thestrategybridge.org/the-bridge/2018/8/21/corbetts-relevance-to-the-modern-strategic-thinker

[89] Corbett J., *Some Principles of Maritime Strategy*, Lightning Source UK Ltd, Milton Keynes, Iso-Britannia, helmikuu 2009, 512 s., ISBN 978-1-4099-6419-3

[90] Lundqvist S., *Continuity and Change in post-Cold War Maritime Security: A Study of the Strategies Pursued by the US, Sweden and Finland 1991-2016*, väitöskirja, Vaasa 2017, Åbo Akademi, poliittisten tieteiden tiedekunta, Painosalama Oy Turku, 2017, ISBN 978-952-12-3602-7, tekstisivuja 265

[91] Tulli, Ulkomaankaupan kuljetukset 2019, 12.3.2020, Helsinki, viitattu 24.1.2021, saatavilla: https://tulli.fi/documents/2912305/3494771/Ulkomaankaupan+kuljetukset+vuonna+2019#

[92] Flemming N.C., Harff J., Moura D., Burgess A., Bailey G.N., *Submerged Landscapes of the European Continental Shelf: Quaternary Paleoenvironments*, John Wiley & Sons Ltd, 2017, Oxford, Iso-Britannia, 523 s., ISBN 978-111-892-771-7

[93] Tuomi L., Kahma K.k., Pettersson H., *Wave hindcast statistics in the seasonally ice-covered Baltic Sea*, Marine Research, Ilmatieteenlaitos, 30.12.2011, Helsinki, ISSN 1797-2469

[94] Vego M., *On Littoral Warfare*, Naval War College Review: Vol.68 : No 2, Article 4, 2015, viitattu 29.1.2021, saatavilla: https://digital-commons.usnwc.edu/nwc-review/vol68/iss2/4

[95] Ilmatieteenlaitos, *Jäätalvi Itämerellä*, verkkojulkaisu, viitattu 29.1.2021, saatavilla https://www.ilmatieteenlaitos.fi/jaatalvi-itamerella

[96] Pesu M., *Hard security dynamics in the Baltic Sea region: From turbulence to tense stability*, FIIA briefing paper 276, 1.2.2020

[97] Kuusisto R., Armistead L., Kuusisto T.; *Common Operational Picture, Situation Awareness and Information Operations*, Proc. of the 4th European Conference on Information Warfare and Security, Glamorgan, UK, 2005

[98] Endsley M.R., *Toward a Theory of Situation Awareness in Dynamic Systems*, Human Factors The Journal of the Human Factors and Ergonomics Society, 37(1), maaliskuu 1995, DOI: 10.1518/001872095779049543

[99] Royal Netherlands Navy, *Fundamentals of Maritime Operations, Netherlands maritime military doctrine*, Netherlands Ministry of Defence, Royal Netherlands Navy, Directorate of Operations, Maritime Warfare Centre, Rjikszee-en Marinehaven, 1780 CA Den Helder, 2014, X-Media, Defence Media Centre

[100] United States Marine Corps Headquarters, *Intelligence Preparation of the Battlefield/Battlespace*, ATP 2-01.3 MCRP 2-10B-1, marraskuu 2014, viitattu 6.7.2021, saatavilla: https://www.marines.mil/portals/1/MCRP-2-10B.1.pdf?ver=2018-10-04-131000-610

[101] Payne C., *Principles of Naval Weapon Systems*, Second edition, Naval Institute Press, Annapolis, Maryland USA, 2010, 412 s., ISBN 978-1-59114-667-4

[102] Esteban J., Starr A., Willetts R., Hannah P., Bryanston-Cross P., *A Review of Data Fusion Models and Architectures: Towards Engineering Guidelines*, 2005, Neural Computing and Applications, 14, DOI: 10.1007/s00521-004-0463-7

[103] Hu G., Wang K., Liu L., *Underwater Acoustic Target Recognition Based on Depthwise Separable Convolution Neural Networks*, 18.2.2021, Sensors, 2021, 21, 1429, viitattu 5.8.2021, saatavilla: https://doi.org/10.3390/s21041429

[104] Wang D., Wang Y., *The Classification of Underwater Acoustic Targets Based on Deep Learning Methods*, tammikuu 2017, Conference Paper, 10.2991/caai-17.2017.118

[105] Pham Q.-V., Nguyen T., Huynh-The T., Bao L., Lee K., Hwang W.-J., *Intelligent Radio Signal Processing: A Survey*, IEEE Access vol. 9, p. 83818-83850, 2021, doi: 10.1109/ACCESS.2021.3087136

[106] Hu H., Wang Y., Song J., *Signal Classification Based on Spectral Correlation Analysis and SVM in Cognitive Radio*, 22nd International Conference on Advanced Information Networking and Applications (aina 2008), 2008, p. 883-887, doi: 10.1109/AINA.2008.27

[107] Brodeski D., Bilik I., Giryes R., *Deep Radar Detector*, 2019 IEEE Radar Conference (RadarConf), 2019, doi: 10.1109/RADAR.2019.8835792.

[108] Shiwen C., Gongming W., Xiaopeng X., Jie H.; *A Method of Radar Signal Feature Extraction Based on Fractional Fourier Transform*, 2019, IEEE 4th International Conference on Signal and Image Processing (ICSIP), 2019, doi: 10.1109/SIPROCESS.2019.8868749.

[109] Neumann C., Brosch T., *Deep Learning Approach for Radar Applications*, Warsaw University of Technology, IPS 2020, Varsova, Puola, 978-83-949421-5-1

[110] Chen C. V., Li F., Ho S-S., Wechsler H., *Micro-Doppler Effect in Radar: Phenomenon, Model, and Simulation Study*, IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS VOL. 42, NO. 1 JANUARY 2006, IEEE Log No. T-AES/42/1/870577

[111] Noyes S.P., *Track classification in a naval defence radar using fuzzy logic*, IEE Colloquium on Target Tracking and Data Fusion (Digest No. 1998/282), 1998, pp. 5/1-5/6, doi: 10.1049/ic:19980423.

[112] Elmenreich, W., *Sensor Fusion in Time-Triggered Systems*, väitöskirja, lokakuu 2002, Wienin yliopisto, kokonaissivumäärä 157

[113] Llinas J., Hall D.L., *An Introduction to Multi-Sensor Data Fusion*, ISCAS 98', Proceedings of the 1998 IEEE International Symposium on Circuits and Systems (Cat. No. 98CH361787), 1998, vol. 6, doi: 10.1109/ISCAS.1998.705329

[114] Farahnakian F., Habhbayan M-H., Poikonen J., Laurinen M., Nevalainen P., Heikkonen J., *Object Detection based on Multi-sensor Proposal Fusion in Maritime Environment*, 17th IEEE International Conference on Machine Learning and Applications (ICMLA), 2018, p. 971-976, doi: 10.1109/ICMLA.2018.00158

[115] Yu Y., Lee H. J., Kim B. C., Kim J. U., Ro Y. M., *Towards Robust Training of Multi-Sensor Data Fusion Network Against Adversarial Examples in Semantic Segmentation*, ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2021, pp. 4710-4714, doi: 10.1109/ICASSP39728.2021.9413772.

[116] Virjonen P, Nevalainen P, Pahikkala T., Heikkonen J., *Ship movement Prediction Using k-NN Method*, 2018 Baltic Geodetic Congress (BGC Geomatics), 2018, doi: 10.1109/BGC-Geomatics.2018.00064

[117] Gan S., Liang S., Li K., Deng J., Cheng T., *Ship trajectory prediction for intelligent traffic management using clustering and ANN*, 2016 UKVACC 11th International Conference on Control (CONTROL), 2016, doi: 10.1109/CONTROL.2016.7737569

[118] Sun Tzu, *The Art of War*, 2014, Arcturus Publishing Limited 26/27 Bickels Yard, 151-153 Bermondsey Street, London, ISBN 978-1-78303-202-8

[119] Tromp J., Farnebäck G., *Combinatorics of Go*, Conference: Computers and Games, 5th International Conference, CG 2006, Turin, Italia, 29-31, 2006, DOI:10.1007/078-3-540-75538-8_8

[120] Mnih V., Kavukcuoglu K., Silver D., Graves A., Antonoglou I., Wierstra D., Riedmiller M., *Playing Atari with deep reinforcement learning*, NIPS Deep Learning Workshop, 2013, viitattu 7.9.2021, saatavilla: https://arxiv.org/abs/1312.5602

[121] Vilnyals O., Babuschkin I., Czarnecki W., Mathie M., Dudzik A., Chung J., Choi D.H., Powell R., Ewalds T., Georgiev P., Oh J, Horgan D., Kroiss M., Danihelka I., Huang A., Sifre L., Cai t., Agapiou J.P., Jaderberg M., Vezhnevets A.S., Leblond R., Pohlen T., Dalibard V., Budden D., Sulsky Y., Molloy J., Paine T.L., Gulcehre C., Wang Z., Smith O., Schaul T., Lillicrap T., Kavukcuoglu K., Hassabis D., Apps C., Silver D., *Grandmaster level in StarCraft II using multi-agent reinforcement learning*, Nature Vol 575, 14.11.2019, https://doi.org/10.1038/s41586-019-1724-z

[122] Boulanin V., Verbruggen M., *Mapping the Development of Autonomy in Weapon Systems*, SIPRI, marraskuu 2017, viitattu 11.3.2022, saatavilla: https://www.sipri.org/publications/2017/other-publications/mapping-development-autonomy-weapon-systems

[123] Jin S., Zhou M., Wu A.S., *Sensor Network Optimization Using a Genetic Algorithm*, Proceedings of the 7th world multiconference on systemics, cybernetics and informatics, 2003

[124] Buczak A.L., Jin Y., Darabi H., Jafari M., *Genetic algorithm based sensor network optimization for target tracking*, Intelligent Engineering Systems through Artificial Neural Networks, 1999, 9

[125] Mukherjee T., *Securing the maritime commons: The role of artificial intelligence in naval operations*, ORF Observer Research Foundation, Occasional Papers, 16.7.2018, viitattu 11.3.2022, saatavilla: https://www.orfonline.org/research/42497-a-i-in-naval-operations-exploring-possibilities-debating-ethics/

Koneoppimismallit ja niiden visualisointi

Tutkielman luvussa 2 käytetty data ja siitä luodut mallit sekä visualisoinnit on toteutettu simuloimalla ilmamaalidataa tämän liitteen Python 3 – ohjelmalla. Ensimmäisessä koodiosassa on suurin osa käytetyistä riippuvaisuuksista eri kirjastojen muodossa.

```
#!/usr/bin/env python3

import numpy as np
import matplotlib.pyplot as plt
import statsmodels
import statsmodels.api as sm
import statsmodels.formula.api as smf
from statsmodels.graphics.regressionplots import abline_plot
import pandas as pd
from numpy.random import randint
import random
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn import preprocessing
from sklearn.linear_model import LinearRegression
from mpl_toolkits.mplot3d import Axes3D
from sklearn.linear_model import LogisticRegression
from collections import Counter

random.seed(42)
```

Ensimmäistä kuvaa varten visualisoitiin sota-alusten piirredataa kolmiulotteisessa avaruudessa, jotta käsitys datan piirreavaruuksista saadaan konkretisoitua. Tähän käytettiin Winstead P.J. opinnäytetyöhönsä keräämää data-aineistoa [1]. Visualisoinnissa kuvataan Winsteadin opinnäytetyössä taulukoitujen alusluokkien ominaisuuksien suhteita niiden painon, nopeuden ja hinnan suhteen kolmiulotteisessa avaruudessa. Lisäksi alusluokkien pituuksia kuvataan datapisteiden väreillä. Näin ollen yhdellä kuvalla visualisoidaan käytännössä neljää ulottuvuutta ja näiden keskinäisiä suhteita.

```
data = pd.read_csv("shipdat.csv")

set1 = data.loc[(data["Length"] > 100)]
set2 = data.loc[(data["Length"] >= 50) & (data["Length"] <= 100)]
set3 = data.loc[(data["Length"] < 50)]

index1 = set1.index
index2 = set2.index
index3 = set3.index

scaler = preprocessing.MinMaxScaler()

data = scaler.fit_transform(data)

set1 = data[index1, :]
set2 = data[index2, :]
set3 = data[index3, :]

data = pd.DataFrame(data)
columns = ["Length", "Tonnage", "Speed", "Cost"]
data.columns = columns

x1 = set1[:,1]
x2 = set2[:,1]
x3 = set3[:,1]

y1 = set1[:,2]
y2 = set2[:,2]
y3 = set3[:,2]

z1 = set1[:,3]
z2 = set2[:,3]
z3 = set3[:,3]

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(x1, y1, z1, label='> 100m', marker = '.', color='blue')
ax.scatter(x2, y2, z2, label='< 100m', marker = '.', color='cyan')
ax.scatter(x3, y3, z3, label='< 50m', marker = '.', color='purple')
ax.set_xlabel("Uppouma")
ax.set_ylabel("Nopeus")
ax.set_zlabel("Hinta")
ax.legend()
plt.show()
```

Seuraavaksi tarkasteltiin koneoppimismenetelmiä. Tähän tuotettiin simuloitua dataa. Tietorakenteen rungon luomiseen käytettiin tekstitiedostoa, jonka pystyi lukemaan suoraan Python pandas dataframe rakenteeksi simuloitavia muita maaleja varten:

```
ALT          CRS    SPD    LAT     LONG   ID

29050.0      102.0  488.8  60.027  28.389  1

14300.0      209.8  367.6  59.955  24.368  1
```

Tämä tekstitiedosto luettiin seuraavalla koodilla dataraamiksi:

```
def load_data():

    fd = pd.read_csv("flightdata.txt", sep="\t")

    fd.describe()

    return fd
```

Ilmamaalit simuloitiin kategorioittain seuraavilla metodeilla, joita kutsuttiin erilaisten mallien ja visualisointien metodeista aina erikseen erilaisten kokoonpanojen mahdollistamiseksi.

```
def neutral():
    alt = random.randint(20000, 35000)
    spd = random.randint(400, 500)
    crs = random.randint(0, 361)
    lat = random.uniform(57.00, 62.00)
    long = random.uniform(15.00, 25.00)
    id = 1

    data = {'ALT': alt,
         'CRS': crs,
         'SPD': spd,
         'LAT': lat,
         'LONG':long,
         'ID': id}

    neu = pd.Series(data)

    return neu
```

Yllä kuvattu ”neutraali” ilmamaali eli käytännössä reittikone, id tunnus numerolla 1.

```
def low_slow_flyer():
    alt = random.randint(300, 6000)
    spd = random.randint(200, 400)
    crs = random.randint(0, 361)
    lat = random.uniform(57.00, 62.00)
    long = random.uniform(15.00, 25.00)
    id = 0

    data = {'ALT': alt,
        'CRS': crs,
        'SPD': spd,
        'LAT': lat,
        'LONG':long,
        'ID': id}

    lsf = pd.Series(data)

    return lsf
```

Yllä matalalla lentävä hidas kone, jollaista yleisesti pidetään potentiaalisena ilmauhkana pinta-alukselle esimerkiksi tiedustelutarkoituksessa. Nimensä mukaisesti satunnaisesti arvottu lentokorkeus on huomattavasti matalampi kuin ylempänä olevan reittikoneen vaihteluväli, eli [300:6000] versus [20000:35000].

```
def fighter_bomber():
    alt = random.randint(6000, 30000)
    spd = random.randint(500, 1000)
    crs = random.randint(0, 361)
    lat = random.uniform(57.00, 62.00)
    long = random.uniform(15.00, 25.00)
    id = 2

    data = {'ALT': alt,
        'CRS': crs,
        'SPD': spd,
        'LAT': lat,
        'LONG':long,
        'ID': id}

    fb = pd.Series(data)

    return fb
```

Yllä olevan hävittäjäpommittajan profiili pitää sisällään lentokorkeuksia välillä [6000:30000] mutta suurin ero aiempiin kategorioihin tulee nopeudessa, joka on huomattavasti suurempi ja alarajaltaan aiempien kategorioiden ylärajalla.

Muodostamalla simuloitu maalidata näillä attribuuteilla käytännössä varmistutaan siitä, että saadaan tuotettua lukuun 2 helposti hahmotettavia visualisointeja ja koneoppimismalleja havainnollistamistarkoituksessa.

Yleensä visualisointi on käytännöllistä kaksiulotteisessa piirreavaruudessa, jossa voidaan helposti arvioida kahden valitun piirteen vaikutusta toisiinsa. Yksinkertaisimmillaan visualisoidaan sitä, ovatko piirteet erotettavissa eli separoitavissa

toisistaan. Mikäli eivät, ei kyseisellä piirreavaruudella ole mahdollista muodostaa ainakaan täydellisesti toimivaa luokittelijaa. Näin on esimerkiksi tässä liitteessä simuloidun ilmamaalidatan paikkatietojen suhteen, sillä kaikkien datapisteiden koordinaattien vaihteluväli on sama ja näin ollen jokaisen kategorian maalit sijoittuvat satunnaisesti samalle maantieteelliselle alueelle, eikä niitä näin ollen ole erotettavissa toisistaan paikkatiedon perusteella. Tästä syystä paikkatietoa ei myöskään käytetä luokittajien koulutuksessa, vaan piirreavaruudesta hyödynnetään pelkästään nopeutta ja korkeutta.

Simuloituun dataan sovitettiin eriasteisia regressioita hyödyntämällä polynomista regressiota eri parametrein seuraavan metodin mukaisesti.

Koska malleissa käytettiin eri määriä simuloituja datapisteitä havainnollistamaan datan määrän vaikutusta mallien kompleksisuuteen ja tarkkuuteen, simuloitiin aineisto jokaisessa metodissa uudelleen vastaamaan tietyn kokoista aineistoa.

```
def polynomial_regression(q, w):
    df = load_data()
    n = q

    for i in range(n*5):
        df = df.append(low_slow_flyer(), ignore_index=True)
    for i in range(n*11):
        df = df.append(fighter_bomber(), ignore_index=True)

    if(q != 8):
        s = 22
    else:
        s = 20

    for i in range(n*s):
        df = df.append(neutral(), ignore_index=True)

    l = len(df)

    scaler = preprocessing.MinMaxScaler()

    d = df.loc[:,["ALT", "SPD"]]
    x = d.values
    data = scaler.fit_transform(x)
    data = pd.DataFrame(data)
    data["ID"] = df["ID"]
    df = data

    labeled_0 = df[df.ID == 0]
    labeled_1 = df[df.ID == 1]
    labeled_2 = df[df.ID == 2]

    df = df.rename(columns={0:'ALT', 1:'SPD'})

    df = df.sort_values(by=['ALT'])
    linear_regressor = LinearRegression()
    X = df.loc[:,'ALT'].values.reshape(-1,1)

    Y = df.loc[:,'SPD'].values.reshape(-1,1)
    linear_regressor.fit(X,Y)
    y_pred = linear_regressor.predict(X)

    from sklearn.preprocessing import PolynomialFeatures

    m=w

    poly = PolynomialFeatures(degree=m)
    X_poly = poly.fit_transform(X)
    linear_regressor.fit(X_poly, Y)
    Y2 = linear_regressor.predict(poly.fit_transform(X))

    plt.scatter(df.loc[:,'ALT'], df.loc[:,'SPD'], color="blue")
    plt.plot(X, Y2, color='orange')
    plt.ylabel('Lentonopeus')
    plt.xlabel('Lentokorkeus')
    title = "{} asteen polynomiregressio, {} datapistettä".format(m, l)
    plt.title(title)
    plt.show()
```

Polynomisen regression lisäksi demonstroitiin multinomiaalinen logistinen regressio.

```
def logistic():

    df = load_data()

    n = 8

    for i in range(n*5):
        df = df.append(low_slow_flyer(), ignore_index=True)

    for i in range(n*11):
        df = df.append(fighter_bomber(), ignore_index=True)

    for i in range(n*20):
        df = df.append(neutral(), ignore_index=True)


    scaler = preprocessing.MinMaxScaler()

    d = df.loc[:,["ALT", "SPD"]]
    x = d.values
    data = scaler.fit_transform(x)
    data = pd.DataFrame(data)
    data["ID"] = df["ID"]
    df = data

    df = df.rename(columns={0:'ALT', 1:'SPD'})

    df = df.sort_values(by=['ALT'])

    logistic_model = LogisticRegression(multi_class='multinomial',
    solver='lbfgs')

    X = df[['ALT','SPD']].values

    Y = df.loc[:,'ID'].values

    logistic_model.fit(X,Y)

    y_pred = logistic_model.predict(X)

    print("training score : %.3f" % (logistic_model.score(X, Y)))
```

```
data = pd.DataFrame(X)
    pred = pd.DataFrame(y_pred)
    data["ID"] = pred
    df = data
    df = df.rename(columns={0:"ALT", 1:"SPD"})

    filtered_0 = df[df.ID == 0]
    filtered_1 = df[df.ID == 1]
    filtered_2 = df[df.ID == 2]

    intercept1 = logistic_model.intercept_[0]
    intercept2 = logistic_model.intercept_[1]
    intercept3 = logistic_model.intercept_[2]

    xmin, xmax = -0.05,1.05
    ymin, ymax = -0.05,1.05

    coef11, coef12 = logistic_model.coef_[0].T
    coef21, coef22 = logistic_model.coef_[1].T
    coef31, coef32 = logistic_model.coef_[2].T

    c1 = -intercept1/coef12
    m1 = -coef11/coef12

    c2 = -intercept2/coef22
    m2 = -coef21/coef22

    c3 = -intercept3/coef32
    m3 = -coef31/coef32

    xd1 = np.array([xmin,xmax])
    yd1 = m1*xd1+c1
    plt.plot(xd1, yd1, 'k', lw=1, ls='--')

    xd2 = np.array([xmin,xmax])
    yd2 = m2*xd2+c2
    plt.plot(xd2, yd2, 'k', lw=1, ls='--')

    xd3 = np.array([xmin,xmax])
    yd3 = m3*xd3+c3
    plt.plot(xd3, yd3, 'k', lw=1, ls='--')

    plt.scatter(filtered_0.loc[:,'ALT'], filtered_0.loc[:,'SPD'], color="r")
    plt.scatter(filtered_1.loc[:,'ALT'], filtered_1.loc[:,'SPD'], color="b")
    plt.scatter(filtered_2.loc[:,'ALT'], filtered_2.loc[:,'SPD'], color="g")
    plt.xlim(xmin, xmax)
    plt.ylim(ymin, ymax)
    plt.ylabel('Lentonopeus')
    plt.xlabel('Lentokorkeus')
    plt.title("Multinomiaalinen logistinen regressio")

    plt.show()
```

Lisäksi demonstroitiin ohjaamatonta oppimista kmeans-metodilla, joka on kuvattu alla.

```
def kmeans():
    df = load_data()

    n = 8

    for i in range(n*5):
        df = df.append(low_slow_flyer(), ignore_index=True)

    for i in range(n*11):
        df = df.append(fighter_bomber(), ignore_index=True)

    for i in range(n*20):
        df = df.append(neutral(), ignore_index=True)

    scaler = preprocessing.MinMaxScaler()

    d = df.loc[:,["ALT", "SPD"]]
    x = d.values
    data = scaler.fit_transform(x)
    data = pd.DataFrame(data)
    df = data

    df = df.rename(columns={0:'ALT', 1:'SPD'})

    df = df.sort_values(by=['ALT'])

    kmeans = KMeans(n_clusters=3)
    label = kmeans.fit_predict(df)

    df['label'] = label

    filtered_0 = df[label == 0]
    filtered_1 = df[label == 1]
    filtered_2 = df[label == 2]

    centroids = kmeans.cluster_centers_

    plt.scatter(filtered_0.loc[:,'ALT'], filtered_0.loc[:,'SPD'], color="r")
    plt.scatter(filtered_1.loc[:,'ALT'], filtered_1.loc[:,'SPD'], color="b")
    plt.scatter(filtered_2.loc[:,'ALT'], filtered_2.loc[:,'SPD'], color="g")
    centroids_x = centroids[:,0]
    centroids_y = centroids[:,1]
    plt.scatter(centroids_x, centroids_y, marker="x", s=150, color='black')
    plt.ylabel('Lentonopeus')
    plt.xlabel('Lentokorkeus')
    plt.title("KMeans-rypästys")

    plt.show()
```

Kmeans-luokiteltua dataa verrattiin alkuperäisiin luokituksiin silmämääräisesti visualisoimalla alkuperäinen data metodilla plot():

```
def plot():

    df = load_data()

    n = 8

    for i in range(n*5):
        df = df.append(low_slow_flyer(), ignore_index=True)

    for i in range(n*11):
        df = df.append(fighter_bomber(), ignore_index=True)

    for i in range(n*20):
        df = df.append(neutral(), ignore_index=True)

    scaler = preprocessing.MinMaxScaler()

    d = df.loc[:,["ALT", "SPD"]]
    x = d.values
    data = scaler.fit_transform(x)
    data = pd.DataFrame(data)
    data["ID"] = df["ID"]
    df = data

    df = df.rename(columns={0:'ALT', 1:'SPD'})

    df = df.sort_values(by=['ALT'])

    group0 = df[df.ID == 0]
    group1 = df[df.ID == 1]
    group2 = df[df.ID == 2]

    plt.scatter(group0.loc[:,'ALT'], group0.loc[:,'SPD'], color="r")
    plt.scatter(group1.loc[:,'ALT'], group1.loc[:,'SPD'], color="b")
    plt.scatter(group2.loc[:,'ALT'], group2.loc[:,'SPD'], color="g")

    plt.ylabel('Lentonopeus')
    plt.xlabel('Lentokorkeus')
    plt.title("Ilmamaalien alkuperäiset luokitukset")

    plt.show()
```

Lisäksi samalla datalla koulutettiin päätöspuu, joka on helposti visualisoitavissa ymmärrettäväksi rakenteeksi. Päätöspuu-luokittelijan kouluttamisen koodi on alla.

```
def decision_tree():

    from sklearn import tree
    df = load_data()
    n = 8
    for i in range(n*5):
        df = df.append(low_slow_flyer(), ignore_index=True)
    for i in range(n*11):
        df = df.append(fighter_bomber(), ignore_index=True)
    for i in range(n*20):
        df = df.append(neutral(), ignore_index=True)

    features = ['ALT','CRS','SPD','LAT','LONG']
    classnames = ['LSF', 'Neutral', 'Fighter']
    X = df[features]
    y = df['ID']

    classifier = tree.DecisionTreeClassifier()
    classifier = classifier.fit(X, y)

    fig = plt.figure(figsize=(25,20))
    _ =tree.plot_tree(classifier,
                      feature_names=features,
                      class_names=classnames,
                      filled=True)

    fig.savefig("decistion_tree.png")
```

Ohjelmisto ajettiin Anaconda-komentorivillä Notepad++ tiedostosta ilman erillistä IDE-ohjelmistoa. Kutsuttu päämetodi on muotoa:

```
def main():
    plot()
    polynomial_regression(8, 6)
    logistic()
    kmeans()
    decision_tree()

if __name__ == "__main__":
    main()
```

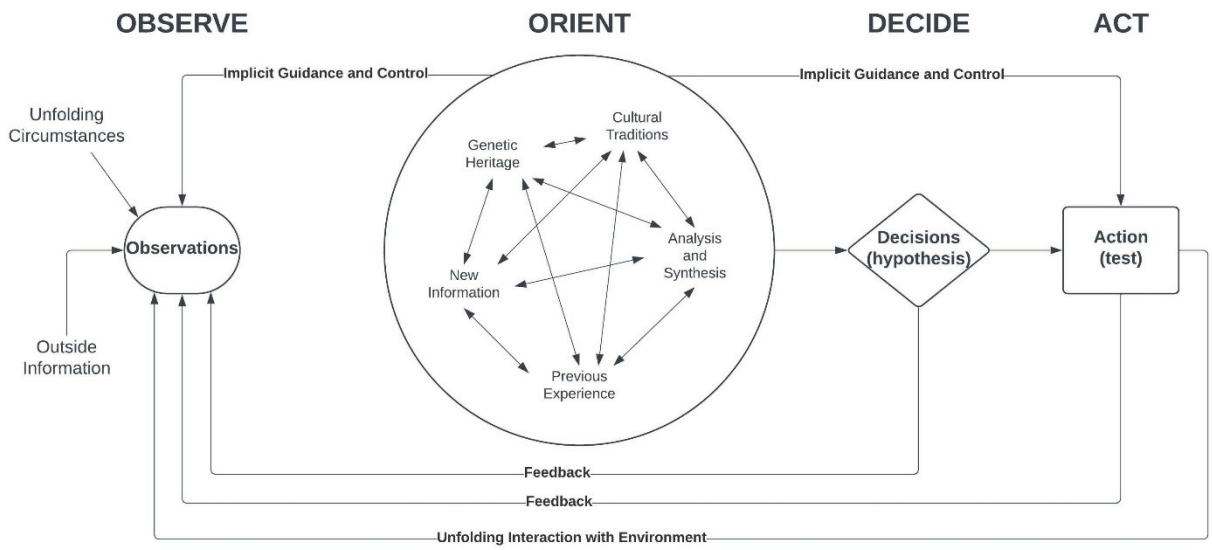
Käytetyt ohjelmistoversiot kaikissa tämän opinnäytetyön liitteissä ovat:

Python 3.9.7	
anaconda-client	1.9.0
anaconda-navigator	2.1.1
conda	4.10.3
ipython	7.29.0
jupyter	1.0.0
keras	2.7.0
Keras-Preprocessing	1.1.2
Markdown	3.3.6
matplotlib	3.4.3
matplotlib-inline	0.1.2
notebook	6.4.5

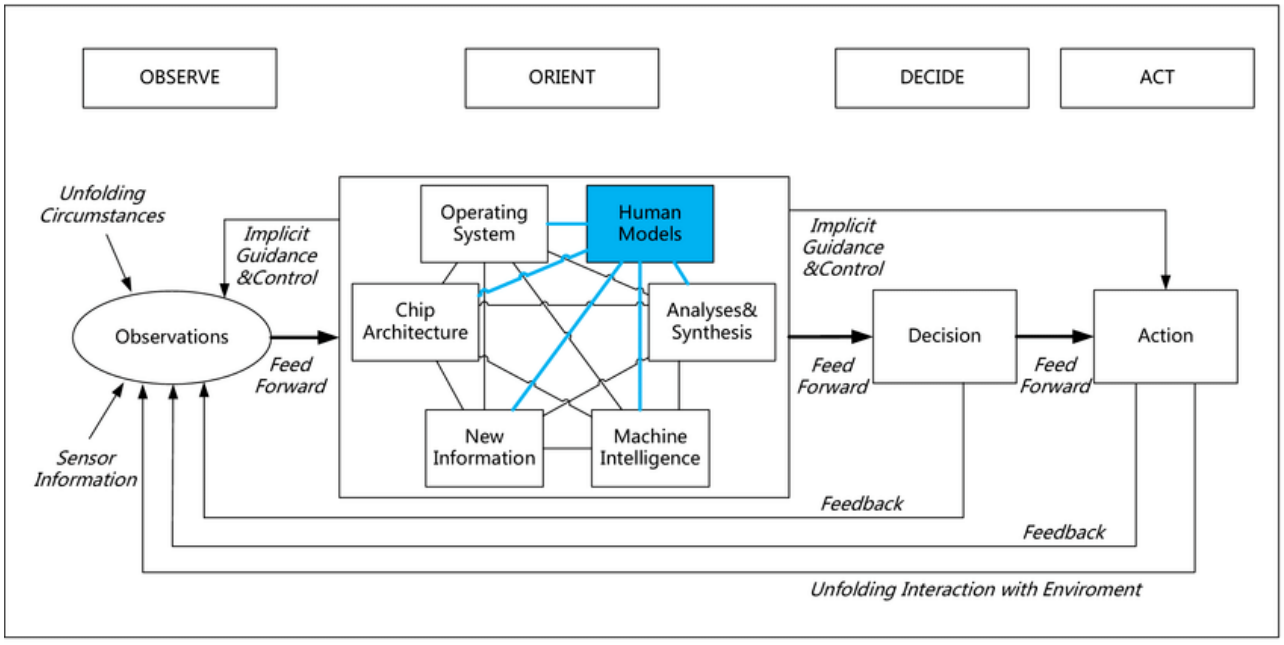
numpy	1.20.3
opencv-python	4.5.5.62
packaging	21.0
pandas	1.3.4
Pillow	8.4.0
pip	21.2.4
plot-keras-history	1.1.30
scikit-image	0.18.3
scikit-learn	1.0.2
scipy	1.7.1
sklearn	0.0
statsmodels	0.12.2
tensorboard	2.8.0
tensorflow	2.7.0
toolz	0.11.1
Unidecode	1.2.0

LÄHTEET

[1] Winstead P.J., ”*Implementation of Unmanned Surface Vehicles in the Distributed Maritime Operations Concept”*, joulukuu 2018, Naval Postgraduate School, Monterey CA, USA



KUVA 1: OODA-silmukka [1]



KUVA 2: Ihmisen ja koneen yhteistyön OODA-silmukka [2]

LÄHTEET

[1] Boyd J.R., Hammond G.T., *A Discourse on Winning and Losing*, Air University Press, Curtis E. LeMay Center for Doctrine Development and Education, Maxwell AFB, Alabama, USA, maaliskuu 2018, ISBN 978-585-566-279-1

[2] Blaha L.M., *Interactive OODA Processes for Operational Joint Human-Machine-Intelligence*, Pacific North-west National Laboratory, Richland, Washington, USA, 18.7.2018, STO-MP-IST-160, DOI 10.14339/STO-MP-IST-160-PP-3P-PDF

Vedenalainen tilannekuva ja tekoäly

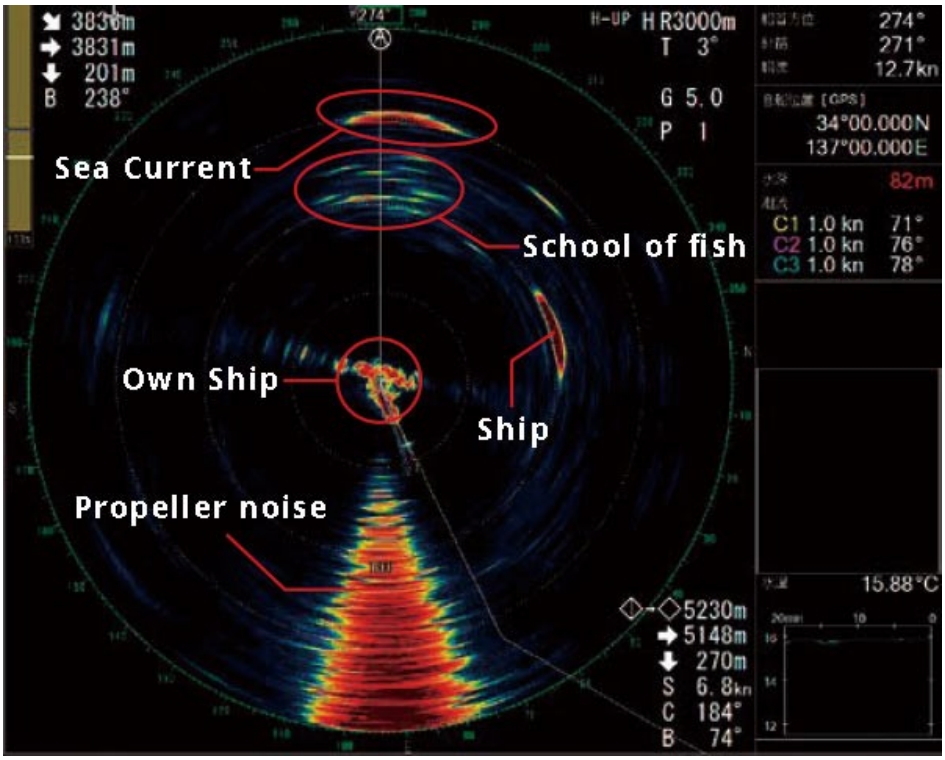
Vedenalainen valvonta

Vedenalaisessa sodankäynnissä, kuten muissakin dimensioissa, oleellista on tilannekuvan muodostus ja siinä erityisesti kyky havaita vastustaja riittävän ajoissa. Vedenalainen tilannekuva muodostetaan vedenalaisella valvonnalla, jota toteutetaan sekä passiivisin että aktiivisin sensorein. Pääasiallinen vedenalaisen valvonnan sensori on sonar, joka nykyisen fysiikan tuntemuksen valossa pitää asemansa toistaiseksi. Tärkeiden kohteiden havaitseminen sonarin tuottamasta datasta on haastavaa niin haittamelun kuin inhimillisten tekijöiden vuoksi. [1, s. 179] Tässä liitteessä tarkastellaan nimenomaisesti aktiivisen mittaimen toimintaperiaatteita ja tuottamaa dataa. Passiivisten järjestelmien tuottaman datan prosessoinnista tekoälyavusteisesti on tehty tutkimusta esimerkiksi Meritaistelukeskuksessa.

Sukellusveneentorjunta-aselajin toimijat aluksella koostuvat sukellusveneentorjuntaupseerista (ASWO), jonka alaisina toimivat kaksi kaikumittainoperaattoria (SONOP). Sukellusveneentorjuntatehtävässä kaikumittainoperaattorit pyrkivät löytämään sukellusveneen käyttämällä aluksen vedenalaisen valvonnan sensoreita. Kaikumittaimen toimintaperiaate koostuu lähetys- ja kuuntelujaksoista, joilla sensori lähettää äänipulssin veden alle ja muodostaa kuuntelujaksolla vedenalaista tilannekuvaa yhdistämällä paluukaiuista suunta- ja etäisyystietoja visualisoiden mittaussektorin havainnot aluksen suhteen. Kaikumittaimen englanninkielinen nimi SONAR tulee sanoista ”sound navigation and ranging”. [2, s. 6-7] Tämän opinnäytetyön tarkastelu perustuu Kongsbergin tuottamaan ST2400 VDS (variable depth sonar) järjestelmään, joka on nimenomaisesti sukellusveneentorjuntaan tarkoitettu syvyytettävä kaikumittain [3]. Kyseisen järjestelmän tekniset ominaisuudet ovat valmistajan julkisen esitteen mukaisesti taulukon 1 mukaiset.

TAULUKKO 1: Kongsberg Maritime AS ST2400 sukellusveneentorjuntakaikumittaimen tekniset tiedot [3]

Aktiivinen mittaustaajuus	22-29kHz
Lähtötaso (ympärisäteilevä)	213dB
Lähetysmoodit	FM (taajuusmodulaatio), CW (kiinteä taajuus)
Vastaanottomoodit	FM, CW, passiivinen
Operointimoodit	Ympärisäteilevä, sektorilähetys, yksittäinen keila
Yksittäisen keilan leveys	Horisontaalisti 8º±2º, vertikaalisti 12º±2º
Horisontaalien vastaanottokeilojen määrä	128
Tyypillinen havaitsemisetäisyys (kohteen vahvuus 0dB, veden suolaisuus 20‰)	5500m lähetystaajuudella 24kHz 6000m lähetystaajuudella 22kHz
Suurin hinausnopeus	16 solmua



KUVA 1: Furunon esimerkkikuva sonar-näkymästä [4]

Kuvassa 1 on esimerkkinäkymä operaattorin käyttöliittymästä sonar-järjestelmässä, johon on piirtynyt edellisellä mittainpulssilla kaksi voimakkaasti kuuluvaa kohdetta, ”ship” ja ”sea current”, sekä vaimeampia paluukaikuja eli kalaparvia. Vedenalaisen tilannekuvan luomisessa kaikumittainoperaattori tarkastaa paluukaikujen profiilit visuaalisesti kuvan 1 kaltaisesta näkymästä ja siirtää kuunteluvektori mielenkiintoisten kaikujen suuntaan luokittelun mahdollistamiseksi. On huomionarvoista, että operaattori pystyy lähtökohtaisesti kuuntelemaan vain yhtä suuntaa eli keilaa kerrallaan. Vaikka järjestelmä taltioi kaikkien valitun sektorin keilojen paluukaiut, ei kohteiden samanaikainen tarkastelu ja tunnistaminen useasta suunnasta äänen perusteella ole mahdollista yhdellä operaattorilla.

Mikäli mittainpulssi lähetetään kiinteällä pistetaajuudella, pystytään liikkuvan kohteen paluukaiun Doppler-siirtymästä laskemaan kohteen olevan liikkeessä aluksen liikkeen suhteen. Aluksen liikenopeus ja suunta ovat tunnettuja, jolloin aluksen liikkeen aiheuttama Doppler-siirtymä voidaan laskea erotuksena paluukaiun taajuussiirtymästä ja mikäli jäljelle jäänyt siirtymä poikkeaa riittävästi nollasta, on havaittu kohde liikkeellä aluksen suhteen. Liikkeellä oleva vedenalainen kohde on helppo tunnistaa ei-luonnolliseksi ja siten oletettavasti sukellusveneeksi.

$$f = f_0 \frac{v}{v \pm v_1}, \qquad kun\ aaltolähde\ liikkuu$$

$$f = f_0 \frac{v \pm v_2}{v}, \qquad kun\ havaitsija\ liikkuu$$

KAAVA 1: Dopplerin ilmiön vaikutus havaittuun taajuuteen [5 s. 121]

Kun kaavaan 5 sijoitetaan ST2400 keskitaajuus 25kHz ultraäänialueelta ja määritetään mittaavan aluksen etsintänopeudeksi 5 solmua eli 2.572m/s ja äänennopeudeksi vedessä noin 1400m/s veden lämpötilan ollessa 0°C, saadaan Doppler-siirtymäksi noin ±46 hertsiä riippuen suhteellisesta nopeudesta eli siitä, lähestytäänkö kohdetta vai loitotaanko

siitä. Täten kaikki kohteet, joilla havaitaan tästä poikkeava Doppler-siirtymä, voidaan olettaa olevan liikkeellä. Mikäli kohde kulkee vastakkaiseen suuntaan samalla 5 solmun nopeudella, Doppler-siirtymä kaksinkertaistuu ja vastaavasti nollaantuu, mikäli kohde ottaa samat liiketekijät kuin aaltolähde. Mahdollinen sukellusvene voi näin ollen minimoida havaituksi tulemisen esimerkiksi täsmäämällä mittaavan aluksen liiketekijät omalla liikevektorillaan minimoiden Doppler-siirtymän tai hidastamalla liikkeensä sekoittuakseen pohjanmuotojen tuottamiin häiriökaikuihin. Pohjassa makaava sukellusvene muistuttaa kiveä tai hylkyä ja näiden eron on aiemmin ollut havaittavissa ainoastaan ihmisoperaattorille [6, s. 2-3]. Mikäli liikettä ei voida indikoida, ainut tapa havaita sukellusvene tavanomaisilla sukellusveneentorjuntaan tarkoitetuilla etsintäsonareilla on analysoida paluukaiun voimakkuutta eli amplitudia sekä kaiun ääniprofiilia. Sukellusvene koostuu nykyisin yhdestä tai useammasta teräksisestä painerungosta. Painerungon sisällä on ilmaa ja yleensä sen ulkopuolella, muotorungon ja painerungon välissä, on nestetankkeja, joissa on polttoainetta tai painolastivettä [2, s. 290]. Äänen nopeus teräksessä on 5100m/s. Vastaavasti äänen nopeus graniitissa on 4000m/s. Äänen nopeuden vedessä ollen suolaisuudesta ja lämpötilasta riippuen noin 1400m/s-1500m/s muodostuu sekä graniitin että teräksen kanssa jyrkkä rajapinta, joka heijastaa ääntä takaisin. Laskettaessa takaisin heijastuvan kaiun voimakkuutta merkittäviä tekijöitä ovat heijastavan pinnan muoto ja ääniaallon tulokulma. Koska näissä on merkittävää hajontaa muun muassa pohjanmuotojen pinnoissa sekä aaltoliikkeen tulokulmassa niin sukellusveneen kuin pohjanmuotojen rajapintaan, ei ole tämän tutkimuksen puitteissa mielekästä iteroida kaikkia vaihtoehtoja läpi. Sen sijaan yleistä eroa kahden havainnon, teräksisen sylinterin ja graniittilohkareen, välillä voidaan havainnollistaa akustisilla impedansseilla sekä heijastumiskertoimella.

$$\text{Akustinen impedanssi } Z = \rho v$$

KAAVA 2: Akustinen impedanssi on väliaineen tiheyden ja sen äänennopeuden tulo [7 s. 123]

$$\mathcal{R}(\theta_I, \omega) = \frac{\zeta(\omega) \cdot \cos(\theta_I) - 1}{\zeta(\omega) \cdot \cos(\theta_I) + 1} \qquad (1)$$

$$\rightarrow \zeta(\omega) = \frac{Z_s}{Z} = \frac{Z_s}{\rho_2 v_2}, Z_s = \frac{\rho_1 v_1}{\cos(\theta_I)}$$

$$\rightarrow \zeta(\omega) = \frac{\frac{\rho_1 v_1}{\cos(\theta_I)}}{\rho_2 v_2} = \frac{\rho_1 v_1}{\rho_2 v_2 \cos(\theta_I)}, \quad sij. kaavaan\ 1$$

$$\mathcal{R}(\theta_I, \omega) = \frac{\frac{\rho_1 v_1}{\rho_2 v_2 \cos(\theta_I)} \cdot \cos(\theta_I) - 1}{\frac{\rho_1 v_1}{\rho_2 v_2 \cos(\theta_I)} \cdot \cos(\theta_I) + 1} = \frac{\left(\frac{\rho_1 v_1}{\rho_2 v_2}\right) - 1}{\left(\frac{\rho_1 v_1}{\rho_2 v_2}\right) + 1} = \frac{\rho_1 v_1 - \rho_2 v_2}{\rho_1 v_1 + \rho_2 v_2} = \left|\frac{Z_1 - Z_2}{Z_1 - Z_2}\right|$$

KAAVA 3: Heijastumiskertoimen kaava. $\zeta(\omega)$ on spesifin akustisen impedanssin ja akustisen ominaisimpedanssin suhde ja Θ on aaltoliikkeen taitekulma. [7 s. 122-125]

Kaavassa 3 on johdettu akustisen impedanssin avulla heijastumiskertoimen kaava taitekulmapohjaisen heijastumiskertoimen kaavasta, jossa heijastus oletetaan kohtisuoraksi, jolloin voidaan sijoittaa $\cos(\theta_I) = \cos(0^0) = 1$. [7 s. 122-125]

Teräksen akustinen impedanssi on $7870\frac{kg}{m^3} \cdot 5100\frac{m}{s} = 40\ 137\ 000 Pa \cdot s/m^3$

Graniitin akustinen impedanssi on $2700\frac{kg}{m^3} \cdot 4000\frac{m}{s} = 10\ 800\ 000 Pa \cdot s/m^3$

Veden akustinen impedanssi on $1000\frac{kg}{m^3} \cdot 1403\frac{m}{s} = 1\ 403\ 000 Pa \cdot s/m^3$, kun vesi oletetaan suolattomaksi ja 0°C lämpöiseksi. [7 s. 76-78, 91]

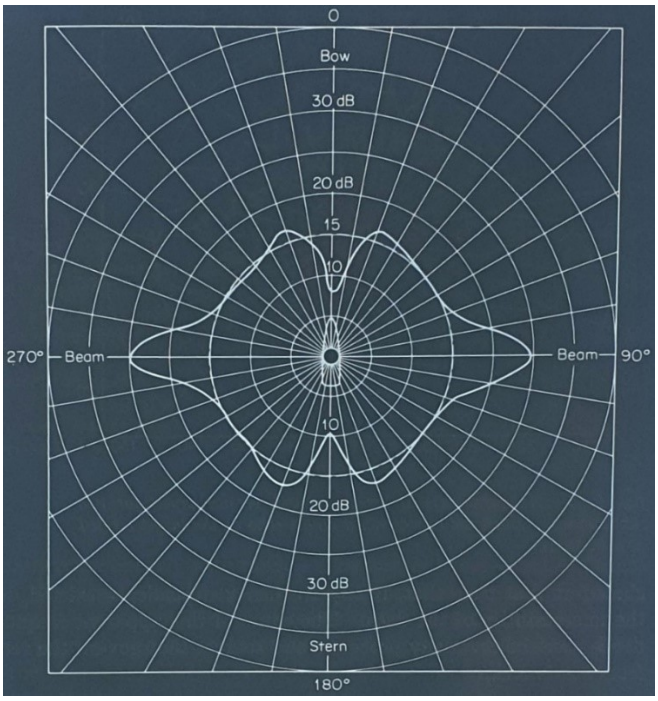
Näin ollen rajapinta teräksen ja graniitin osalta muodostuu erilainen akustinen heijastuma, teräksen akustisen impedanssin ollessa noin nelinkertainen graniittiin nähden. Tämä ero impedansseissa, joka johtuu välittäjäaineen äänennopeuseroista, muodostaa heijastumiskertoimiksi kaavan 3 perusteella ylläolevilla veden oletusarvoilla lasketusti seuraavat:

$$\mathcal{R}_{teräs} = \left|\frac{1\ 403\ 000 - 40\ 137\ 000}{1\ 403\ 000 + 40\ 137\ 000}\right| \approx 0.9325$$

$$\mathcal{R}_{graniitti} = \left|\frac{1\ 403\ 000 - 40\ 137\ 000}{1\ 403\ 000 + 40\ 137\ 000}\right| \approx 0.77$$

Ylläolevien heijastumiskertoimien ero kuvaa pinnasta heijastuvan aallon osuutta ja mahdollistaa sukellusveneen teräsrungon erottamisen esimerkiksi vastaavan kokoisesta ja muotoisesta graniittilohkareesta, jotka olisivat samassa kulmassa. Näiden kertoimien valossa muuten identtisten, samalla etäisyydellä olevien kohteiden paluukaiun pistemäisen voimakkuuden pitäisi olla noin 21% voimakkaampi, kun kyseessä on teräksinen kohde verrattuna graniittiseen kohteeseen. Lisäksi, koska teräksisen kohteen heijastuskerroin on suurempi, tulisi kaiun olla ”terävämpi”. Ero on kuitenkin ihmiskorvalle pieni ja lisäksi akustisen näytteen tunnistettavuus voi vaihdella järjestelmittäin. Kokenut kaikumittainoperaattori pystyy, muiden havaintojen tukemana, päättelemään akustisen havainnon luokittelun varmuuden tietyllä sukellusveneen etsinnän havaintojen asteikolla. Muut luokittelua tukevat havainnot ovat esimerkiksi pohjanmuotojen poissulkeminen karttatietojen perusteella, pinta-alusten poissulkeminen sekä se, mahdollistaako havaintopaikan syvyys sukellusveneen operoinnin.

Kuten todettua, ylläolevassa yksinkertaistuksessa jää huomiotta muun muassa havaitun kohteen pinnanmuodot ja kaikumittainaallon kohtaamiskulma kohteen pinnan suhteen. Yhdysvaltojen merellisten pinta-asejärjestelmien keskuksen vanhempana tutkijana toiminut Robert J. Urick esittää kirjassa ”*Priciples of underwater sound*” useita havainnollistavia kuvaajia sukellusveneestä muodostuvan paluukaiun ja mittauskulman suhteesta. Kyseiset data-aineistot on koostettu mittaamalla sukellusvenettä todellisissa olosuhteissa merellä. [2 s. 281-284] Kerätty, joskin vanhoihin sukellusvenerunkoihin perustuva data on arvokasta, sillä nykyisellään vastaavanlaisen, säädellyistä olosuhteista kerätyn ja kattavan kokeellisen datan saaminen ei ole helppoa sukellusvenekyvyn omaavien valtioiden varjellessa kalustonsa muodostamia herätteitä erittäin tarkasti.



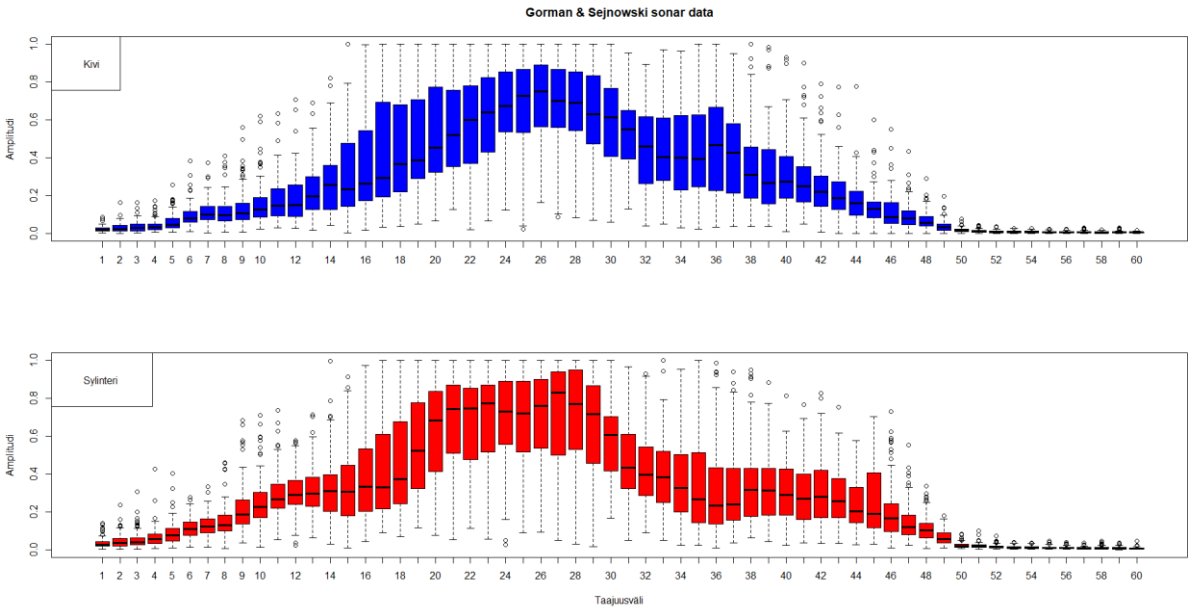
KUVA 2: Urickin ”*Principles of underwater sound for engineers*” kuva 9.13 vastaväreissä. [2 s. 283]

Kuvassa 2 on geneerinen, ”perhosmalliksi” kutsuttu kuvaaja sukellusveneen rungon orientaation vaikutuksesta paluukaiun voimakkuuteen. Kuten kuvasta havaitaan, heikoin paluukaiku tulee keulasta ja perästä, sillä tällöin mittainta kohden on myös pienin pinta-ala. Urick selittää keulan ja perän sivuilla noin 20° kulmassa olevat ja 2-3dB vahvemmat kaiut mahdollisina resonansseina sukellusveneen kyljissä olevista tankeista, jotka saattavat toimia signaalin vahvistajina. Paluukaiun amplitudin vaihtelut sukellusveneen orientaation mukaan aiheuttavat merkittävän haasteen kyseisen paluukaiun erottelemiseksi muista kaiuista. Ratkaisu ongelmaan on nykyisessä operoinnissa yksinkertainen, sillä mikäli kohde liikkuu suhteessa havaitsijaan, siitä havaitaan edellä kuvatusti Doppler-ilmiö. Mikäli se ei liiku, mittausta voidaan suorittaa useammasta kulmasta kiertäen kohdetta ja saaden näin laajempi ja parempi käsitys paluukaiun laadusta. Ongelmaksi muodostuu käyttökelpoisen tekoälymallin luominen tässä kuvattuun ongelmaan, jossa halutaan havaita teoreettisesti korkeamman heijastuskertoimen kohde siitä muodostuvan, suuremman amplitudin ja oletettavasti lyhyemmän keston paluukaiun perusteella tilanteessa, jossa kohteen orientaatio aiheuttaa yli 15 desibelin vaihteluvälin mittaukselle.

Datan ymmärrys

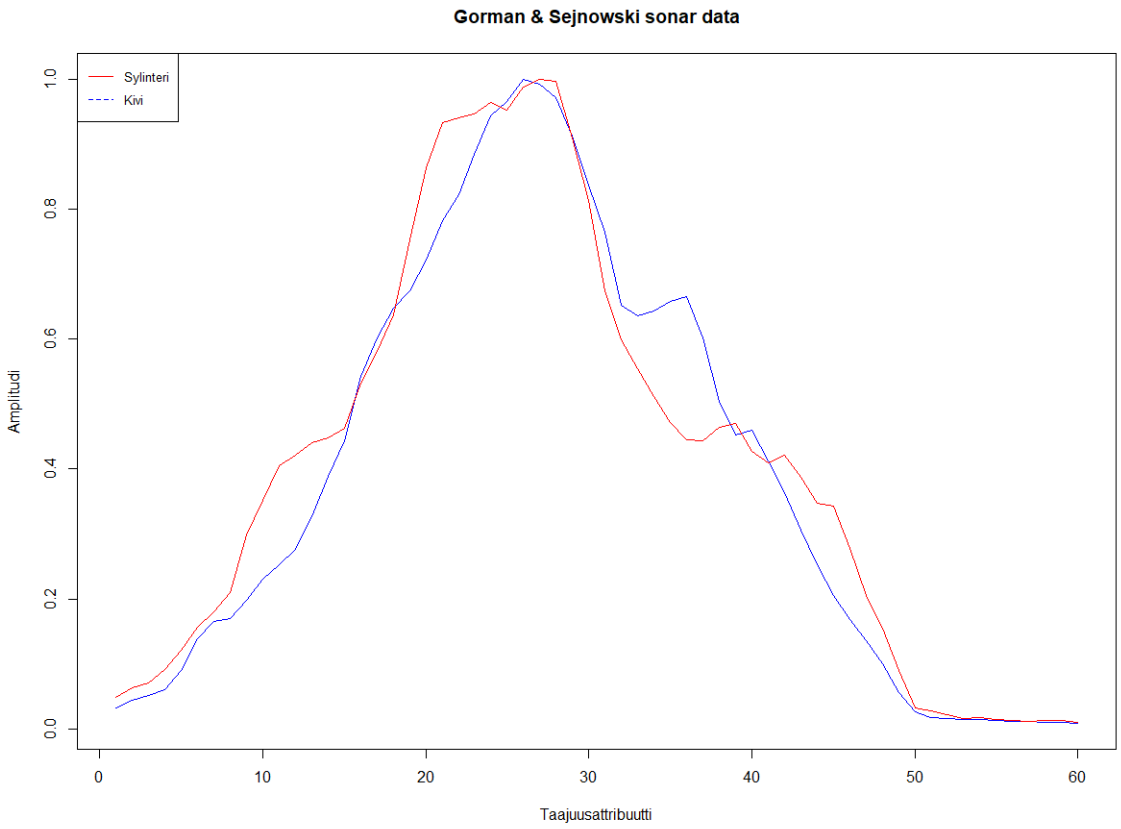
Yllä kuvatusti sonar muodostaa datana aikasarjan, jossa taltioidaan ajan suhteen kyseisestä suunnasta vastaanotettu paluukaiun amplitudi ja taajuus. Aikasarja visualisoidaan karttapohjalle tunnetun tai oletetun äänennopeuden avulla, piirtämällä amplitudihavainnot karttapohjalle, jolloin operaattori kykenee hyödyntämään myös topografiatietoa havaintojen luokittelussa. Topografialla kyetään, etenkin hyvin kartoitetun merenpohjan suhteen, avustamaan luokittelua merkittävästi, sillä pohjan topografian suhteen voidaan arvioida sukellusveneen mahdollisia sijainteja sekä mahdollisesti sukellusveneeltä vaikuttavien kohteiden, kuten kivien, läsnäoloa alueella.

Sonardatan luokitteluongelmaa tekoälyllä on tarkasteltu jo vuonna 1987 R. Paul Gormanin ja Terrence J. Sejnowskin toimesta. Kyseisessä analyysissa vertailtiin neuroverkon piilotettujen kerrosten vaikutusta kykyyn luokitella kaikumittainkaikuja kahteen eri luokkaan, jotta olivat metallisylinteri ja kivi. Artikkelissa todetaan luodun mallin saavuttaneen 90.4% tarkkuuden validointijoukolla, joka on ollut huomattavasti parempi kuin nearest neighbor – luokittelijalla saavutettu 82.7%. Koeolosuhteet, joista kaikumittaimen ääninäytteet oli kerätty, olivat hyvin säädellyt. Kohteet olivat noin 5 jalan mittaisia objekteja ja 10 metrin etäisyydellä mittaimesta. Käytetty mittausääni oli laajakaistainen, lineaarinen taajuusmodulaatio, niin sanottu ”FM-viserrys”. 1200 näytteestä valittiin 208 näiden paluukaiun spektrin voimakkuuden perusteella, vaihteluvälin ollessa 4-15 dB. Kohteita oli mitattu eri kulmista, joten näytteisiin valittiin keskiarvoisesti 5 näytettä jokaisesta kulmasta. Metallisylinterin osalta kulmien vaihteluväli oli 90º ja kiven osalta 180º. Tutkijoiden julkaisemassa näyteaineistossa ei tarkenneta, mitkä näytteet ovat mistäkin kulmasta otettuja. Näyteaineistossa tai artikkelissa ei myöskään tarkenneta näytteiden taajuusaluetta. Näytteet on koostettu laskennallisesti helpommiksi taajuusvälien yhteenlasketuiksi kokonaisamplitudeiksi muodostamalla paluukaiuista lyhyen välin Fourier-muunnoksen (short-term Fourier transform) spektrogrammi ja summaten spektrogrammin amplitudivaihtelut yhteensä 60 taajuusvälille, jotka muodostavat aineiston näytteiden attribuutit. Näin ollen aineistoa ei käsitelty alkuperäisenä raakadatan aikasarjana, vaan taajuusmoduloidun paluupulssin eri taajuusvälien voimakkuuksien eroista koostuvana spektrinä, joka on aikajärjestyksessä. Alla olevassa kuvassa 3 on visualisoitu aineisto luokittain laatikko-janakuvioina, jolloin muodostuu käsitys mittausten hajonnasta ja poikkeavien havaintojen määrästä. Laatikko-janakuviossa numeroidun taajuusvälin (1-60) laatikon koko kuvaa ylä- ja alakvartiilin väliä [Q1:Q2], laatikon sisällä oleva viiva havaintojen mediaania ja viikset ylhäällä joukon laskennallista maksimiarvoa ja vastaavasti alhaalla minimiarvoa. Viiksien ulkopuolelle jäävät pisteet ovat poikkeavia havaintoja eli yli puolitoistakertaisen kvartiilivälin verran ylä- tai alakvartiilin rajan ylä- tai alapuolella. Tutkimuksen raporttia tarkastellessa on valitettavaa, ettei näytteiden taajuutta tai taajuusväliä ole kerrottu. Lisäksi raportissa on huomio, että neuroverkko saavutti kulmariippuvaisella datalla ja 24 piilotetulla kerroksella 100% tarkkuuden koulutusdatalla, mikä voisi viitata ylioppimiseen. Validoinnissa tarkkuus oli tästä huolimatta hyvä, 89.2%, ja parempi kuin kohteen kulmasta riippumattomalla saavutettu 99,8%/84.5% koulutus- ja validointitarkkuus. [8 s. 78-80]



KUVA 3: Gorman ym. käyttämä kaikumittaindata esitettynä laatikko-janakuvioina luokittain. Visualisoitu liitteen 3 ohjelmalla

Koska Gormanin ja Sejnowskin tutkimuksessa mittausetäisyys sekä oletettavasti pulssinpituus ja modulaatio pysyivät muuttumattomina sylinterin ja kiven suhteen, pitäisi yllä kartoitetun fysikaalisten lainalaisuuksien tarkastelun perusteella muodostua kaavamaisuus paluukaiun erosta kahden kohteen välillä. Kun tutkimuksessa käytetty data laskettiin kategorioittain taajuusattribuuttien keskiarvoiksi ja piirrettiin rinnakkain, saatiin kuvan 4 kuvaaja, jossa havaitaan sylinterin tuottavan keskimääräisesti kiveä voimakkaampi paluukaiku taajuusattribuutin 10 ja 20 ympäristöissä sekä hieman verrokkia voimakkaampi paluukaiku attribuutin 45 ympäristössä. Vastaavasti kivestä tulee huomattavasti voimakkaampi paluukaiku 35-40 attribuuttivälillä. Laskettaessa molempien kohteiden paluukaikujen attribuuttien summat, havaitaan, että teräksisestä kohteesta tulee yhteensä noin 24% voimakkaammat paluukaiut, joka korreloi varsin hyvin yllälaskettuun ominaisimpedansseihin perustuvaan 21% heijastuskertoimien eroon. Gorman ym. neuroverkko koulutettiin juurikin kuvan 4 muotoisella datalla, joskin yksittäisistä mittauksista. Keskiarvoisista kuvaajista on helppo havaita edellä mainitut kaavamaiset poikkeavuudet, jotka mahdollistavat korkean erottelukyvyn luokittelijalle. Oletettava käyttötarkoitus tutkimustuloksille onkin ollut miinanetsinnän tukeminen, sillä taltioidussa datassa sylinterin annotaatio on ”miina” ja erittäin lyhyt mittausetäisyys sekä taajuusmoduloitu pulssi viittaavat paikallaan olevan, pienen kohteen etsintään. Sukellusveneen etsinnässä etäisyydet ovat suurempia, kuten kohdekin, ja aktiivisessa kaikumittauksessa käytetään moduloidun taajuuden lisäksi pistetaajuutta Doppler-ilmiön havaitsemiseksi. Lyhyt mittausetäisyys mahdollistaa lisäksi luokittelun amplitudispektristä ilman aikaleimaa, toisin kuin useita kilometrejä matkaava mittauspulssi, joka sisältää huomattavan määrän voimakkuusvaihteluita havainnoista riippuen. Huomionarvoista ja ansiokasta Gorman ym. tutkimuksessa on myös neuroverkon suorituskyvyn vertailu ihmisoperaattorin kanssa. Tästä on visualisoitu koulutusdata, joka vastaa kuvaukseltaan nykyisiä neuroverkkojen koulutuksessa taltioitavia gradientin päivitysväleittäin kirjattuja oppimistuloksia. Kuvassa on visualisoitu ihmisoperaattorin oppimiskäyrä näytteiden erottelemisessa yhteensä 19 opetuskerrassa, joista jokainen on käsittänyt 104 näytettä, päätyen parhaimmillaan lähes 90% luokittelutarkkuuteen. Tutkimuksessa arvioidaan myös neuroverkon muodostamia painokertoimia verrattuna ihmisoperaattorin painottamiin ominaisuuksiin havainnoista, jotka ovat varsin yhtenevät. [8 s. 88]



KUVA 4: Gorman ym. tutkimuksessa käyttämän datan attribuuteittain keskiarvotettu visualisointi. Kuva tuotettu liitteen 4 ohjelmalla.

Koska Gorman ym. näytedata ei ole muokkaamaton aikasarja eikä tarkka taajuusalue ole tiedossa, ei kyseisellä aineistolla voida täysin vahvistaa hypoteesia niin sanotusti terävämmän eli voimakkaan ja lyhyen kuuloisesta paluukaiusta teräksisen kohteen ja kiven välillä huolimatta merkittävästä heijastuskertoimien erosta. Itse julkaisussa on kuitenkin esimerkkikuvat puhtaasti aikasarjaisesta amplitudidatasta, jossa havainnollistetaan kiven ja sylinterin paluukaikujen eroja. Visualisoitu sylinterin paluukaiku näyttää viivästetyltä ja epäyhtenäisemmältä verrattuna kiven paluukaikuun. Kuitenkin aineistosta tehdyt visualisoinnit kuvissa 3 ja 4 osoittavat metallisylinterin tuottavan keskiarvoisesti suuremman amplitudin paluukaiun. Lisäksi, huomioiden taajuusmodulaatio eli se, että näytteiden attribuutit ovat modulaatiojärjestyksessä ja siten aikajärjestyksessä, voidaan kuvista 3 ja 4 tulkita sylinteristä tulevan paluukaiun voimakkuuden kasvavan nopeammin ja vastaavasti laskevan nopeammin kuin kivestä tulleen paluukaiun. Nämä havainnot tukevat hypoteesia sukellusveneestä tulevan paluukaiun tunnistamisesta, mutta sisältävät useita epävarmuustekijöitä, epäselvyyksiä ja oletuksia.

Koneoppimisen kannalta, kun pyritään luomaan luokittelijamalli, joka täydentää kaikumittainoperaattorin työpanosta, voidaan ymmärtää, että yksittäisen keilan mittaus on datapiste, joka muodostaa aikasarjan kaikumittaimen kuuntelujaksollaan vastaanottamista amplitudivaihteluista. Jokin sarja näitä amplitudivaihteluja on varsinainen positiivinen datanäyte eli havainto sukellusveneestä. Koneoppimisongelma on ensinnäkin koostaa riittävä määrä dataa mallin luomiseksi sekä valita datan pohjalta käsitetystä hypoteesiavaruudesta tehtävään sopivin algoritmi ja virhefunktio.

Kuten Urichin kirjan perusteella todettiin, kaikumittaimen paluukaiuissa on merkittävää hajontaa riippuen kohteen orientaatiosta mittaimen suhteen. Intuitiivinen ratkaisu koneoppimisongelman kannalta olisi Gorman ym. tyyppisesti

koostaa taltioidusta datasta näytteet esimerkiksi kohteen keulasta perään 180° sektorista ja tuottaa mahdollisimman monta näytettä per suunta. Ylitsepääsemättömäksi ongelmaksi muodostuu kohteen orientaatiotieto. Suomen laivastolla ei ole omaa sukellusvenettä, eivätkä sukellusveneen omaavat valtiot luovuta omia suorituskykyjään tämän kaltaisiin, säädeltyihin herätemittauksiin. Lisäksi Urich huomauttaa sukellusvenerunkojen paluukaikujen olevan hyvin erilaisia eri sukellusveneluokkien välillä [2 s. 281]. Tämä on luonnollista, jos Urichin oletus on oikea ja esimerkiksi painerunkojen määrä ja tankkien sijoittelu vaikuttaa vahventavasti tai tukahduttavasti tietyssä kulmassa saapuvaan mittainpulssiin. Näin ollen täysin oikeanlaisen datan kerääminen edellyttäisi paitsi varman sukellusvenehavainnon mittaamista riittävällä määrällä erilaisia mittauskulmia, taajuuksia, modulaatioita ja etäisyyksiä, myös sukellusveneen rungon olevan mieluiten sama kuin potentiaalisen vastustajan sukellusveneen rungon. Koska tällainen data-aineisto on mahdotonta toteuttaa, vaihtoehdoiksi jää toteuttaa mahdollisimman hyvä malli sillä datalla, mitä on käytettävissä, tunnistamaan niitä ominaisuuksia ja kaavamaisuuksia, joita edellä on tunnistettu oleellisiksi. Yksi ratkaisu yleispätevän mallin luomiseen on tuottaa operaattorien kokemuksella vahvistettuja ja kerättyjen taltioiden kaltaisia, synteettisiä näytteitä, joiden avulla voidaan luoda merkittävä määrä koulutusdataa, esimerkiksi mallintaen kuvan 2 perhosmallin mukaiset paluukaiun voimakkuudet erilaisten häiriöäänten joukkoon. Näin tuotettua mallia voidaan testata harjoituksissa taltioidulla datalla sen todellisen, operatiivisen suorituskyvyn toteamiseksi.

Koneoppimisen ja datan ymmärryksen kannalta merkittävän tekijä on käsitys tarkasteltavasta ilmiöstä. Siinä missä havaitseva sensori tuottaa aikasarjaa, ei kyseessä ole aikasarjan regressio kuten esimerkiksi pörssikursseja tarkasteltaessa. Sen sijaan tarkoitus on kyetä eristämään pistemäinen havainto aikasarjasta ja luokitella kyseinen havainto johonkin luokkaan. Teoriassa havainnon amplitudin kulmakertoimen eli suhteen edelliseen aikayksikköön pitäisi indikoida mielenkiintoisen havainnon alku. Näin voitaisiin eristää mittauspulssin aikasarjasta mielenkiintoiset havainnot erillisiksi datapisteiksi, joiden pituus voitaisiin määrittää joko dynaamisesti äänennopeuden mukaan vastaamaan esimerkiksi alle 100m etäisyysporttia, johon oletettava dieselsähköinen sukellusvene mahtuisi. Näiden pilkottujen datapisteiden kanssa oltaisiin samassa tilanteessa kuin Gorman ym. tutkimuksessa, jossa vakiomittainen pulssi on muunnettu vakioituihin attribuutteihin. Tällaisten, irrotettujen ja normalisoitujen tapahtumien luokittelu voi onnistua useilla koneoppimismetodeilla.

Datan valmistelu

Tässä tutkimuksessa lähdettiin liikkeelle ajatuksesta, että yksittäisen keilan mittaama aikasarja muutettaisiin spektrogrammiksi, joka kuvaa y-akselilla taajuuksia, x-akselilla aikaa ja yx- pisteiden värein kyseisellä ajanhetkellä ja taajuudella havaittua amplitudia eli äänenvoimakkuutta. Näin yksi datapiste kuvaisi, edellä mainitun kaltaisesti, yhden kuuntelusuunnan ympäristöä. Hypoteesi tässä lähestymisessä oli, että kuvien tunnistamisessa hyödylliseksi osoittautuneet konvoluutioverkot voisivat onnistua hahmottamaan havainnon jyrkkyydestä eli spektrogrammin tapauksessa kirkkaudesta ja niin sanotun kaiun hännän lyhyydestä kyseessä olevan mahdollinen sukellusvene. Hypoteesin testaamiseksi käytettiin julkisista lähteistä saatavilla ollutta, nauhoitettuja ja synteettisiä kaikumittainääniä sekä muita vedenalaisia ääniä kuten propulsiomelua ja merieläinten ääntelyä. Ääninäytteet pilkottiin yksitäisin havaintoihin, jotka luokiteltiin mielenkiintoisiin kohteisiin ja ei-mielenkiintoisiin. Tämän jälkeen näytteet muutettiin spektrogrammeiksi, jotka koulutettiin erilaisille konvoluutoneuroverkoille hyödyntäen myös siirto-oppimista. Alustavien testien perusteella luokitustarkkuus jäi kuitenkin hyvin vaatimattomaksi tällä lähestymisellä.

Vaikka spektrogrammien käyttäminen konvoluutioverkon kouluttamisessa on mahdollista, on kyseisessä lähestymistavassa useita ongelmia. Ensinnäkin aikasarjan visualisointi aiheuttaa sen, etteivät spektrogrammin eri akselit kuvaa samantyyppistä tietoa. Näin ollen konvoluutioverkon tiivistävät kerrokset voivat muuttaa kuvaksi tulkittua ääntä esimerkiksi eritaajuiseksi. Toinen ongelma on siinä, etteivät konvoluutioverkon kaavantunnistusominaisuudet välttämättä sovellu tulkitsemaan kuvaksi muutettua aikasarjaa oikein, sillä siinä missä normaali kuva sisältää objekteja, spektrogrammi sisältää sarjan tapahtumia. Näiden tapahtumien luokittelu edellyttää tapahtuman edellisen vaiheen tulkitsemista suhteessa nykyiseen vaiheeseen. Tästä syystä RNN-struktuuri ja toimintaperiaate vaikuttavat vaihtoehtoisesti parhaalta valinnalta tämän koneoppimisongelman hypoteesiavaruudesta. Kuitenkin, viitaten edellisen luvun pohdintaan aikasarjan pilkkomisesta kiinnostaviin havaintoihin, on aikasarjasta mahdollista irrottaa kiinnostavat havainnot erillisiksi tapahtumiksi ja soveltaa eri metodeja näihin eristettyihin ajanhetkiin.

Edellä mainitun pohdinnan opponointina on todettava, että konvoluutioneuroverkon käyttö on tuottanut myös hyviä tuloksia sonardatan luiokittelussa. Yksi viimeisimpiä tutkimuksia aihepiirissä on Nato Science & Technology Organization, Centre for Maritime Research & Experimentation julkaisema tutkimus, jossa tiivistelmän mukaan on käytetty konvoluutioneuroverkkoa luokittelemaan liikkuvia maaleja aktiivisesta sonardatasta vedenalaisessa valvonnassa. Tutkimuksessa raaka signaalidata muutettiin aika-taajuus muotoon ja luokiteltiin binäärisesti kohteeksi tai ei-kohteeksi. Tutkimuksessa tekoälyn kouluttamiseen käytetty data on kerätty kahdesta merikokeesta käyttäen ”echo-repeater” maalia, joka palauttaa aktiivisen mittainpulssin imitoiden sukellusvenettä. Koulutetun luokittelijan validointi toteutettiin kolmannella merikokeella, jotta saatiin spatiaalisesti ja temporaalisesti muutettua vaikuttavia ympäristötekijöitä ja siten testattua luokittelijan todellista toimivuutta. Tiivistelmän mukaan CNN luokittelija vähensi merkittävästi väärien tunnistusten määrää sekä suoriutui perinteistä, aiemmin kehitettyä piirrepohjaista luokittelijaa paremmin. [9] Koko tutkimus ei ole saatavilla, sillä Naton tutkimuskeskus mahdollistaa tutkimuksen lataamisen vain jäsenmaille.

Koska alustava testaus spektrogrammeilla ja konvoluutioneuroverkoilla ei tuottanut avoimista lähteistä haalitulla data-aineistolla erityisen hyviä tuloksia, tarkasteltiin Gormanin ym käyttämää data-aineistoa muilla kuin heidän käyttämillään menetelmillä. Alkuperäisessä tutkimuksessa on käytetty hyvin yksinkertaista neuroverkkoa eri määrillä piilotettuja neuroneja, määrällisesti 0-24 kappaletta. Paras luokittelutulos on saatu 24 neuronilla. Data-aineisto on saatavilla muun muassa osoitteessa https://datahub.io/machine-learning/sonar.

Datan valmistelussa tarkasteltiin myös eri attribuuttien merkitystä havaintojen varianssin selittämisessä. Suoritettaessa PCA yli 90% selittävyyden kannalta havaittiin 11 attribuutin selittävän yli 90% piirreavaruuden varianssista. Aiempien, kuvasta 4 tehtyjen havaintojen ohella nämä 11 attribuuttia ovat yhteneviä selitetyn varianssin näkökulmasta, sillä selittävimmät attribuutit ovat numerot 11, 17, 19, 20, 23, 27, 29, 35, 37, 39 ja 45.

Mallinnus

Koska data-aineiston 208 näytettä koostuvat kukin 60 vakioidusta taajuusattribuutista, on kyseessä saman mittaisten aikasarjojen piirreavaruuksien erottelu. Tähän dataan testattiin muun muassa klassista perseptronia, kolmenlaisia implementaatioita SVM algoritmista, logistista regressiota sekä kahdenlaisia, yksinkertaisia RNN neuroverkkoja ja

monikerroksista, 24 neuronin tavanomaista, kolmikerroksista neuroverkkoa. Tulosten validointi toteutettiin alkuperäisen tutkimuksen mukaisesti tavanomaisille koneoppimisalgoritmeille jakamalla data 13 joukkoon k-fold metodilla ja toteuttaen mallin kouluttamisen käyttäen n-1 joukkoa yhden joukon toimiessa testijoukkona. Kuten alkuperäisessä tutkimuksessa on havaittu, satunnaisesti jaoteltu data tuottaa erittäin vaihtelevia tuloksia validoinnissa. Koska data koostuu eri mittauskulmista otetuista näytteistä, riippuu koulutetun mallin tarkkuus näytteiden jakautumisesta koulutus- ja validointijoukon välillä. Yksinkertaisella perseptron-algoritmilla voidaan saavuttaa esimerkiksi 0.90 testaustarkkuus, kun data on jaettu 90-10 suhteessa satunnaisiin koulutus- ja testausjoukkoihin. Kuitenkin suoritettaessa 13 kertainen k-fold validaatio, perseptronin tarkkuus jää, implementaatiosta riippuen 0.56 ja 0.69 väliin, sillä data ei ole lineaarisesti separoitavissa ja datanäytteissä on merkittäviä eroja mittauskulmista riippuen. Siinä missä Gorman ja Sejnowski ovat tulleet tulokseen, että on hyödyllistä käyttää mittauskulmatietoa koulutus- ja validointidatan jaottelussa paremman tarkkuuden saavuttamiseksi, voidaan todeta, ettei yllä olevin perusteluin ole järkevää olettaa vedenalaisessa valvonnassa olevan tiedossa, mistä kulmasta kohdetta mitataan sen orientaation suhteen. Tästä syystä erilaisten metodien testaamisessa ei pyritty rypästämään dataa mittauskulmien mukaan ja siten tasaamaan validointituloksia.

Alla olevassa taulukossa on esitetty keskiarvoiset testaustulokset koneoppimismetodeilta 13 k-fold iteraatiolla eri versioineen.

TAULUKKO 2: Keskiarvoiset tulokset 13 kertaisella k-fold validoinnilla

Menetelmä	**Validointitarkkuus**
K-Nearest Neighbours	85.7%
Perseptron (yksinkertaistettu)	56.7%
Perseptron (Sklearn versio)	69.2%
SVM (yksinkertaistettu)	53.3%
SVM dual	55.7%
SVM (Sklearn versio)	80.3%
Logistinen regressio	91.2%
Yksinkertainen RNN	80.77%
LSTM	73.08%
ANN	92.79%

Evaluointi

Koska kaikki data evaluointia varten on jo käytetty, eikä mallia olla viemässä tuotantoon, on evaluointi tehtävä arvioimalla alkuperäisen hypoteesin soveltuvuutta saavutettujen tuloksien valoissa, huomioiden käytetyn datan rajoitteet.

Tarkasteltaessa taulukon 2 tuloksia, havaitaan, että keskiarvoisesti paras menetelmä on perinteinen neuroverkko, kuten alkuperäisessäkin tutkimuksessa. Kuitenkin yksinkertainen logistinen regressio yltää lähes identtiseen tarkkuuteen huomattavasti yksinkertaisemmalla laskennalla, ja lisäksi K-NN ja Sklearn-kirjaston SVM pääsevät varsin lähelle jopa ilman erinäistä optimointia, sillä algoritmeja käytettiin näiden perusasetuksin.

Tämän tarkastelun suurin ongelma on datan vähyys, sillä 208 datapistettä on erittäin vaatimaton näyte. Lisäksi data-aineisto on kerätty temporaalispatiaalisesti staattisesta ympäristöstä, jolloin on oletettavaa, että parhainkin malli soveltuu huonosti todelliseen käyttöön.

Käyttökelpoisen mallin kouluttaminen edellyttäisi laajamittaisen data-aineiston keräämistä mahdollisimman monista vesiääniolosuhteista, eri etäisyyksillä ja mittausparametreilla. Iso vaikutus on Itämeren ominaispiirteillä, sillä vuodenaikojen mukaan vaihtuvat vesiääniolosuhteet ja matalan veden aiheuttamat paikalliset tekijät kuten pohjan muodot vaikuttavat dataan merkittävästi. Tätä varten olisi käytettävä esimerkiksi Naton tutkimuksen tapaan vaihtoehtoista maalia eli responderia, joka pitäisi konfiguroida vastaamaan sukellusveneen aiheuttamaa paluukaikua. Kattavasti kerätystä datasta pitäisi kyetä paikantamaan mielenkiintoiset ilmiöt ja korostamaan ne erilleen aikasarjasta. Käytännössä data voidaan oletettavasti muodostaa paloittelemalla aikasarjat osiin valitun kynnysarvon mukaan, sillä pisinkään mielenkiintoinen vedenalainen kohde ei ole yli 200 metriä pitkä. Näin ollen aikasarjan pilkkominen esimerkiksi 200 millisekunnin näytteisiin asetetun havaitsemiskynnyksen ylittyessä mahdollistaisi kiinnostavien näytteiden eristämisen ja erillisen tarkastelun. Näitä näytteitä sen sijaan voitaisiin, tämän kokeellisen osuuden valossa, luokitella joko perinteisellä neuroverkolla, logistisella regressiolla tai SVM-luokittelijalla, tai sitten näiden menetelmien yhdistelmällä.

LÄHTEET

[1] Payne C.M., *Principles of Naval Weapon Systems, Second Edition*, Naval Institute Press, 291 Wood Road, Annapolis, MD 21402, ISBN 978-1-59114-667-4

[2] Urick R.J., *Principles of Underwater Sound for Engineers*, McGraw-Hill Book Company, New York, 1967, ISBN 0-07-066086-7

[3] Kongsberg Maritime AS, *ST2400 Anti Submarine Warfare Sonar VDS*, Strandpromenaden 50, P.O. Box 111 N-3191 Horten, Norja, viitattu 16.1.2021, saatavilla: https://www.kongsberg.com/globalassets/maritime/km-products/product-documents/anti-submarine-warfare-sonar-st-2400

[4] Sonar Basics: *How to read Sonar image*, Furuno, viitattu 6.3.2022, saatavilla: https://www.furuno.com/en/technology/sonar/basic/view

[5] Seppänen R., Kervinen M., Parkkila I., Karkela L., Meriläinen P., *maol taulukot*, Otavan Kirjapaino Oy, Keuruu 2007, ISBN-13: 978-951-1-20607-1

[6] Friedman N, *Littoral Anti-Submarine Warfare: Not as Easy as it Sounds*, Jane's International Defence Review, kesäkuu 1995

[7] Allan D. P., *Acoustics: An Introduction to Its Physical Principles and Applications*, Vol 3rd ed. 2019. Springer; 2019. Viitattu 181.2021. Saatavilla: http://search.ebscohost.com.mp-envoy.csc.fi/login.aspx?direct=true&db=nlebk&AN=2536838&site=ehost-live

[8] Gorman P.R., Sejnowski T.J., *Analysis of Hidden Units in a Layered Network Trained to Classify Sonar Targets*, Neural Networks Vol 1, pp. 75-89, Pergamon Journals Ltd. 1988, 0893-6080/88

[9] De Magistris G., Stinco P., Canepa G., Ferri G., Fois M., Tesei A., LePage K.D., *Big data exploitation for active sonar contact-level classification*, 11.5.2020, CMRE-FR-2019-005, viitattu 5.1.2022, saatavilla: https://www.cmre.nato.int/research/publications/latest-techreports/1546-big-data-exploitation-for-active-sonar-contact-level-classification-1-1

```
data <- read.csv("sonar_csv.csv")
rock <- data[data$Class == "Rock",]
mine <- data[data$Class == "Mine",]

rock <- rock[1:60]
mine <- mine[1:60]
f <- seq(1,60,1)

rock_sum <- colSums(rock)
mine_sum <- colSums(mine)

rs <- rock_sum/max(rock_sum)
ms <- mine_sum/max(mine_sum)

rock_means = colMeans(rock)
mine_means = colMeans(mine)

boxplot(rock, col='blue', main="Gorman & Sejnowski sonar data",
          ylab="Amplitudi",
          names=seq(1:60))
legend("topleft", legend=c("Kivi"))

boxplot(mine, col='red',
          xlab="Taajuusväli",
          ylab="Amplitudi",
          names=seq(1:60))
legend("topleft", legend=c("Sylinteri"))

plot(rs, type='l', col='blue', main="Gorman & Sejnowski sonar data",
            xlab="Taajuusattribuutti",
            ylab="Amplitudi",)
            lines(ms, col='red')
          legend("topleft", legend=c("Sylinteri", "Kivi"),
          col=c("red", "blue"), lty=1:2, cex=0.8)

plot(rock_means, type='l', col='blue',
          main="Gorman & Sejnowski keskiarvoinen sonar data",
          xlab="Taajuusattribuutti",
          ylab="Amplitudi",)
          lines(f, mine_means, col='red')
          legend("topleft", legend=c("Sylinteri", "Kivi"),
          col=c("red", "blue"), lty=1:2, cex=0.8)
```

Data saatavilla: archive.ics.uci.edu/ml/datasets/connectionist+bench+(sonar,+mines+vs.+rocks)
Alkuperäinen lähde: Gorman R.P., Sejnowski T.J., *"Analysis of Hidden Units in a Layered Network Trained to Classify Sonar Targets",* 1988, Neural Networks, Vol. 1, s. 75-89

Tässä liitteessä on liitteen 3 Python-lähdekoodi, johon on lisätty neuroverkkojen rakennekoonnokset sekä lopulliset tulokset.

```
# -*- coding: utf-8 -*-

import pandas as pd
import numpy as np
import os

df = pd.read_csv('sonar_csv.csv')

results = {}
```

Havainnollistavana esimerkkinä tässä liitteessä on itse implementoidut versiot perseptroni-algoritmista sekä kahdesta versiosta toteuttaa tukivektorikone.

```
class Perceptron:

    def __init__(self, t):
        self.w = None
        self.threshold = t

    def model(self, x):
        return 1 if(np.dot(self.w.T, x) >= self.threshold) else -1

    def predict(self, X):
        classified = []
        for x in X:
            result = self.model(x)
            classified.append(result)

        return np.array(classified)

    def fit(self, X, Y, epochs):
        self.w = np.zeros(X.shape[1])

        stop = False
        t = 0

        while(stop = False):
          if t >= epochs:
            stop = True
            break
          for x, y in zip(X,Y):
            xyw = np.dot(self.w.T, x)*y

            if np.sum(xyw) <= 0:
              self.w = self.w + x*y
          t = t + 1


    def get_weight(self):
        return self.w
```

Toteutetun perseptron algoritmin kouluttamisen (*fit*) toimintaperiaate:

```
aseta suoritettavien iteraatioiden määrä
aseta vektorin w painoiksi 0
    jos i < iteraatiot, toista:
            jos y_i w^T x_i ≤ 0:
                    w = w + y_i x_i
            muuten:
                    lopeta
            i = i + 1
```

SVM algoritmin kouluttamisen (*fit*) toimintaperiaate:

```
aseta suoritettavien iteraatioiden määrä n
asetetaan vektorin w painoiksi 0
m = pituus(w)
jos i < n, toista:
            jos j < m:
                    näyte (x_i, y_i) = (X[j], y[j])
                    jos y_i w^T x_i < 1:
                            w = −y_i x_i + λw
                    muuten:
                            w = λw
            i = i + 1
```

SVM dual algoritmin kouluttamisen (*fit*) toimintaperiaate:

```
aseta suoritettavien iteraatioiden määrä n
asetetaan α = 0, C = c
m = pituus(w)
jos i < n:
            jos j < m:
                    näyte (x_i, y_i) = (X[j], y[j])
                    α_i = (1 − y_i Σ_{j≠i} α_i y_i x_i^T x_j) / (x_i^T x_j)
                    α_i = min(C / m, max(0, α_i))
```

Algoritmit on mukailtu Mohri ym. kirjan *Foundations of Machine Learning – second edition* mukaisesti [1 s. 79-85, 190, 207]. Yllä kuvattujen pseudokoodien on tarkoitus havainnollistaa kirjastomenetelmien tekemää laskentaa painojen säätämisessä ratkaisun löytämiseksi, joka muutoin tapahtuu menetelmien sisällä jääden lukijalta näkymättömiin. Kyseiset mallit ovat yksinkertaisia versioita hienostuneemmista menetelmistä. Tärkein huomio lukijalle on koneoppimisen perusolemuksen ymmärtäminen painokertoimien oppimisessa. Kuvatuissa virhefunktioissa painokertoimia säädetään niin kauan, kuin mallissa on väärin luokiteltuja havaintoja, pyrkien separoitavuuteen. Pysäytysehto on ennalta määrätty iteraatioiden lukumäärä. Koska data ei ole lineaarisesti separoitavissa onnistuneesti, jää kyseisten menetelmien luokitustarkkuus vaatimattomaksi.

```
class SVM():

    def __init__(self, xlambda, eta):
        self.w = np.zeros(60)
        self.X = X
        self.y = y
        self.eta= eta
        self.xlambda = xlambda
        self.t = 1

    def model(self, x):
        return 1 if(np.dot(self.w.T, x) > 1) else -1

    def threshold(self, x, y):
        d = np.dot(self.w.T, x)*y
        jiw = np.zeros(60)
        if(d < 1):
            jiw = -y*x+self.xlambda*self.w.T
        else:
            jiw = self.xlambda*self.w.T

        self.w = self.w - self.eta*jiw

    def predict(self, X):
        classified = []

        for x in X:
            result = self.model(x)
            classified.append(result)

        return np.array(classified)

    def fit(self, X, y):
        m = X.shape[0]

        for i in range(10):
            for j in range(m):
                x_sample = X[j]
                y_sample = y[j]

                self.threshold(x_sample, y_sample)

    def get_weights(self):
        return self.w

    def get_norm(self):
        normv = np.linalg.norm(self.w)

        return normv
```

```
class SVM_dual():

    def __init__(self, m, x, y, C):
        self.w = np.zeros(60)
        self.a = np.zeros(m)
        self.x = x
        self.y = y
        self.m = m
        self.C = C

    def model(self, x):
        self.w = self.weight()
        return 1 if(np.dot(self.w.T, x) > 1) else -1

    def predict(self, X):
        classified = []

        for x in X:
            result = self.model(x)
            classified.append(result)

        return np.array(classified)

    def alpha(self, i):
        sigma = 0
        for j in range(self.m):
            if i != j:
                s = self.a[j]*self.y[j]*np.inner(self.x[i].T, self.x[j])
                sigma = sigma + s

        ai = (1-sigma*self.y[i])/np.inner(self.x[i].T, self.x[i])

        self.a[i] = min(self.C/self.m, max(0, ai))

    def fit(self, n):
        for i in range(n):
            for j in range(self.m):
                self.alpha(j)

    def weight(self):
        w = 0

        for i in range(self.m):
            w = w+self.a[i]*self.y[i]*self.x[i]

        return w

    def get_ai(self):
        return self.a
```

Koodissa erotetaan luokitukset vektoriin y, jolloin datan attribuutit jäävät x vektoreihin.

```
y = df.loc[:,"Class"]

y.replace({"Rock": -1, "Mine": 1}, inplace=True)

df = df.drop(["Class"], axis=1)

from sklearn import svm
from sklearn.model_selection import LeaveOneOut
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

loo = LeaveOneOut()

X = df.to_numpy()
y = y.to_numpy()

l = []

for train_index, test_index in loo.split(X):
    clf = svm.SVC()
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    score = accuracy_score(y_test, y_pred, normalize=False)
    l.append(score)

print(np.mean(l))

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1,
random_state=42)
```

Kaikki testatut menetelmät, paitsi k-Nearest Neighbors, käydään läpi Gorman ym. alkuperäisen tutkimuksen tyyliin k-fold validoinnilla. Vertailun vuoksi testattiin myös LeaveOneOut (loo) validointi, joka on testattu yllä. Samoin data jaettiin aluksi kiinteisiin koulutus- ja validointijoukkoihin, mutta datan vinoumien johdosta luokitustarkkuudet eivät kuvastaneet todellisuutta, vaan esimerkiksi yksinkertainen Perceptron-luokka saavutti jopa 90% tarkkuuden kiinteästi jaetulla datalla.

Vertailukohdaksi toteutettiin edellä mainittu KNN, joka on toteutettu alla. Lisäksi alustettiin k-fold joukot, tässä tapauksessa k=13 Gorman ym. alkuperäisen tutkimuksen mukaisesti.

```
from sklearn.model_selection import KFold
kf = KFold(n_splits=13, shuffle=True, random_state=0)

from sklearn.cluster import KMeans

kmeans = KMeans(n_clusters=2, random_state=42).fit(X)

from sklearn.metrics.cluster import normalized_mutual_info_score as
nmi

print(nmi(y, y_means))

from sklearn.neighbors import KNeighborsClassifier

neighbor = KNeighborsClassifier(n_neighbors=2)
neighbor.fit(X_train, y_train)
print(accuracy_score(y_test, neighbor.predict(X_test)))
```

Alla toteutettiin k-fold validointi kahdelle perseptron-algoritmille.

```
sc = 0

for train_index, test_index in kf.split(X):
  perceptron = Perceptron(0)
  X_tr, X_t = X[train_index], X[test_index]
  y_tr, y_t = y[train_index], y[test_index]
  perceptron.fit(X_tr, y_tr, 1000)
  perceptron_prediction = perceptron.predict(X_t)
  score = accuracy_score(y_t, perceptron_prediction)
  sc = sc + score

results["Perceptron"] = sc/13

perceptron = Perceptron(0)
perceptron.fit(X_train, y_train, 250)

from sklearn.linear_model import Perceptron as percep

sc = 0

for train_index, test_index in kf.split(X):
  perceptron = percep()
  X_tr, X_t = X[train_index], X[test_index]
  y_tr, y_t = y[train_index], y[test_index]
  perceptron.fit(X_tr, y_tr)
  perceptron_prediction = perceptron.predict(X_t)
  score = accuracy_score(y_t, perceptron_prediction)
  sc = sc + score

results["Sklearn Perceptron"] = sc/13
```

```
eta = 0.1
xlambda = 0.01
sc = 0

for train_index, test_index in kf.split(X):
  svm_model = SVM(xlambda, eta)
  X_tr, X_t = X[train_index], X[test_index]
  y_tr, y_t = y[train_index], y[test_index]
  svm_model.fit(X_tr, y_tr)
  svm_pred = svm_model.predict(X_t)
  score = accuracy_score(y_t, svm_pred)
  print(score)
  sc = sc + score

results["SVMi"] = sc/13

C = 1000
sc = 0

for train_index, test_index in kf.split(X):
  X_tr, X_t = X[train_index], X[test_index]
  y_tr, y_t = y[train_index], y[test_index]
  mdata,ndim = X_tr.shape
  svc = SVM_dual(mdata, X_tr, y_tr, C)
  svc.fit(100)
  svc_pred = svc.predict(X_t)
  score = accuracy_score(y_t, svc_pred)
  print(score)
  sc = sc + score

results["SVM_dual"] = sc/13

sc = 0

for train_index, test_index in kf.split(X):
  X_tr, X_t = X[train_index], X[test_index]
  y_tr, y_t = y[train_index], y[test_index]
  clf.fit(X_tr, y_tr)
  y_pred = clf.predict(X_t)
  score = accuracy_score(y_t, y_pred)
  sc = sc + score

results["Sklearn SVM"] = sc/13
```

```
from sklearn.decomposition import PCA

pca = PCA(n_components=60)
pca.fit(X)

sum = 0
expvar = pca.explained_variance_ratio_

for i in range(61):
    sum = sum + expvar[i]
    if(sum > 0.9):
        print(i)
        break

model = PCA(n_components=11).fit(X)
reduced = model.transform(X)

n_pcs = model.components_.shape[0]

most_important = [np.abs(model.components_[i]).argmax() for i in
range(11)]

feature_names = df.columns

most_important_features = [feature_names[most_important[i]] for i in
range(11)]

selected = df.loc[:,most_important_features]

selected = selected.to_numpy()

sc = 0

for train_index, test_index in kf.split(selected):
  perceptron = Perceptron(0)
  X_tr, X_t = selected[train_index], selected[test_index]
  y_tr, y_t = y[train_index], y[test_index]
  perceptron.fit(X_tr, y_tr, 1000)
  perceptron_prediction = perceptron.predict(X_t)
  score = accuracy_score(y_t, perceptron_prediction)
  sc = sc + score

results["Perceptron PCA"] = sc/13

r_train, r_test, y_train, y_test = train_test_split(selected, y,
test_size=0.1, random_state=42)

clf = svm.SVC()
clf.fit(r_train, y_train)

yp = clf.predict(r_test)

score = accuracy_score(yp, y_test)
results["Sklearn SVM reduced"] = score
```

```
from sklearn.linear_model import LogisticRegression

sc = 0

for train_index, test_index in kf.split(X):
  X_tr, X_t = X[train_index], X[test_index]
  y_tr, y_t = y[train_index], y[test_index]
  clf = LogisticRegression(random_state=0).fit(X_tr, y_tr)
  ypredicted = clf.predict(X_test)
  score = accuracy_score(y_test, ypredicted)
  print(score)
  sc = sc + score

results["Logistic regression"] = sc/13
```

Seuraavaksi testattiin takaisinkytkettyjä neuroverkkoja, sekä yksinkertaisena (SimpleRNN) että Long-Short Term Memory (LSTM) versioina.

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, SimpleRNN
from sklearn.metrics import mean_squared_error

X_rnntrain = X_train.reshape(187, 60, 1)

X_rnn = X.reshape(208, 60, 1)

print(X_rnntrain.shape)

y_train = np.where(y_train < 0, 0, y_train)

y_test = np.where(y_test < 0, 0, y_test)

y_rnn = np.where(y < 0, 0, y)

print(y_rnn)

model = Sequential()

model.add(SimpleRNN(10, activation="tanh"))
model.add(Dense(units=1, activation="sigmoid"))
model.compile(loss="mse", optimizer="adam", metrics=['accuracy'])
```

```
Model: "sequential_9"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 simple_rnn_3 (SimpleRNN)    (None, 10)                120

 dense_18 (Dense)            (None, 1)                 11

=================================================================
Total params: 131
Trainable params: 131
Non-trainable params: 0
```

```
sc = 0

for train_index, test_index in kf.split(X_rnn):
  X_tr, X_t = X_rnn[train_index], X_rnn[test_index]
  y_tr, y_t = y_rnn[train_index], y_rnn[test_index]
  model.fit(X_tr, y_tr, epochs=25, batch_size=1)
  prediction = model.predict(X_t)
  prediction = np.where(prediction < 0.5, 0, 1)
  score = accuracy_score(y_t, prediction)
  print(score)
  sc = sc + score

results["Simple RNN 13-fold"] = sc/13

model.summary()

from tensorflow.keras.layers import LSTM
from tensorflow.keras.layers import Dropout

import re
lstm = Sequential()

lstm.add(LSTM(8, activation="tanh", recurrent_activation='sigmoid',
dropout=0.2, recurrent_dropout=0.2))
lstm.add(Dense(units=1, activation="sigmoid"))
lstm.compile(loss="binary_crossentropy", optimizer="adam",
metrics=['accuracy'])


lstm.summary()
```

Model: "LSTM"

Layer (type)	Output Shape	Param #
lstm_3 (LSTM)	(None, 8)	320
dense_19 (Dense)	(None, 1)	9

Total params: 329
Trainable params: 329
Non-trainable params: 0

```
sc = 0

for train_index, test_index in kf.split(X_rnn):
  X_tr, X_t = X_rnn[train_index], X_rnn[test_index]
  y_tr, y_t = y_rnn[train_index], y_rnn[test_index]
  lstm.fit(X_tr, y_tr, epochs=25, batch_size=1)
  prediction = lstm.predict(X_t)
  prediction = np.where(prediction < 0.5, 0, 1)
  score = accuracy_score(y_t, prediction)
  print(score)
  sc = sc + score

results["LSTM 13-fold"] = sc/13
```

```
nn = Sequential()

nn.add(Dense(units=12, activation="relu"))
nn.add(Dense(units=8, activation="relu"))
nn.add(Dense(units=4, activation="relu"))
nn.add(Dense(units=1, activation="sigmoid"))

nn.compile(optimizer='adam', loss='mse', metrics=['accuracy'])

nn.summary()
```

```
Model: "ANN"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_28 (Dense)            (None, 12)                732

 dense_29 (Dense)            (None, 8)                 104

 dense_30 (Dense)            (None, 4)                 36

 dense_31 (Dense)            (None, 1)                 5

=================================================================
Total params: 877
Trainable params: 877
Non-trainable params: 0
_________________________________________________________________
```

```
sc = 0

for train_index, test_index in kf.split(X):
  X_tr, X_t = X[train_index], X[test_index]
  y_tr, y_t = y_rnn[train_index], y_rnn[test_index]
  nn.fit(X_tr, y_tr, epochs=25, batch_size=1)
  print(X_tr.shape)
  prediction = nn.predict(X_t)
  print(prediction.shape)
  prediction = np.where(prediction < 0.5, 0, 1)
  score = accuracy_score(y_t, prediction)
  print(score)
  sc = sc + score

results["ANN 13-fold"] = sc/13
print(results)
```

Lopulta saadaan tulostettua kaikkien testattujen menetelmien k-fold validoitujen tulosten keskiarvot.

```
{'Perceptron': 0.5673076923076923,
'Sklearn Perceptron':  0.6923076923076923,
'SVMi':                0.5336538461538461,
'SVM_dual':            0.5576923076923077,
'Sklearn SVM':         0.8028846153846154,
'Perceptron PCA':      0.5384615384615384,
'Logistic regression': 0.9120879120879123,
'Simple RNN':          0.9047619047619048,
```

```
'LSTM RNN':             0.6666666666666666,
'ANN':                  0.8571428571428571,
'Simple RNN 13-fold':  0.8076923076923077,
'LSTM 13-fold':         0.7307692307692307,
'ANN 13-fold':          0.9278846153846154}
```

LÄHTEET

[1] Mohri M., Rostamizadeh A., Talwalkar A., *Foundations of Machine Learning – second edition,* The MIT Press, Cambridge, Massachusetts USA, 2018, ISBN 9780262039406

Pinta-alusten automaattinen tunnistaminen

Alusluokkien kuvien tunnistaminen konenäöllä

Meritilannekuvan ylläpitäminen edellyttää useiden sensorien käyttöä. Meri- eli pintatilannekuvan muodostamiseen käytettäviä sensoreja ovat valvontatutkat, AIS-vastaanottimet (*Automated Identification System*) sekä kamerajärjestelmät niin näkyvän valon kuin infrapunan alueella. Tutkimuksen luvussa 3 on kuvattu näiden muodostamien datapisteiden ominaisuuksia ja eroja. Tässä liitteessä on tarkoitus demonstroida konvoluutioneuroverkkojen käyttöä kuvien tunnistamisessa. Varmennettu alustunnistus edellyttää käytännössä kuvaa tunnistettavasta kohteesta, etenkin yksilötunnistusta varten. Luokkatunnistukseen voidaan päästä myös elektronisen sormenjäljen perusteella, mutta elektronisen tiedustelun dataa ei käytetä tämän opinnäytetyön puitteissa.

Kuvien tunnistaminen tekoälyn avulla, eli niin kutsuttu konenäkö, on kattavasti tutkittu tekoälyn osa-alue. Tässä liitteessä havainnollistetaan lyhyessä ajassa koostetun kuva-aineiston luokittelemista eri menetelmin, menetelmien tulosten eroja sekä esitetään toimenpiteitä mallien kehittämisen edistämiseksi.

Liitteessä esitetyt kuvat ovat lähtökohtaisesti Merivoimien omia kuvia, ellei erillistä viittausta ole mainittu.

Datan ymmärrys

Data-aineisto koostettiin kahdesta lähteestä, avoimista lähteistä haettuina kuvina alusluokkien edustajista, sekä Merivoimien taltioimista tunnistus- ja edustuskuvista, jotka irrotettiin kontekstistaan ja hyödynnettiin julkisena datamateriaalina. Tekoälymallien luomiseen käytettiin kuvia seitsemästä eri alusluokasta, jotka olivat Hamina-, Kilo, Pomornik-, Karakurt-, Rauma-, Ropucha- sekä Stereguštši-luokka. Kuvat ovat merellisestä ympäristöstä, Merivoimien taltioimien kuvien osalta niin kamera kuin IR-kuvia ja avoimien lähteiden osalta pelkästään valokuvina. Suurin osa avoimista kuvista on satamista tai satama-alueen läheisyydestä. Merivoimien ottamat kuvat ovat todellisesta operatiivisesta ympäristöstä. Kuvissa on jonkin verran sekä temporaalista että spatiaalista hajontaa, mutta kuvat painottuvat purjehduskaudelle keväästä syksyyn, sekä ovat usein samasta kuvaustilanteesta, omaten vain lieviä variaatioita kuvien välillä kyseisessä kuvaustilanteessa. Tämä vaikuttaa osiltaan saatavien mallien jykevyyteen (*"robustness"*). Taulukossa 1 on listattu, montako näytettä aineistossa on per alusluokka.

TAULUKKO 1: Kuvien määrän jakautuminen alusluokittain

Alusluokka	**Kuvien lukumäärä**
Hamina	138
Kilo	134
Pomornik	59
Karakurt	53
Rauma	58
Ropucha	104
Stereguštši	177
Yhteensä	723

Kuten taulukosta 1 nähdään, ovat niin Pomornik-, Project 22800 eli Karakurt- kuin Rauma-luokka aliedustettuina aineistossa, joskin keskenään samansuuruisissa määrissä. Tästä voidaan tehdä oletus, että näiden alusluokkien piirreavaruuden oppiminen onnistuu huonommin kuin suuremman näytemäärän omaavien alusluokkien kohdalla, etenkin, jos piirteissä on konvoluutioneuroverkkojen näkökulmasta huomattavaa samankaltaisuutta.

Data koostettiin valmiiksi alusluokkakohtaisiin kansioihin, jotta se voidaan hyödyntää yksinkertaisesti kansiorakenteen mukaisesti. Seuraavaksi data esikäsiteltiin Tensorflow/Keras ohjelmistokirjastojen menetelmin hyödynnettäviin koulutus- ja validointijoukkoihin sekä yhteneviin dimensioihin.

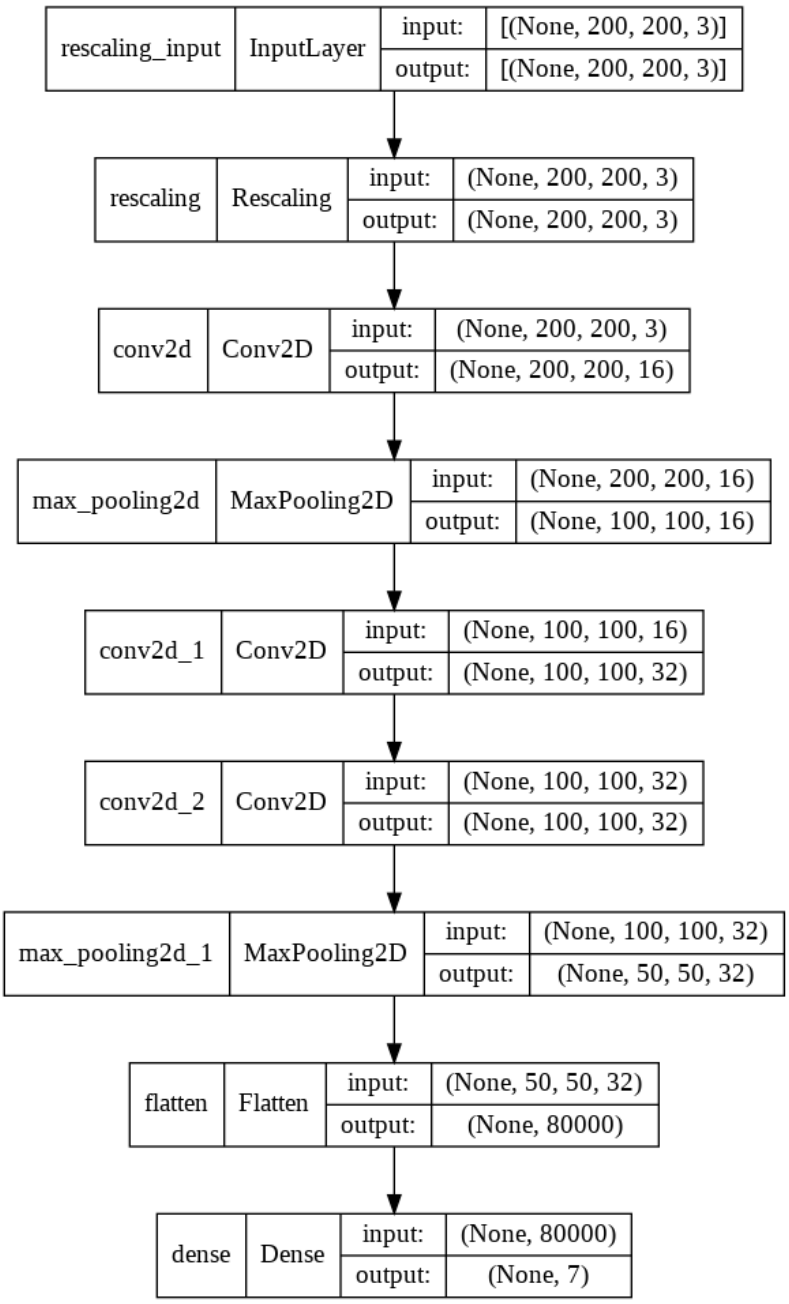
Datan valmistelu

Suurin vaiva datan valmistelussa muodostui data-aineiston manuaalisesta kokoamisesta. Internetlähteistä on mahdollista tuottaa kuva-aineistoa yksinkertaisilla skripteillä, mutta aineisto pitää joka tapauksessa käydä läpi, sillä tulokset perustuvat hakujen tuloksiin, joiden annotaatiot on varmennettava. Merivoimien keräämän kuva-aineiston osalta kuvat on kansioitu näytteittäin luotettavasti, mutta esimerkiksi nimeämiskäytännöt aiheuttavat päällekkäisyyksiä. Lisäksi kuvat sijaitsevat polveilevissa kansiorakenteissa, joissa etsintä ja haarukointi täytyy toteuttaa manuaalisesti. Suljetuissa ympäristöissä, joissa kuvia on, ei voi myöskään käyttää yhtä vapaasti ohjelmistotyökaluja kuvien helppoon, massamaiseen koostamiseen ja esimerkiksi uudelleennimeämisiin. Tästä johtuen koostaminen oli hidasta tiedostojen uudelleennimeämistä ja kansioimista manuaalisesti.

Käytettäviä menetelmiä varten data valmisteltiin muokkaamalla kuvien kokoa, tuottaen kaikista kuvista joko (200, 200) kokoisia versioita, tukivektorikoneita varten (150, 150) kokoisia ja siirto-oppimisessa VGG16 edellyttämän koon mukaisia. Datasta muodostettiin neuroverkkoja varten generaattorit ja tukivektorikoneita varten kiinteät koulutus- ja validointijoukot.

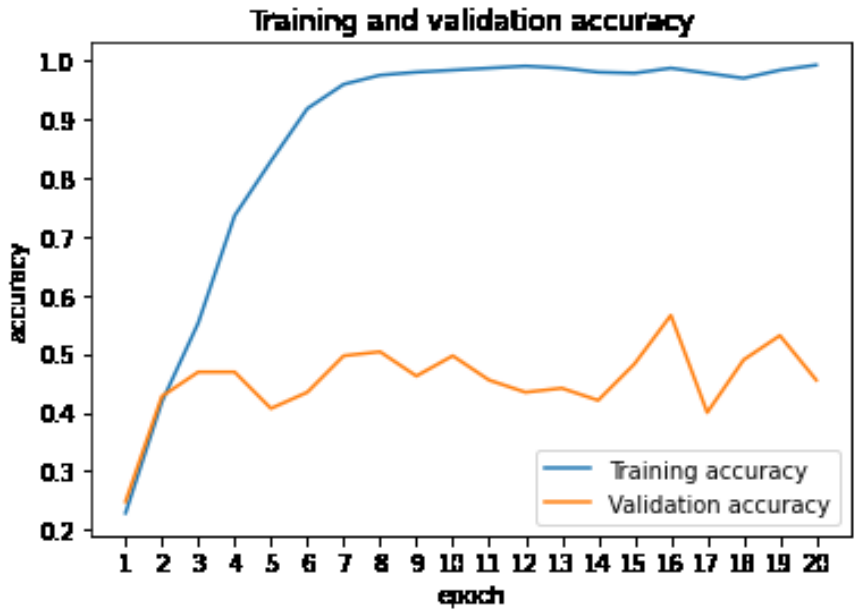
Mallinnus

Ensimmäiseksi testattiin yksinkertaisen, itse strukturoidun konvoluutioverkon suorituskykyä tällä data-aineistolla. Mallissa on yhteensä kolme konvoluutiokerrosta sekä kaksi pooling-kerrosta, jotka muodostavat ytimineen yhteensä 574 343 koulutettavaa parametria. Kouluttavien parametrien määrä on varsin suuri suhteessa data-aineiston määrään. Vähemmän koulutettavia parametreja sisältäviä rakenteita testattiin, mutta merkittävästi yksinkertaisemmat mallit lähtökohtaisesti alioppivat data-aineiston piirteet.



KUVA 1: Ensimmäisen neuroverkon visualisoitu rakenne

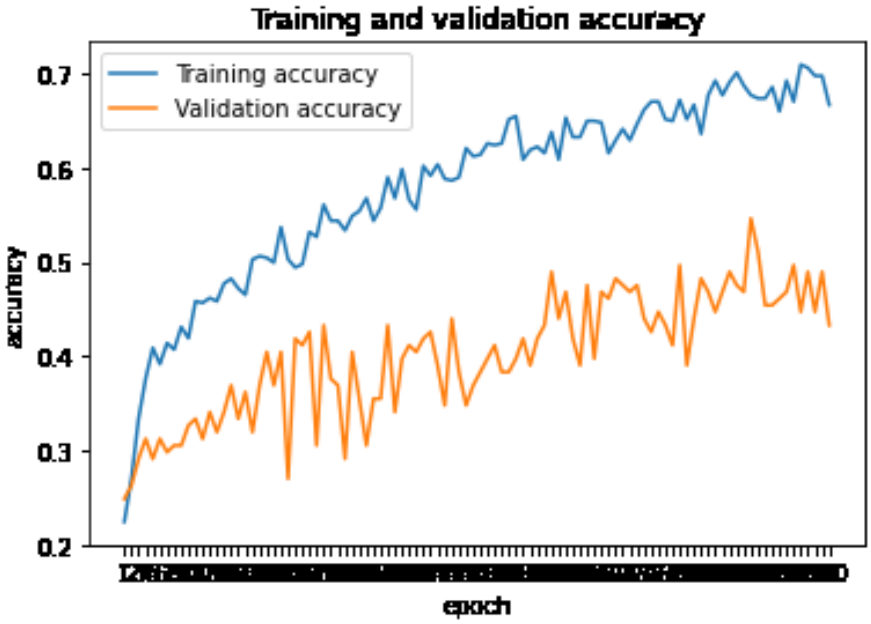
Testattaessa ensimmäistä mallia ilman data-augmentaatiota tapahtuu huomattavaa ylioppimista validointitarkkuuden jäädessä hyvin vaatimattomaksi. Epookeittain taltioitu tarkkuus sekä virhe koulutusjoukolla ja validointijoukolla on esitetty kuvassa 2.



KUVA 2: Kuvan 1 neuroverkon koulutushistoria 20 epookin ajalta

Neuroverkon tuottaman mallin saavuttaessa lähes 100% luokitustarkkuus koulutusjoukolla saavutetaan alle 50% luokitustarkkuus validointijoukolla. Tulos on malliesimerkki ylioppimisesta, joka johtuu pääasiassa datan määrästä suhteessa mallin monimutkaisuuteen: malli kyetään kouluttamaan siten, että se huomioi jokaisen koulutusnäytteen saavuttaen lähes täydellisen tarkkuuden kyseisellä datalla ja epäonnistuen surkeasti tästä poikkeavalla datalla.

Tästä syystä sama neuroverkko koulutettiin 100 epookin verran augmentoidulla datalla. Augmentoinnilla tuotettiin automaattisesti satunnaisia poikkeamia alkuperäisiin kuviin, tuottaen näin lisää näennäisesti uusia näytteitä neuroverkon opittavaksi.

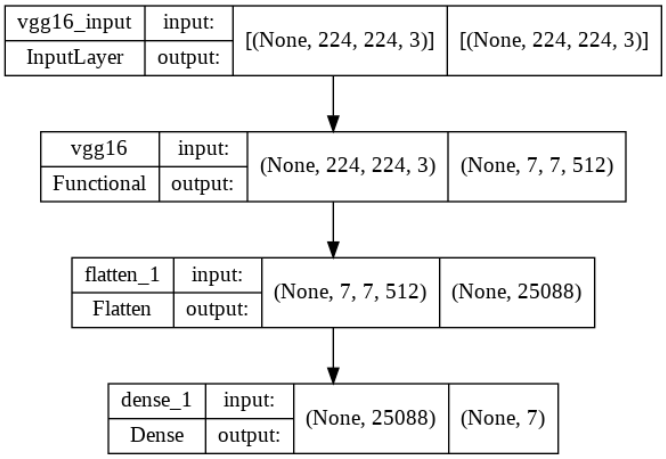


KUVA 3: Augmentoidulla datalla koulutetun neuroverkon koulutushistoria 100 epookin osalta

Kuten kuvasta 3 voidaan havaita, saavutetaan augmentoidulla datalla matalampi luokitustarkkuus kuin muokkaamattomalla aineistolla, mutta validointitarkkuus seuraa asteittain paranevaa koulutustarkkuutta paremmin kuin aiemmassa. Kuitenkin, 100 epookin jälkeen, ei ole vielä saavutettu merkittävää eroa validointitarkkuudessa verrattuna aiempaan.

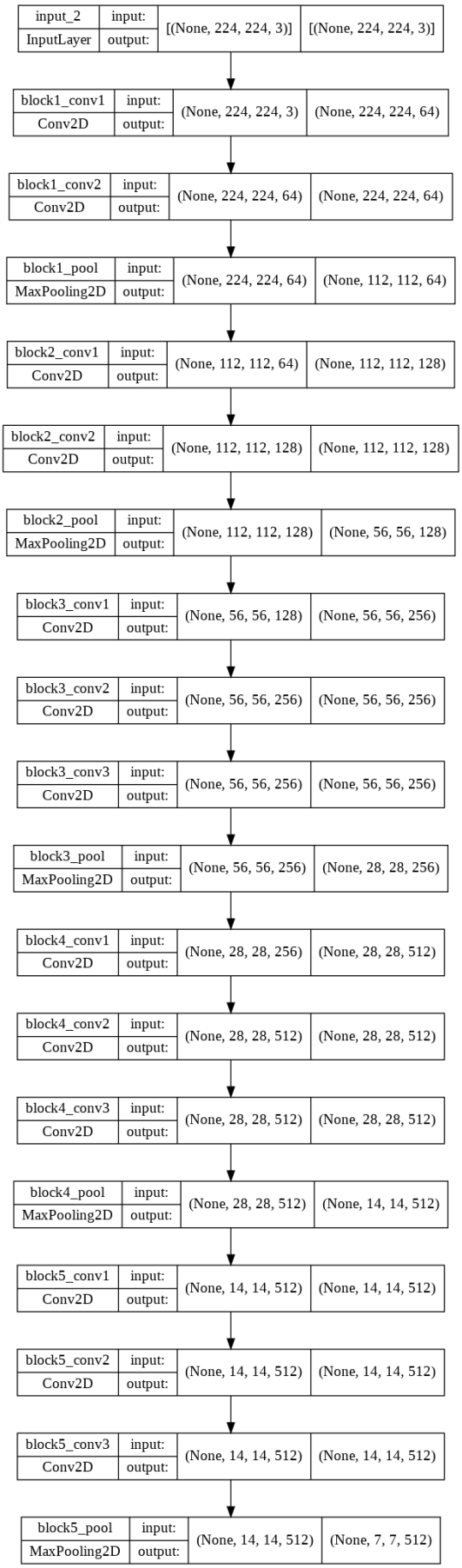
Konvoluutioverkkojen ohella testattiin tukivektorikoneita. Optimoimatta tukivektorikoneiden hyperparametreja saavutettiin 49.3% luokittelutarkkuus, jossa korkeimmat f1-arvot ja luokitustarkkuudet saavutettiin Hamina- ja Stereguštši-luokkien osalta. Optimoimalla hyperparametreja saavutettiin 54.8% luokitustarkkuus arvoilla {C: 10, gamma: 0.001, kernel: 'rbf'}. Näistä C on tukivektorikoneen tuottaman marginaalin leveys hypertasolla, gamma laskettavan tason kaareutuminen ja RBF ("*radial basis function"*) tyypillinen ydin piirreavaruuden projisoimiseksi korkeampaan dimensioon luokiteltavuuden parantamiseksi [1 s. 744-748]. Tulosten perusteella tukivektorikone ei onnistu tällä aineistolla tuottamaan riittävän hyvää luokittelijaa tällä kuva-aineistolla.

Koska data-aineisto on kooltaan vaatimaton, on seuraava looginen askel kokeilla siirto-oppimista. Siirto-oppimisella voidaan hyödyntää erittäin syvän neuroverkon jo oppimaa piirre-erottelukykyä, opettamalle niiden pohjalta uudet ylimmät kerrokset kyseistä data-aineistoa varten. Tätä tarkoitusta varten valittiin Oxfordin tutkimusryhmän kehittämä VGG16 rakenne, joka on koulutettu Imagenet – data-aineistolla saavuttaen erittäin hyvän suorituskyvyn miljoonien kuvien aineistosta ja siten hyvän pohjan erotella myös tämän vaatimattoman aineiston piirteitä toisistaan [2].



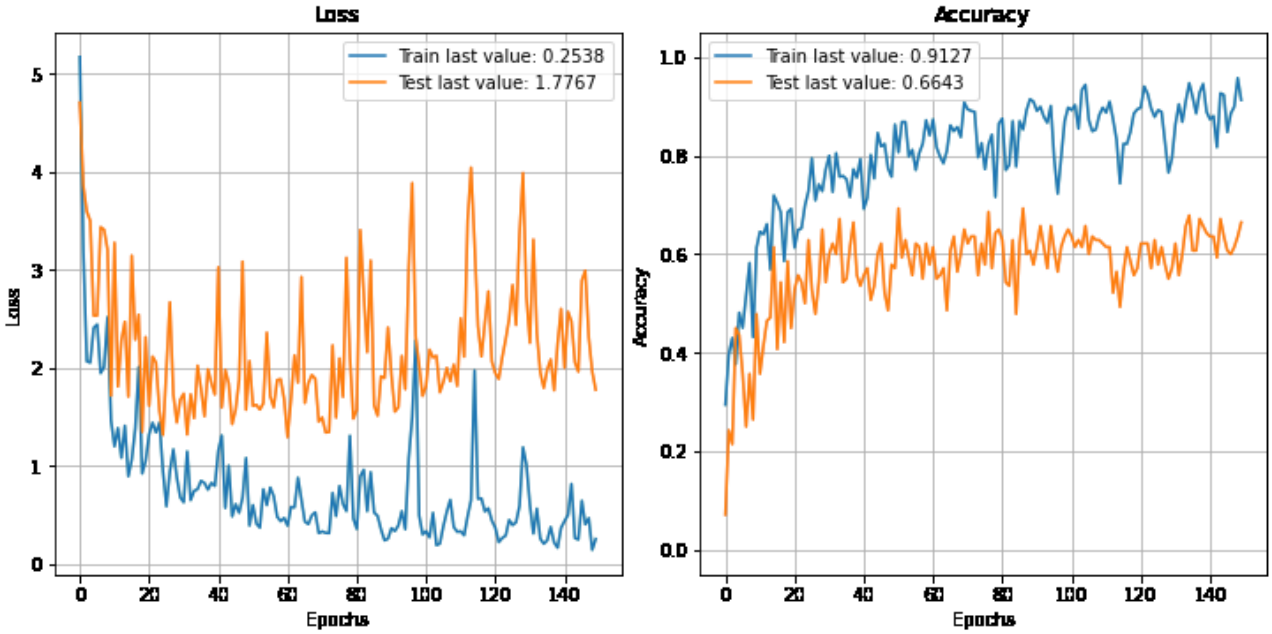
KUVA 4: Siirto-oppimiseen käytettyyn VGG16 rakenteeseen lisättyine kerroksineen

Alkuperäinen VGG16 rakenne, joka on esitetty kuvassa 5, koulutettiin siten, että ensin VGG16 rakenne ilman ylintä täysin kytkettyä kerrosta jäädytettiin painokertoimien osalta mahdollistamaan piirre-erottelu. Tällöin koulutettavien parametrien määrä oli 175 623 kappaletta. Kun aineistoa oli koulutettu riittävästi ylimmille, lisätyille kerroksille, avattiin VGG16 rakenteesta kaksi kerrosta lisää painokertoimien hienosäätämiseksi löydetyn virheminimin ympäristössä. Säätämällä optimoijan gradienttiaskel pieneksi varmistetaan oppimisen tapahtuvan kyseisessä ympäristössä, mikä näkyy kuvan 7 oikeassa laidassa pienempänä vaihteluna mallin laskennallisessa virheessä.



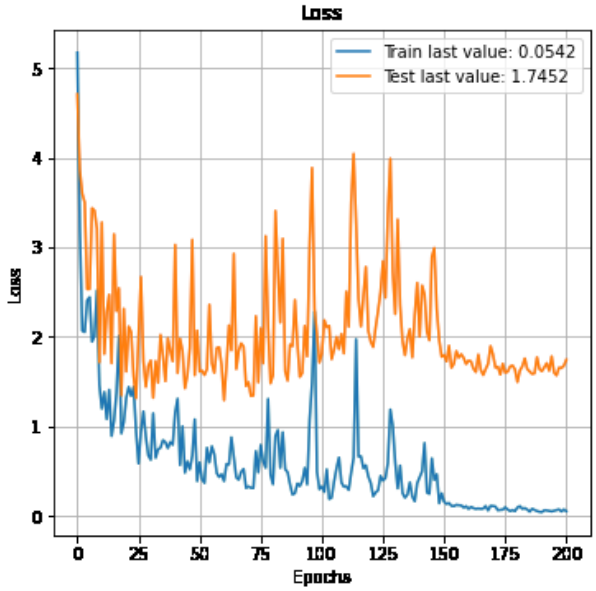
KUVA 5: VGG16 rakenne ilman ylimpiä kerroksia

Lopputuloksena rakenne oppi paremmaksi kuin edeltävin menetelmin. Koulutusjoukon tarkkuus saatiin vakuuttavaksi, mutta validointijoukon tarkkuus jäi valitettavan matalaksi.



KUVA 6: Siirto-oppimisella saavutetut koulutustulokset

Kuvassa 6 on esitetty 150 koulutusepookin historiadatan visualisointi. Kuvassa 7 on kuvattu hienosäädön aikainen virheen asteittainen väheneminen. Hienosäätö ei kuitenkaan merkittävästi vaikuttanut luokitustarkkuuteen, parantaen sitä kahden prosenttiyksikön verran verrattuna piirreoppimisen vaiheeseen. Evaluoinnin mukaisesti mallin tarkkuus validointijoukolla on 68.57%, virheen ollessa kuvan 7 mukaisesti 1.7452.

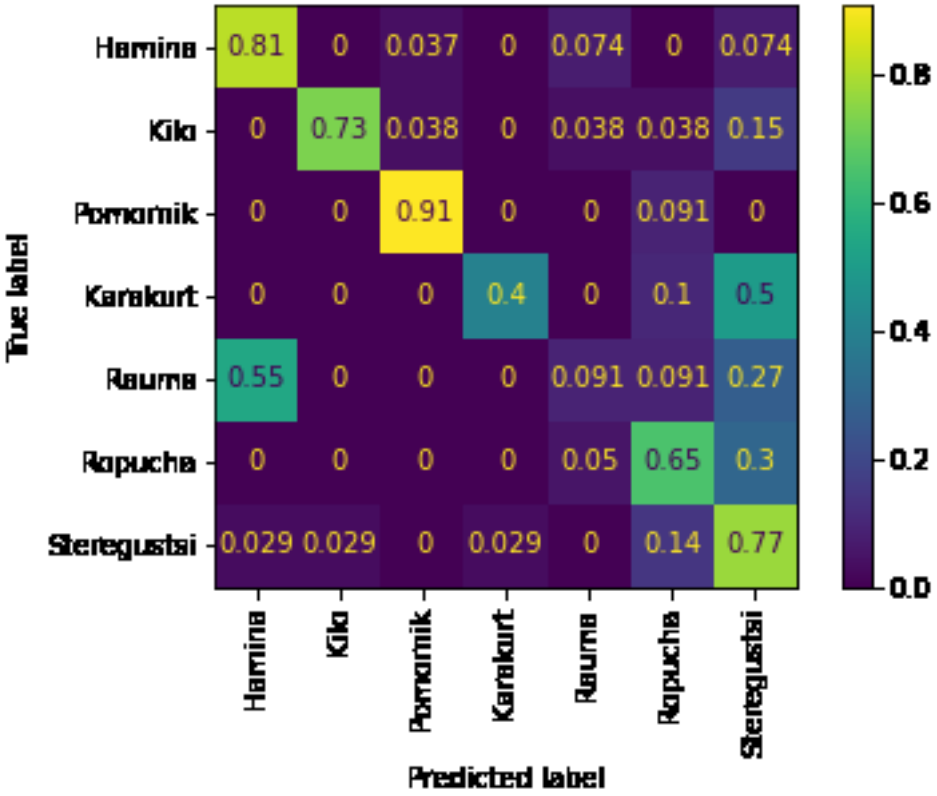


KUVA 7: Siirto-oppimismallin 150+50 epookin koulutuksen virhekuvaaja

Evaluointi

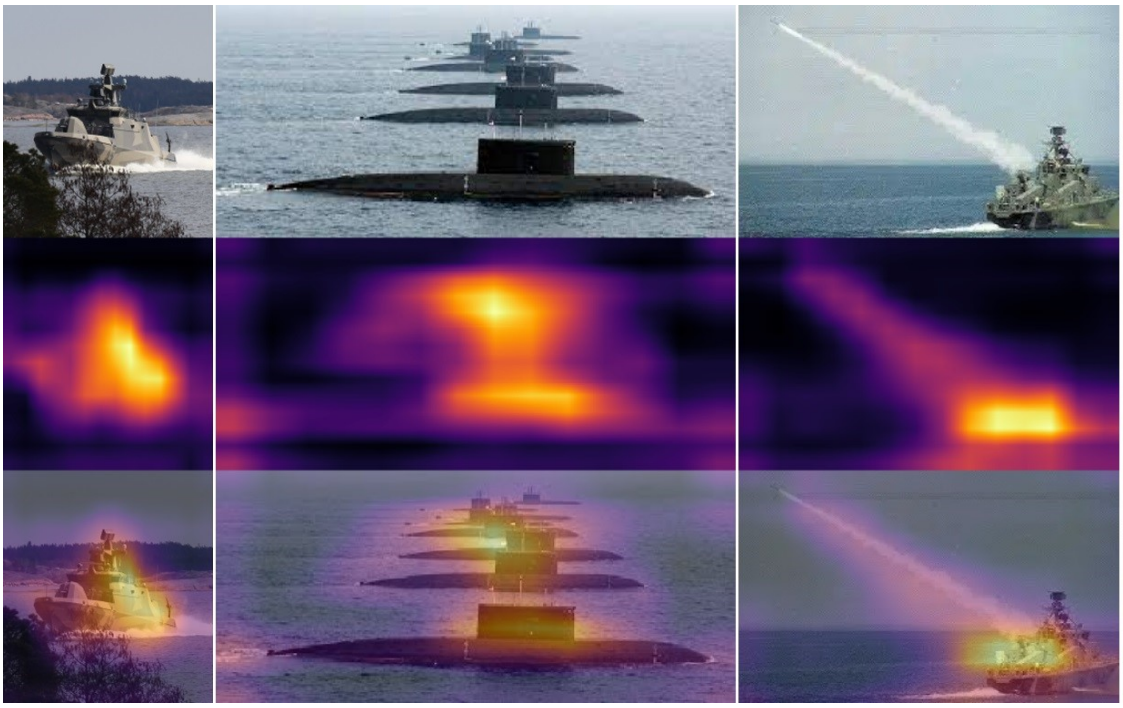
Mallintaessa tehdyn keskinäisen vertailun tuloksena evaluoitavaksi valittiin siirto-oppimisella tuotettu malli. Kyseisen mallin evaluointiin käytettiin validointidataa, sekä yksittäisiä kuvia jokaisesta luokasta. Edellä mainitusti malli saavutti validointijoukolla 68.57% tarkkuuden koulutustarkkuuden ollessa 98.6%. Satunnaisella testijoukolla, yksittäisillä kuvilla, tarkkuus oli vaatimaton 28.57%.

Kuvassa 8 on mallin virhematriisi validointijoukon osalta.



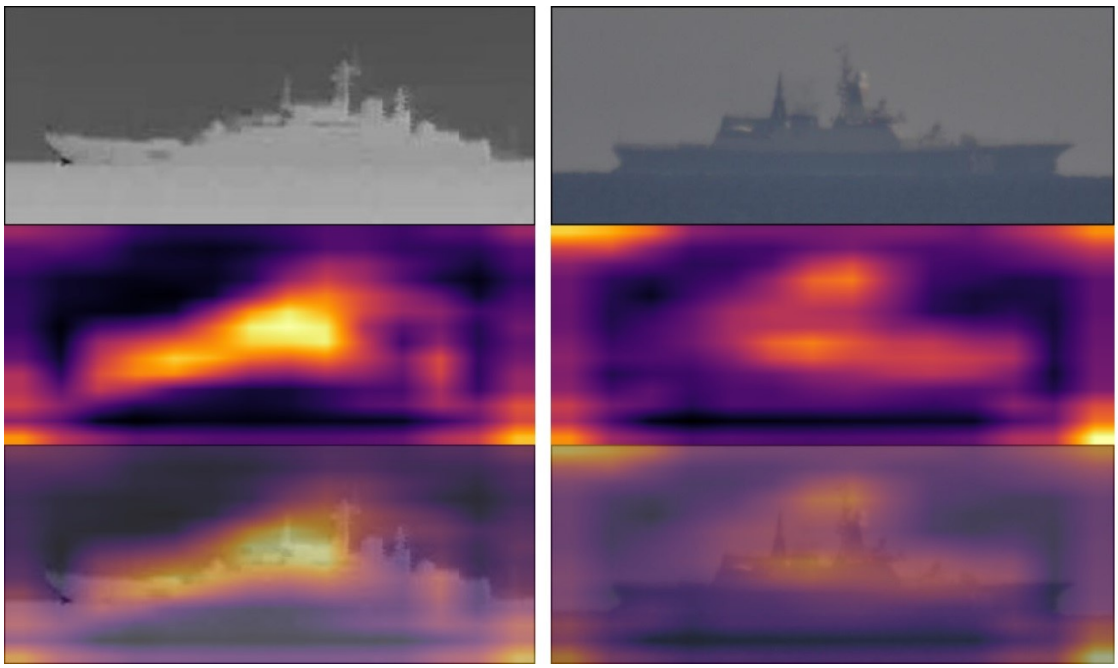
KUVA 8: Virhematriisi (”*confusion matrix*”) alusluokkien oikeiden ja arvioitujen luokitusten välillä

Kuten kuvan 8 matriisista nähdään, malli luokittelee yli 70% tarkkuudella oikein suurimman osan alusluokista. Alusluokat, joissa on vain vähän näytteitä, menevät suurella todennäköisyydellä sekaisin muiden alusluokkien kanssa. On oletettavaa, että algoritmin on ollut edullisinta minimoida virhefunktion arvo luokittelemalla esimerkiksi lähes kaikki Rauma-luokan alukset väärin, mutta saavuttaen parempi tarkkuus muissa luokissa, joissa näytteitä on enemmän. Huomionarvoisesti Rauma-luokan ohjusveneet menevät pääasiassa sekaisin Hamina-luokan ohjusveneiden kanssa, samoin kuin Karakurt-luokan alukset Stereguštši-luokan alusten kanssa. Tästä voidaan päätellä mallin olevan todennäköisesti oikeilla jäljillä opittujen piirreavaruuksien suhteen. Sen sijaan esimerkiksi Pomornik-luokan alukset on luokiteltu hyvällä tarkkuudella pienestä datamäärästä riippumatta. Tämä johtuu oletettavasti ilmatyynyaluksen piirteistä, jotka poikkeavat riittävästi muista datan alusluokista.

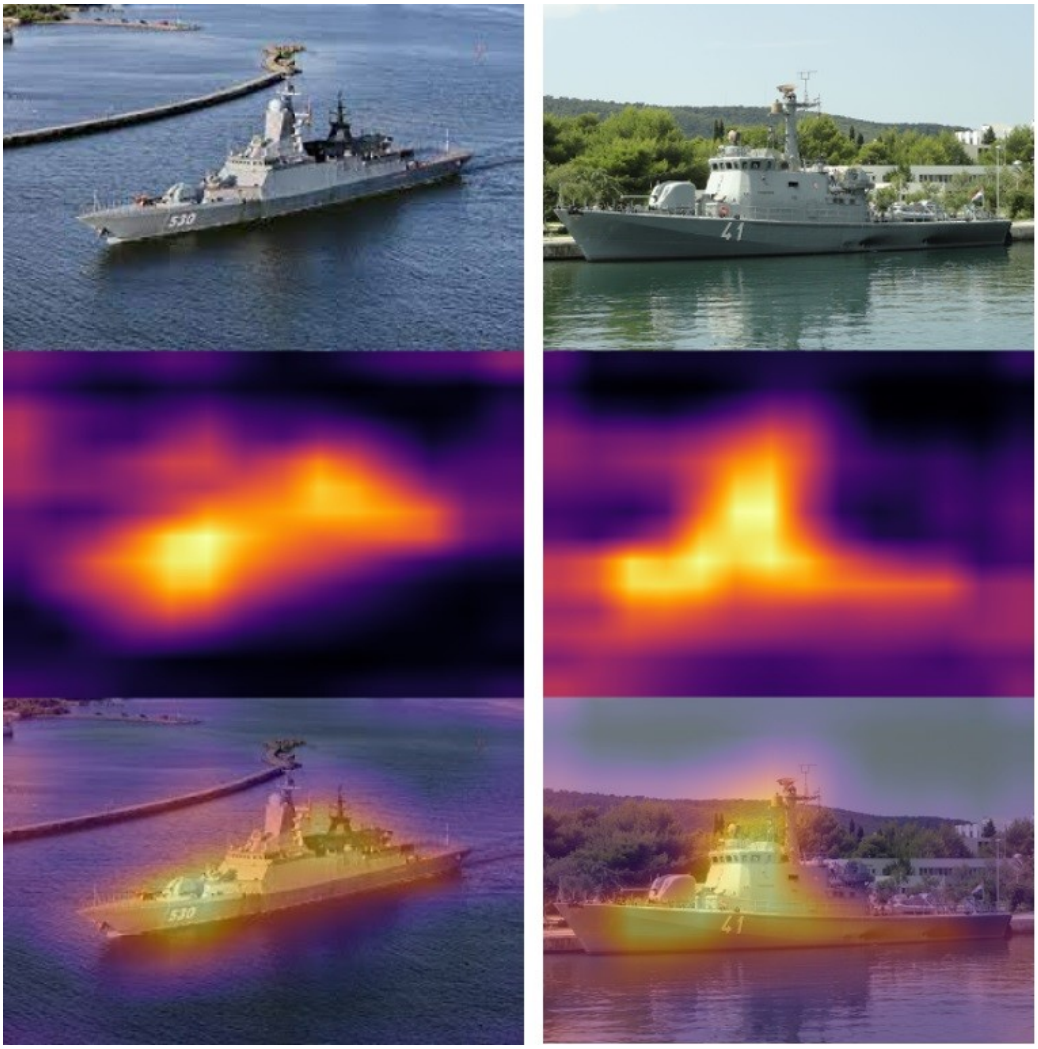


KUVA 9: Kolmeen alusluokkaan sovitetut VGG16 mallin aktivaation lämpökartat [3]

Kuvassa 9 on havainnollistettu siirto-oppimisella tuotetun mallin aktivaatiota yhden testikuvan osalta per alusyksikkö. Ylimpänä on alkuperäinen kuva, keskellä konvoluutioverkon aktivaatio esitettynä lämpökarttana siten, että kirkkaimmat kohdat ovat aiheuttaneet suurimman aktivaation. Menetelmää kutsutaan luokka-aktivaation kartoittamiseksi, jossa on otettu aktivaatiofunktioiden arvot mallin viimeisestä konvoluutioverkosta ennen luokittavia kerroksia, jotta saadaan visualisoitua mallin päätöksenteon perusteet. Kuvasta nähdään, että mitkä rakenteet aiheuttavat suurimman aktivaation, jonka perusteella voidaan tulkita luokitustarkkuutta ja -virhettä. Kuvassa 10 nähdään, miten häiriöllisissä kuvissa aktivaatiota aiheuttavat myös triviaalit ilmiöt.



KUVA 10: Ropucha-luokan ja Stereguštši-luokan alusten heikkolaatuisten kuvien aktivaatio



KUVA 11: Paremman Stereguštši-kuvan sekä Helsinki-luokan aktivaatioiden lämpökartat [4; 5]

Kuvassa 11 nähdään, miten malli tunnistaa selkeämmän Stereguštši-luokan kuvan oleelliset piirteet paremmin kuin kuvan 10 visualisoinnissa. Lisäksi testattiin, miten malli reagoi täysin vieraaseen alusluokkaan, tässä tapauksessa Helsinki-luokan alukseen FNS Ouluun, joka nykyään on Kroatian laivaston RTOP-41 Vukovar. Tässä yksittäistapauksessa malli tunnistaa oleellisia rakenteita, sekä luokittelee kenties sattumalta Helsinki-luokan aluksen seuraajaluokkaansa, Rauma-luokan ohjusveneeksi.

Johtopäätökset

Ratkaisuna väärin luokiteltujen kohteiden luokitustarkkuuden parantamiseen on yksiselitteisesti näytteiden määrän lisääminen. Käyttökelpoisen mallin luominen edellyttäisi kaikkien haluttujen alusluokkien sisällyttämisen data-aineistoon ja sellaisina määrinä, että riittävä luokitustarkkuus on mahdollista saavuttaa. Lisäksi kuvia pitäisi olla kaikista toimintaympäristön olosuhteista niin sää- kuin valaistusolosuhteiden osalta.

Tässä mallissa virheluokituksia voitaisiin myös käsitellä siten, että luodaan binäärisiä luokittajia kahden samankaltaisen alusluokan välille, jolloin esimerkiksi tunnisteella Stereguštši tehtäisiin uusi luokitus binäärisellä mallilla Stereguštši – ja

Karakurt – luokan välillä. Yksinkertaisin tapa on kuitenkin luoda riittävän laaja aineisto, jolla pystytään luomaan yksi malli luokittelemaan kaikki kiinnostavat alusluokat. Binääristä luokittajaa voitaisiin sen sijaan käyttää luokittelemaan sota-alukset kauppa-aluksista, jonka jälkeen tunnistus toteutettaisiin moniluokkaisella luokittajalla kyseisestä aineistosta.

Lisäksi malli edellyttäisi esimerkiksi ”*selective search”* tai R-CNN algoritmin käyttöä, jotta kiinnostavat kohteet voidaan poimia sensorinkuvasta erillistä luokittelua varten [6; 7]. Ohjelmistoautomaatiolla voidaan laittaa automaattinen kamerajärjestelmä tarkastamaan esimerkiksi kaikki tutka- ja AIS-maalit näkyvyysalueella. Tekoälymalli luokittelisi havaitut maalit automaattisesti ja järjestelmä ilmoittaisi luokituksista tilannekuvaa tekevälle operaattorille. Lisäksi malli olisi sovellettavissa niin pinta- kuin ilma-aluksissa ja merivalvontakeskuksissa, helpottaen tunnistetun meritilannekuvan muodostamista organisaation eri tasoilla.

Tärkeimpinä askeleina toimivan mallin luomiseksi on koostaa valvontakuvista riittävä ja helposti hyödynnettävä data-aineisto kansiorakenteineen tai tietokanta, jolla malli voidaan kouluttaa, ja jonka pohjalta mallia voidaan päivittää. Käytön mahdollistamiseksi malli tulisi integroida operaattorien käyttämiin järjestelmiin, jotta automaatio vähentäisi ihmisten tekemän työn kuormittavuutta. Irrallisena järjestelmänä mallin käytettävyys ei merkittävästi vähennä operaattorien työtaakkaa.

Lisäksi yhtenä tulevaisuuden mahdollisuutena kyseisen kaltaisen mallin hyödyntämisessä on esimerkiksi maalinosoituslennokkien käyttö. Lennokin hyödyntämänä malli voi tuottaa alusluokituksen paikan päällä ja lähettää kuvan sijaan takaisin maalitiedon, vähentäen tiedonsiirtokapasiteetin tarvetta ja mahdollistaen siten maalinosoituksen onnistumisen tehokkaasti pitkilläkin etäisyyksillä.

LÄHTEET

[1] Russel S., Norvig P., *Artificial Intelligence – A Modern Approach, Third Edition*, Pearson Education Limited, Harlow, Englanti, 2016, ISBN: 978-1292-1453-96-4

[2] Simonyan K., Zisserman A., *Very Deep Convolutional Networks for Large-Scale Image Recognition*, 4.8.2014, Computer Vision and Pattern Recognition, arXiv: 1409.1556, viitattu 27.1.2022, saatavilla: https://arxiv.org/abs/1409.1556

[3] Kilo-luokan alusten kuva osoitteesta: https://militarywatchmagazine.com/article/russia-s-kilo-class-black-hole-submarines

[4] Stereguštši-luokan kuva osoitteesta https://en.wikipedia.org/wiki/Steregushchiy-class_corvette#/media/File:10-YearAnniversary2018-02.jpg

[5] Helsinki-luokan kuva osoitteesta https://en.wikipedia.org/wiki/Helsinki-class_missile_boat#/media/File:RTOP-42_Dubrovnik_lora.jpg

[6] Uijlings J.R.R., Gevers T., Smeulders A.W.M., *Selective Search for Object Recognition*, International Journal of Computer Vision, 104, elokuu 2013, s. 154-171, 10.1007/s11263-013-0620-5

[7] Ren S., He K., Girschick R., Sun J., *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*, Computer Vision and Pattern Recognition (cs.CV), 4.6.2015, arXiv:1506.01497, saatavilla: https://doi.org/10.48550/arXiv.1506.01497

Kuvatunnistuksen lähdekoodi

Syväoppimisen hyödyntämiseksi ladataan ensin tarvittavat riippuvuudet ja kirjastomenetelmät.

```
#!/usr/bin/env python


import os
import sklearn
import pandas as pd
import tensorflow as tf
import matplotlib.pyplot as plt
import numpy as np
from sklearn.metrics import classification_report
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow import keras


from numpy.random import seed
seed(42)
tf.random.set_seed(42)

batch_size = 16
img_height = 200
img_width = 200

data_dir = cwd()
```

Data-aineisto on manuaalisesti kansioitu siten, että alakansiot ovat annotaatioita alusluokille. Näin ollen voidaan käyttää valmiita metodeja data-aineiston hyödyntämiseen.

```
train_ds = tf.keras.preprocessing.image_dataset_from_directory(
    data_dir,
    validation_split = 0.2,
    subset="training",
    seed = 42,
    image_size=(img_height, img_width),
    batch_size = batch_size
)

val_ds = tf.keras.preprocessing.image_dataset_from_directory(
    data_dir,
    validation_split = 0.2,
    subset="validation",
    seed = 42,
    image_size=(img_height, img_width),
    batch_size = batch_size
)

class_names = train_ds.class_names
print(class_names)
print('Number of classes: ')
print(len(class_names))

num_classes = len(class_names)
```

Rakennetaan ja koulutetaan CNN-malli.

```
from tensorflow.keras import layers

model1 = tf.keras.Sequential()

model1.add(tf.keras.layers.preprocessing.Rescaling(1./255))
model1.add(tf.keras.layers.Conv2D(filters=16, kernel_size=3,
                                  padding='same', activation='relu'))
model1.add(tf.keras.layers.MaxPooling2D(pool_size=2))
model1.add(tf.keras.layers.Conv2D(filters=32,kernel_size=3,
                                  padding='same', activation='relu'))
model1.add(tf.keras.layers.Conv2D(filters=32,kernel_size=3,
                                  padding='same', activation='relu'))
model1.add(tf.keras.layers.MaxPooling2D(pool_size=2))
model1.add(tf.keras.layers.Flatten())
model1.add(tf.keras.layers.Dense(num_classes, activation="softmax"))

model1.compile(
    optimizer = 'adam',
    loss = tf.losses.SparseCategoricalCrossentropy(),
    metrics=['accuracy']
)

history = model1.fit(
    train_ds,
    validation_data=val_ds,
    epochs=20
)
```

Koulutushistoria on visualisoitu seuraavalla metodilla, kuva itse raportissa.

```
model1.summary()

if int(tf.__version__.split('.')[0]) > 1:
    acc_key = 'accuracy'
else:
    acc_key = 'acc'

acc      = history.history[acc_key]
val_acc  = history.history['val_'+acc_key]
loss     = history.history['loss']
val_loss = history.history['val_loss']
epochs   = range(1,len(acc)+1)

plt.plot(epochs, acc,  label='Training accuracy')
plt.plot(epochs, val_acc,  label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xticks(epochs)
plt.xlabel('epoch')
plt.ylabel('accuracy')
plt.legend();
```

Seuraavaksi generaattoreihin lisättiin augmentaatio, jolla luodaan satunnaista vaihtelua koulutusdatan kuviin.

```
train_datagen = ImageDataGenerator(rescale = 1./255,
                                   rotation_range=40,
                                   width_shift_range = 0.2,
                                   height_shift_range = 0.2,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True,
                                   fill_mode = 'nearest',
                                   validation_split=0.2)

val_datagen = ImageDataGenerator(rescale=1./255,
                                 validation_split=0.2)

train_generator =
train_datagen.flow_from_directory(batch_size=batch_size,
                                  directory=data_dir,
                                  target_size=(200,200),
                                  color_mode='grayscale',
                                  class_mode='categorical',
                                  subset='training')

val_generator =
val_datagen.flow_from_directory(batch_size=batch_size,
                                  directory=data_dir,
                                  target_size=(200,200),
                                  color_mode='grayscale',
                                  class_mode='categorical',
                                  subset='validation')
```

Toteutettu augmentaatio muun muassa kiertää ja rajaa kuvia, tuottaen eri tavoin muokkautuneita versioita alkuperäisestä aineistosta.

```
from tensorflow.keras import optimizers
model2.compile(loss='categorical_crossentropy',
               optimizer=keras.optimizers.Adam(),
               metrics=['accuracy'])

history = model2.fit(train_generator,
                     epochs=100,
                     validation_data = val_generator)
```

Mallin koulutusdata ja tarkkuus on raportoitu liitteessä 5 vastaavalla metodilla kuin aiempana.

Seuraavaksi demonstroidaan tukivektorikoneiden suoriutuminen kyseisestä luokittelutehtävästä. Ohjelma on kirjoitettu viitteessä olevan mallin pohjalta [1]. Ensin data ladataan SVM:lle sopivaan taulukkomuotoon, haetaan SVM:n kirjastometodit ja asetetaan parametrit, joiden permutaatioilla vertaillaan koulutustuloksia. Data jaetaan suhteessa 80:20 ja malli koulutetaan.

```
from skimage.transform import resize
from skimage.io import imread

categories = class_names
flat_data_arr = []
target_arr = []

for i in categories:
    path = os.path.join(data_dir,i)

    for img in os.listdir(path):
      img_array = imread(os.path.join(path,img))
      img_resized = resize(img_array,(150,150,3))
      flat_data_arr.append(img_resized.flatten())
      target_arr.append(categories.index(i))

flat_data = np.array(flat_data_arr)
target = np.array(target_arr)
df = pd.DataFrame(flat_data)
df['Target'] = target
x = df.iloc[:,:-1]
y = df.iloc[:,-1]

from sklearn import svm
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score as accuracy_score

param_grid={'C':[1, 10, 100],'gamma':[0.01, 0.001,
0.0001],'kernel':['rbf']}
model3 = svm.SVC()

from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(x, y, test_size=
0.20, random_state=77, stratify=y)

model3.fit(x_train,y_train)
```

```
y_pred=model3.predict(x_test)

print(f"Mallin tarkkuus on {accuracy_score(y_pred,y_test)*100}%")
```

Mallin parantamiseksi testattiin eri parametrien permutaatiot.

```
svc=svm.SVC(probability=True)
grid=GridSearchCV(svc, param_grid, refit=True, verbose=3)

grid.fit(x_train,y_train)

y_pred=grid.predict(x_test)

print(f"Mallin tarkkuus on {accuracy_score(y_pred,y_test)*100}%")

print(grid.best_params_)

print(classification_report(y_test, y_pred))
```

Tulokset on raportoitu liitteessä 6.

Seuraavaksi testattiin siirto-oppimista lataamalla VGG16 rakenne ja sen imageNet-aineistosta oppimat painokertoimet.

```
from tensorflow.keras.applications.vgg16 import preprocess_input
from tensorflow.keras.applications import VGG16

batch_size = 16
img_height = 224
img_width = 224

tf.keras.backend.clear_session()

conv_base = VGG16(weights='imagenet',
                  include_top=False,
                  input_shape=(img_height, img_width, 3))

conv_base.summary()

test_datagen =
ImageDataGenerator(preprocessing_function=preprocess_input)

test_generator = test_datagen.flow_from_directory(batch_size=1,
                                directory=test_data_dir,
                                target_size=(img_height,img_width),
                                class_mode='categorical',
                                shuffle = False,
                                seed=42)
```

VGG16 rakenne jäädytettiin ja sen ylimmiksi kerroksiksi lisättiin kolme uutta kerrosta, jotka ovat koulutettavissa sillä datalla, jonka jäädytetyt kerrokset tuottavat aiemmin VGG16 rakenteessa.

Tämän lisäksi muokattiin datageneraattoreita.

```
conv_base.trainable = False
model_vgg16 = tf.keras.models.Sequential()

model_vgg16.add(conv_base)
model_vgg16.add(tf.keras.layers.Flatten())
model_vgg16.add(tf.keras.layers.Dense(num_classes,
                        activation="softmax"))

model_vgg16.summary()

model_vgg16.compile(optimizer=keras.optimizers.RMSprop(),
              loss=keras.losses.CategoricalCrossentropy(),
              metrics=['categorical_accuracy'])

print('Training set:')
train_generator =
train_datagen.flow_from_directory(batch_size=batch_size,
                                  directory=data_dir,
                                  target_size=(img_height,img_width),
                                  class_mode='categorical',
                                  subset='training',
                                  shuffle = False,
                                  seed=42)

print('Validation set:')
val_generator =
val_datagen.flow_from_directory(batch_size=batch_size,
                                  directory=data_dir,
                                  target_size=(img_height,img_width),
                                  class_mode='categorical',
                                  subset='validation',
                                  shuffle = False,
                                  seed=42)
```

Seuraavaksi mallia koulutettiin yhteensä 150 epookin verran, taltioimalla koulutuksen vaihe jokaisen epookin aikana 5 osan välein ja jatkamalla koulutusta tallennetusta pisteestä.

```
cp_callback =
tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_path, verbose=
1, save_weights_only=True, save_freq = 5*batch_size)

history = model_vgg16.fit(train_generator,
                      epochs=150,
                      validation_data=val_generator,
                      callbacks=[cp_callback])
```

Kun malli oli saavuttanut nähtävästi maksimitarkkuuden ympäristön virheen minimin ympäristössä, avattiin muutamia jäädytettyjä kerroksia hienosäätöä varten ja hienosäädettiin koko avatun rakenteen painokertoimia 50 epookin verran.

```
conv_base.trainable = True

for layer in conv_base.layers[:-2]:
  layer.trainable = False

for layer in conv_base.layers:
  print(layer, " ", layer.trainable)

conv_base.summary()

model_vgg16.compile(loss=keras.losses.CategoricalCrossentropy(),
              optimizer=keras.optimizers.RMSprop(1e-5),
              metrics=['categorical_accuracy'])

total = 150+50
history_tune = model_vgg16.fit(train_generator,
                         epochs=total,
                         validation_data = val_generator,
                         initial_epoch = history.epoch[-1],
                         callbacks=[cp_callback])

from plot_keras_history import chain_histories

history = chain_histories(history, history_tune)

plot_history(history)

model.evaluate(test_generator)

model.evaluate(val_generator)
```

Seuraavaksi tarkasteltiin mallin tarkkuutta niin ohuella testiaineistolla kuin muodostamalla luokittainen tarkkuusmatriisi validointidatan luokituksista. Testiaineistolla, jossa jokaisessa luokassa on tasan yksi kuva, tarkkuus on 28.57%. Validaatiojoukossa evaluaatiotarkkuus on 68.57%.

```
Y_pred = model_vgg16.predict(val_generator)

predictions = tf.argmax(Y_pred,1)

class_names = ['Hamina', 'Kilo', 'Pomornik', 'Project 22800',
'Rauma', 'Ropucha', 'Steregustsi']

class_names_dict = {0: 'Hamina', 1: 'Kilo', 2: 'Pomornik', 3:
'Project 22800', 4: 'Rauma', 5: ' Ropucha', 6: 'Steregustsi'}

y_true = val_generator.classes

from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay

cm = confusion_matrix(y_true, predictions,
labels=test_generator.classes)

print(confusion_matrix(y_true, predictions))

disp = ConfusionMatrixDisplay.from_predictions(y_true, predictions,
                              normalize="true",
                              display_labels=class_names,
                              xticks_rotation="vertical")
```

Konvoluutioverkon aktivaation visualisointi lämpökarttoina on toteutettu seuraavalla ohjelmalla. Alkuperäisen ohjelman on kirjoittanut Adrian Rosebrock. [2] Siirto-oppimista tukeva muutos sisäisenä mallina käytettyjen kerrosten hyödyntämisestä on toteutettu viitteen [3] esittämällä tavalla.

```
class GradCAM:
  import os
  import sklearn
  import pandas as pd
  import tensorflow as tf
  import matplotlib.pyplot as plt
  import matplotlib.cm as cm
  import numpy as np
  from sklearn.metrics import classification_report
  from tensorflow.keras.preprocessing.image import ImageDataGenerator
  from tensorflow import keras
  from tensorflow.keras import layers
  from tensorflow.keras.models import Model
  import cv2

  def __init__(self, model, classIdx, inner_model=None,
                  layerName=None):

      self.model = model
      self.classIdx = classIdx
      self.inner_model = inner_model

      if self.inner_model == None:
            self.inner_model = model
            self.layerName = layerName

  def find_target_layer(self):
      for layer in reversed(self.model.layers):
            if len(layer.output_shape) == 4:
                  return layer.name
      raise ValueError("Could not find 4D Layer")
```

Kuvien valinta ja muokkaus visualisointia varten alla.

```
cwd = os.getcwd()

images = [cwd+'/testi/Hamina/hamina43.jpg',
      cwd+'/testi/Rauma/rauma1.jpg',
      cwd+'/testi/Kilo/kilo1.jpg',
      cwd+'/testi/Pomornik/pomo1.jpg',
      cwd+'/testi/Project 22800/karakurt1.jpg',
      cwd+'/testi/Ropucha/ropu1.jpg',
      cwd+'/testi/Steregustsi/stere1.jpg',
      cwd+'/testi/Steregustsi/stere2.jpg',
      cwd+'/testi/Rauma/vukovar.jpg']

img_path = images[img_ind] #valitaan visualisoitava kuva

display(Image(img_path)) #esitetään alkuperäinen kuva

orig = cv2.imread(images[img_ind])

resized = cv2.resize(orig, (224, 224))#muokataan oikean kokoiseksi

image = resized[np.newaxis is None,:,:,:]
```

```
def compute_heatmap(self, image, eps=1e-8):
    gradModel =
      tf.keras.models.Model(inputs=[self.inner_model.inputs],
      outputs=[self.inner_model.get_layer(self.layerName).output,
                  self.inner_model.output])

    with tf.GradientTape() as tape:
      inputs = tf.cast(image, tf.float32)
      (convOutputs, predictions) = gradModel(inputs)
      loss = predictions[:, self.classIdx]

    grads = tape.gradient(loss, convOutputs)

    castConvOutputs = tf.cast(convOutputs > 0, "float32")
    castGrads = tf.cast(grads > 0, "float32")
    guidedGrads = castConvOutputs * castGrads * grads

    convOutputs = convOutputs[0]
    guidedGrads = guidedGrads[0]

    weights = tf.reduce_mean(guidedGrads, axis=(0, 1))
    cam = tf.reduce_sum(tf.multiply(weights, convOutputs), axis=-1)

    (w, h) = (image.shape[2], image.shape[1])
    heatmap = cv2.resize(cam.numpy(), (w, h))

    numer = heatmap - np.min(heatmap)
    denom = (heatmap.max() - heatmap.min()) + eps
    heatmap = numer / denom
    heatmap = (heatmap * 255).astype("uint8")

    return heatmap
```

```
def overlay_heatmap(self, heatmap, image, alpha=0.5,
    colormap=cv2.COLORMAP_INFERNO):

    heatmap = cv2.applyColorMap(heatmap, colormap)
    output = cv2.addWeighted(image, alpha, heatmap, 1 - alpha, 0)

    return (heatmap, output)
```

Aktivaatiokuvien luominen yllä kuvattuja metodeja kutsumalla.

```
cam = GradCAM(model, None, inner_model=model.get_layer("vgg16"),
                  layerName="block5_conv3")
heatmap = cam.compute_heatmap(image)

heatmap = cv2.resize(heatmap, (orig.shape[1], orig.shape[0]))
(heatmap, output) = cam.overlay_heatmap(heatmap, orig, alpha=0.5)

plt.imshow(cv2.cvtColor(output, cv2.COLOR_BGR2RGB))
plt.show()

cv2.imwrite('kuvan_nimi.png',output)
```

LÄHTEET

[1] Shanmukh V., *Image Classification Using Machine Learning – Support Vector Machine (SVM)*, 3.3.2021, viitattu 6.3.2022, saatavilla: https://medium.com/analytics-vidhya/image-classification-using-machine-learning-support-vector-machine-svm-dc7a0ec92e01

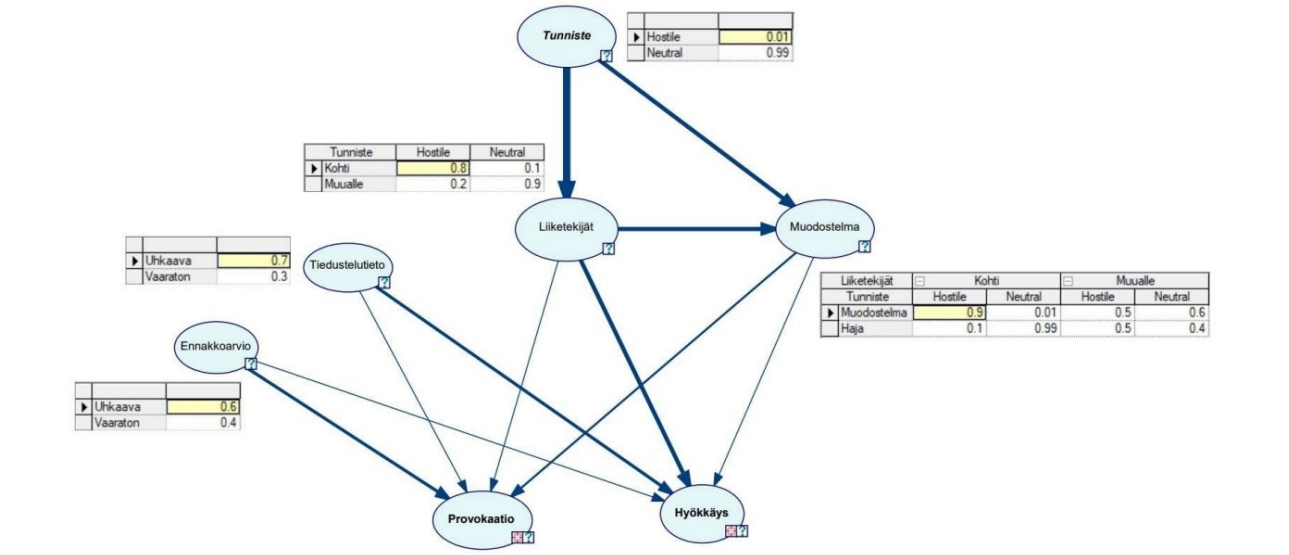
[2] Rosenbrock Adrian, ”*GradCAM: Visualize class activation maps with Keras Tensorflow and deep learning”*, 9.3.2020, viitattu 31.1.2022, saatavilla: https://www.pyimagesearch.com/2020/03/09/grad-cam-visualize-class-activation-maps-with-keras-tensorflow-and-deep-learning/

[3] Saatavilla: https://stackoverflow.com/questions/60623869/gradcam-with-guided-backprop-for-transfer-learning-in-tensorflow-2-0

Bayes-verkot epävarman tiedon käsittelyssä

Tässä liitteessä havainnollistetaan tilannekuvan maalitiedon synteesissä mahdollista priorien vaikutusta Bayes-verkon tuottamaan posterioritodennäköisyyteen. Havainnollistusta varten käytetään GeNIe Academic-ohjelmistoa, joka on BayesFusion LLC:n tuote ja saatavilla akateemista tutkimusta varten osoitteesta http://www.bayesfusion.com/.

Bayes-verkkojen käytettävyyden havainnollistamiseksi on luotu esimerkkimalli, jossa käsitellään tilannekuvaelementin havaintoa. Havainnolle on tuotettu aiemmin toimenpitein tunniste, mallissa joko neutraali tai vihamielinen. Tunniste perustuu havainnoinnin tekemiin toimenpiteisiin, jonka jälkeen siihen liitetään muuta saatavilla olevaa dataa, kuten liiketekijät ja mahdollinen osuus muodostelmaan. Liiketekijät ovat havaitsijan näkökulmasta joko kohti tärkeää kohdetta tai muualle. Vastaavasti havaittu kohde joko on osa muodostelmaa tai on hajallaan muista mahdollisista kohteista. Lisäksi maalin tarkoitteen tulkitsemiseen liittyvät tiedustelutieto uhkatilanteesta sekä ennakkoarvio vallitsevasta yleistilanteesta. Näistä tiedustelutiedolla on merkittävämpi luotettavuuskerroin, sillä laajempi arvio on ei ole mahdollisesti päivittynyt tai yhtä eksakti. Näillä tiedoilla saadaan luotua yksinkertainen Bayes-verkko, joka on kuvattu alla.



Provokaation totuustaulu

Tiedustelutieto	Uhkaava								Vaaraton							
Muodostelma	Muodostelma				Haja				Muodostelma				Haja			
Ennakko-arvio	Uhkaava		Vaaraton		Uhkaava		Vaaraton		Uhkaava		Vaaraton		Uhkaava		Vaaraton	
Liiketekijät	Kohti	Muualle	Kohti	Muualle	Kohti	Muualle	Kohti	Muualle	Kohti	Muualle	Kohti	Muualle	Kohti	Muualle	Kohti	Muualle
False	0.3	0.6	0.65	0.7	0.2	0.3	0.8	0.9	0.1	0.4	0.7	0.8	0.6	0.7	0.9	0.95
True	0.7	0.4	0.35	0.3	0.8	0.7	0.2	0.1	0.9	0.6	0.3	0.2	0.4	0.3	0.1	0.05

Hyökkäyksen totuustaulu

Liiketekijät	Kohti								Muualle							
Tiedustelutieto	Uhkaava				Vaaraton				Uhkaava				Vaaraton			
Muodostelma	Muodostelma		Haja		Muodostelma		Haja		Muodostelma		Haja		Muodostelma		Haja	
Ennakko-arvio	Uhkaava	Vaaraton	Uhkaava	Vaaraton	Uhkaava	Vaaraton	Uhkaava	Vaaraton	Uhkaava	Vaaraton	Uhkaava	Vaaraton	Uhkaava	Vaaraton	Uhkaava	Vaaraton
False	0.05	0.1	0.2	0.2	0.6	0.6	0.7	0.7	0.7	0.7	0.8	0.8	0.9	0.9	0.95	0.95
True	0.95	0.9	0.8	0.8	0.4	0.4	0.3	0.3	0.3	0.3	0.2	0.2	0.1	0.1	0.05	0.05

KUVA 1: Maalidatan synteesin Bayes-verkko

Kuvan 1 Bayes-verkossa oletetaan havaitun maalin pyrkivän joko provokaatioon, hyökkäykseen tai ei kumpaankaan näistä. Käsitteiden erona on se, että provokaatio voi olla alueloukkaus tai muu kevyempi toimenpide, kun taas hyökkäys edellyttää jonkinasteisia aktiivisia toimenpiteitä omia yksiköitä tai oman valtion aluetta kohtaan.

Bayes-verkkoon on asetettu esimerkinomaiset todennäköisyydet kullekin tapahtumalle. Lapsisolmujen posteriorit muodostuvat näiden vanhempien eli priorien todennäköisyyksien tuloista tutkimuksessa kuvatun kaavan 5 mukaisesti. Esimerkiksi solmun tunniste todennäköisyys vihamieliselle maalille on 1%, kun taas neutraalille maalille 99%. Kaikkien solmujen reunatodennäköisyydet ja ehdolliset todennäköisyydet ovat koostettuina kuvassa näkyviin taulukoihin.

Näin ollen, mikäli tunnistetietoa ei ole saatavilla, saadaan laskettua posterioritodennäköisyys hyökkäykselle tunnistamattoman maalin tapauksessa molemmilla ehdoilla. Tällöin hyökkäyksen posteriori on 25.4 %, kun mitään prioritietoja ei ole saatavilla. Vastaavasti rauhanajan posteriori, jossa tiedustelutiedot ja ennakkoarvio pitävät uhkaa epätodennäköisenä, hyökkäyksen posteriori olisi 10.4% ja provokaation 13.7%. On huomioitava, ettei tässä esimerkissä käytetä edes viitteellisesti todellisia arvoja.

Tarkentamalla prioritietoja esimerkiksi siten, että tunniste on vihamielinen, kohteen liiketekijät ovat kohti ja tiedustelutieto ennakoi hyökkäystä, kasvaa posterioritodennäköisyys $p(Hyökkäys = True \,|Liiketekijät = kohti, Tiedustelutieto = uhkaava, Tunniste = Vihamielinen)$ arvoon 80.4%. Vastaavasti, vaikka tunniste olisi vihamielinen, mutta tiedustelutieto ei tue uhkaavuutta, saadaan todennäköisyydelle

$$p(Hyökkäys = True \,|Liiketekijät = kohti, Tiedustelutieto = vaaraton, Tunniste = Vihamielinen, Muodostelma = Tosi) = 40\%,$$

kun taas

$$p(Provokaatio = True \,|Liiketekijät = kohti, Tiedustelutieto = vaaraton, Tunniste = Vihamielinen, Muodostelma = Tosi) = 66\%.$$

Annetuilla esimerkeillä kyseinen Bayes-verkko vaikuttaa tuottavan intuitiivisesti oikeansuuntaisia tuloksia esimerkiksi yllä olevassa tilanteessa, jossa tiedustelutiedon mukaan hyökkäys on epätodennäköinen mutta profiili on muutoin uhkaava, jolloin provokaation todennäköisyys on suurempi ja hyökkäyksen pienempi. Tässä tilanteessa nousee esiin kysymys riskin hallittavuudesta ja sen toteutumisen kustannuksesta. Mikäli tulkinta on, että havaittu tapahtuma tulee olemaan provokaatio, mutta tapahtuukin hyökkäys, on toteuman riski kustannuksiltaan suuri, jos vastatoimenpide todennäköisimpään havaintoon on liian vaatimaton vastaamaan myös mahdolliseen vaihtoehtoiseen tilannekehitykseen. Riskien painottaminen probabilistisessa lähestymistavassa on haastavaa ja edellyttää päätöksentekijän tilannekohtaista harkintaa.

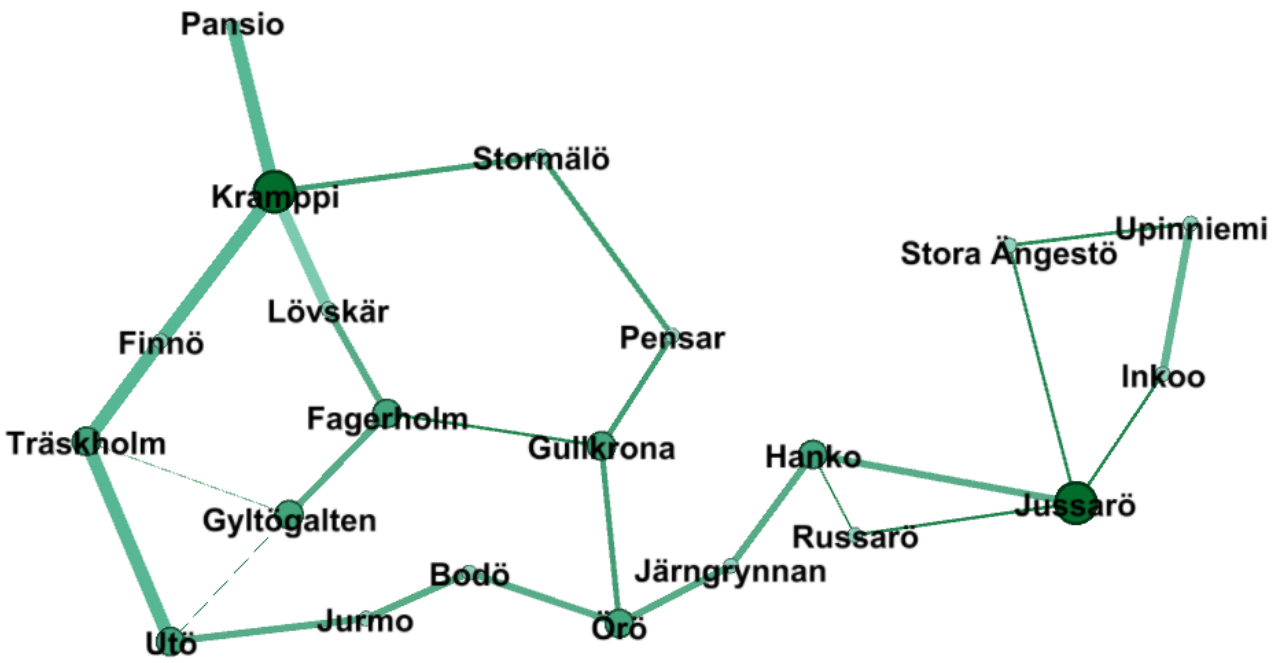
Bayes-verkolla kyetään yllä esitetyllä tavalla mallintamaan jonkin tapahtuman ehdollista todennäköisyyttä saatavilla olevan tiedon funktiona. Kuten tästä lyhyestä tarkastelusta käy ilmi, edellyttää toimivan Bayes-verkon toteuttaminen kattavaa asiantuntijatyötä, jotta käytetyt todennäköisyysarvot solmuille vastaavat mahdollisimman hyvin arviota todellisuudesta. Sotilasympäristön tapauksessa ongelmallisia ovat laadulliset arviot, joista ei löydy tilastollista dataa, sekä erillisten tapahtumien keskinäiset riippuvuudet ja näiden mallintaminen. Esimerkiksi alueloukkauksista löytyy tilastollista dataa, joka ei kuitenkaan ole riippumatonta muista tapahtumista. Tästä syystä alueloukkauksen todennäköisyyttä ei voi käsitellä pelkästään frekvenssipohjaisena, erillisenä ilmiönä, vaan siihen täytyy liittää muun muassa poliittinen tilanne.

Kuten työssä on mainittu, Bayes-verkko voidaan myös oppia datasta. Bayes-oppimiseen soveltuvaa tietoa työssä luonnosteltuihin käyttötarkoituksiin ei kuitenkaan löydetty tai sitä ei ole saatavilla. Havainnoinnin kategoriaan olisi sovellettavissa muun koneoppimisen tavoin probabilistisia metodeja, mutta synteesin ja päätöksenteon viitekehyksissä probabilistinen päättely ja sen verkkorakenne tulee lähtökohtaisesti mallintaa asiantuntijatyönä.

Älykkäät agentit, hakualgoritmit ja optimointi

Laivastotaisteluosaston kontekstissa voidaan päätöksentekijälle muodostaa toimintavaihtoehtoja symbolisen tekoälyn menetelmin. Symbolisen tekoälyn käsityksessä älykkäät agentit vuorovaikuttavat ympäristössään siirtyen eri tilojen välillä, päätyen älykkäästi tai älykkäältä vaikuttavasti jollain tavalla optimaaliseen ratkaisuun kyseisessä tila-avaruudessa. Taistelutila on termi, jolla kuvataan taistelevia joukkoja ja näiden toimintoja kyseisessä toimintaympäristössä. Todellinen taistelutila esitetään tilannekuvaprojektiona, joka voidaan nähdä älykkäiden agenttien visualisoimisena niiden digitalisoituun, laskettavaan ympäristöön. Taistelutilan projektiossa tapahtuvia siirtymiä eri tilojen välillä voidaan simuloida ja optimoida erilaisin tekoälyn menetelmin.

Yksinkertainen esimerkki laivastojoukkojen kontekstissa on siirtymisten optimointi joidenkin ehtojen suhteen. Kun käytettävä väylästö syvyystietoineen on datana saatavilla, voidaan väylästöstä muodostaa esimerkiksi verkko, josta voidaan laskea erilaisilla hakualgoritmeilla nopeimpia siirtymiä tilojen eli tässä tapauksessa sijaintien välillä esimerkiksi sallitun syvyyden tai väylän leveyden suhteen [1]. Lisäksi verkosta voidaan tunnistaa muun muassa liikenteelle kriittisiä solmukohtia.



KUVA 1. Osa Suomen saariston väylästöä kuvattu verkkona

Kuvassa 1 on osa Suomen eteläisen saariston väylästöä suuntaamattomana verkkona, nimeten tiettyjä väylien risteyskohtia tai reittipisteitä läheisten paikannimien mukaan. Solmujen väliset kaaret kuvaavat väylän karttaan merkittyä kulkusyvyyttä. Solmujen koko kuvaa solmun astetta, eli sitä, montako kaarta kyseiseen solmuun tulee ja siitä lähtee. Näin ollen solmun koko kuvaa reittipisteen merkitystä risteyskohtana. Verkko on visualisoitu Exceliin julkisesta karttamateriaalista kerätystä aineistosta Gephi-ohjelmiston avulla. Gephi on saatavilla osoitteessa: https://gephi.org/

Kuvatussa verkossa on visualisoitu ainoastaan väylien syvyyksiä ja liittymistä toisiinsa eri reittipisteiden kautta. Oleellinen tieto alusosaston siirtymien kannalta on näiden reittipisteiden välinen etäisyys. Reittiä kuvaavaan kaareen

voidaan sitoa niin monta ominaisuutta kuin katsotaan tarpeelliseksi, mutta suurin mitattu kulkusyvyys ja etäisyys ovat näistä oleellisimpia.

Tarkasteltaessa alustaisteluosastoa älykkäinä agentteina, voidaan olettaa lähtötilaksi esimerkiksi se, että kaikki agentit eli alukset ovat solmuissa Pansio ja Upinniemi. Oletetaan molemmissa olevan neljä ohjusvene-oliota eli agenttia, joihin sidotaan tieto sijainnista Pansio- tai Upinniemi-solmuissa. Alla on kuvattu pseudokoodilla ohjusvene-olio, jolle asetetaan muun muassa pituus, paino, syväys ja nopeus.

```
Ohjusvene (loa, wgt, d, spd, aaw, surv, sonar, asw, coord)  {
        pituus                          = loa,
        paino                           = wgt,
        syväys                          = d,
        nopeus                          = spd,
        IT-ohjusten lukumäärä           = aaw,
        valvontakyky                    = surv,
        vedenalainen valvontakyky       = sonar,
        torpedojen lukumäärä            = asw,
        sijainti                        = coord
        }
```

Yllä kuvattuun olioon liitetään myös sille mahdolliset toimenpiteet ja niiden rajoitteet. Toimenpiteitä olisi esimerkiksi siirtyminen toiseen tilaan eli uuteen sijaintiin ja sen rajoitteena valittavissa olevien väylien kulkusyvyys. Esimerkissä neljä agenttia sijaitsee samassa tukikohdassa kerrallaan. Tehtävä olisi hajauttaa alusyksiköt kuvan 1 verkossa oleviin, todellisuudessa kiintomerkkialueen reunalla oleviin sijainteihin {Utö, Jurmö, Bodö, Örö, Russarö, Jussarö} mahdollisimman nopeasti ja siten, että ilmavalvonta- ja torjuntakyky sekä sukellusveneen etsintäkyky on maksimoitu kattamaan mahdollisimman suuri kokonaisalue. Kuvataan, että puolet ohjusveneistä molemmissa tukikohdissa on Rauma-luokan ohjusveneitä ja vastaavasti toinen puoli Hamina-luokan ohjusveneitä.

Ongelma voidaan muotoilla esimerkiksi predikaattilogiikalla semanttiseksi totuuslauseeksi, joka voidaan sen jälkeen ratkaista ja saada tulos, joka joko täyttää ehdot tai toteaa vaatimuksen toteuttamiskelvottomaksi. Relaatioiden kuvaamisen hyödyllisyydestä ja monikäyttöisyydestä huolimatta predikaattilogiikka on tässä yhteydessä vaikea toteuttaa järjestelmäksi, joka ratkaisisi laivastotaisteluosaston dynaamisia ongelmia. Toisaalta loogisen lauseen muotoilu voisi olla osa esimerkiksi taisteluosaston komentajan tahtotilan formalisointia. Predikaattilogiikan syntaksi ja loogisten konnektiivien merkitystä on käsitelty työn teoriaosuudessa [2 s. 137-143; 3 s. 285-312].

Esimerkki predikaattilogiikan lauseesta alusten sijoittelulle voitaisiin muotoilla seuraavasti. Universumi, jossa logiikka toimii, olisi sijaintien ja alusyksiköiden avaruus kuvattuna paikanniminä ja kylkinumeroina, eli $U = \{Pansio, Bodö, 70, 80, \dots, Örö, Jussarö, Pansio, Upinniemi\}$, jolloin sijainnit $s \in U$ ja maalisolmut eli tavoitteet olisivat joukko $y \subset s$. Alusluokat ovat $x = \{[x_1, s_1], .. [x_n, s_n]\} \in U$ ja $z = \{[z_1, s_1], .. [z_n, s_n]\} \in U$, siten, että $\forall x. (Hamina(x)$ ja $\forall z. (Rauma(z)$, $x \notin z.$ $f([x, s]) \rightarrow s$ on funktio, joka palauttaa kaksiosaisen muuttujan jälkimmäisen arvon eli tässä tapauksessa sijainnin. $P(s, y)$ on kaksipaikkainen predikaattilause, joka palauttaa totuusarvon kahden muuttujan vertailusta. Näin voidaan muodostaa lause, jossa esimerkiksi jokaisessa maalisolmussa tulee olla vähintään yksi alus kummastakin muuttujien joukosta muodossa

$$\forall y \exists x \exists z((P(f(x), y) \wedge (P(f(z), y))$$

Lisäksi voidaan täsmentää, että yhdessä maalisolmussa y tulee olla korkeintaan yksi alus kummastakin luokasta, jolloin lause voidaan muotoilla muuttujien $c = x$ ja $d = z$ avulla siten, että

$$\forall y \left(\left(\exists x \left(\left(P(f(x), y) \right) \wedge (\forall c (P(f(c), y) \rightarrow c = x) \right) \right) \wedge \left(\exists z \left((P(f(z), y) \wedge \forall d (P(f(d), y) \rightarrow d = z) \right) \right) \right) \right)$$

joka tarkoittaa sanallisesti, että kaikille sijainneille y on olemassa sellainen ohjusvene x, että sen kanssa samassa sijainnissa olevia saman alusluokan yksiköitä on vain se itse, sekä samat ehdot ohjusveneille z.

Mikäli lause on ratkaistavissa, on mahdollista löytää ehdot tyydyttävä ratkaisu tässä tietopohjassa. Predikaattilogiikkaan vaadittava tietopohja ja sen formaalien logiikkalauseiden muotoilu käyttökelpoiseksi asiantuntijajärjestelmäksi tässä tutkimuksessa kuvattuihin ongelmatilanteisiin vaikuttaa haastavalta ja kankealta, mutta on huomioitava, että vastaavia optimointeja tekevä ohjelmisto toimii kuitenkin vastaavan kaltaisesti yksinkertaisemmalla tasolla, propositioloogisesti, testaten erilaisia totuusarvoja erilaisten toimenpiteiden ja niiden rajoitteiden välillä.

Predikaattilogiikan lauseiden sijaan ongelmaa voidaan lähestyä ohjelmistollisesti hakualgoritmeilla ja erilaisilla optimointialgoritmeilla kuten työssä mainituilla hill-climb algoritmilla tai gradienttimenetelmällä. Jotta ratkaisun tilaan päätyviä toimenpiteitä voidaan optimoida, on tämän liitteen esimerkissä ensimmäinen askel laskea, mitä reittejä alusosaston yksiköiden tulee siirtyä tavoitteisiin, eli lyhimmät polut lähtösolmuista maalisolmuihin. Näin yksinkertaisessa verkossa haku voidaan toteuttaa esimerkiksi leveyshakuna, mutta laajemmissa kokonaisuuksissa muun muassa suosittu A*-algoritmi laskee tehokkaasti parhaat reittivaihtoehdot [1].

Kun lyhimmät polut on määritetty lähtösolmuista kaikkiin maalisolmuihin, voidaan iteroida eri vaihtoehtoja osaston yksiköiden sijoittelulle. Tätä varten tarvitaan virhefunktio, jolla voidaan arvioida kutakin ratkaisua ja tuottaa iteraatioista suhteellinen optimi eli virheminimi. Koska tehtävä on maksimoida valvottu pinta-ala, voidaan se saavuttaa minimoimalla päällekkäisyydet valvontakantamissa. Tämä voidaan toteuttaa laskennallisesti esimerkiksi siten, että virhefunktio on

$$\sum_{n=1}^{N} \sum_{k=1}^{N} \left(\left| \sqrt{(c_n - c_k)^2} - surv \right| \right) + \left(\left| \sqrt{(c_n - c_k)^2} - sonar \right| \right),$$

jossa lasketaan alusten koordinaattipisteiden (c) välisen Euklidisen etäisyyden ja valvontakantamien $surv$ ja $sonar$ erotuksien itseisarvojen summa. Koska tehtävässä vaihtoehtoisia sijainteja aluksille on 6 ja sijoitettavia alusyksiköitä $2 \cdot 4 = 8$, muodostuu $(6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 \cdot 6 \cdot 5) = 21600$ vaihtoehtoista tapaa sijoittaa alukset tavoitesijainteihin siten, että jokaisessa solmussa on vähintään yksi ohjusvene ja kahdessa solmussa on kaksi. Näille vaihtoehdoille lasketaan virhefunktiolla arvo, jonka perusteella valitaan pienimmän virheen vaihtoehto ja saavutetaan siten optimaalinen sijoittelu. Jotta laskennassa voidaan huomioida tehtävän täyttämisen nopeus ja pienentää huomattavasti laskettavien vaihtoehtojen määrää, voidaan esikarsintana valita ne vaihtoehdot, joissa siirtymät ovat lyhimpien polkujen mukaisia, ja tämän jälkeen laskea virhefunktion suhteen lopullinen optimaalinen vaihtoehto.

Kuvattuun ongelmaan on varsin helppo nähdä intuitiivinen ratkaisu: Pansion tukikohdan alukset siirtyvät lähimpiin solmuihinsa siten, että neljä ohjusvenettä sijoittuvat solmuihin Utö, Jurmo, Bodö ja Örö, kun taas Upinniemestä lähtevät

alukset sijoittuvat solmuihin Russarö ja Jussarö, sijoittuen siten, että molemmissa sijainneissa on kaksi eri suorituskyvyin varustettua alusta. Verrattuna intuitiivisesti optimaaliselta vaikuttavaan ratkaisuun tekoälyn hyödyntäminen ongelman käsittelyyn vaikuttaa turhan monimutkaiselta. Kuitenkin lisättäessä ratkaistavan mallin monimutkaisuutta saavutetaan niin kompleksinen järjestelmä, että optimaalisen ratkaisun löytäminen ihmisen päättelyllä ei ole enää yhtä suoraviivaista.

Jotta tämän kaltaisella päätöksentekoa tukevalla tekoälyjärjestelmällä olisi kyky tuottaa todellista lisäarvoa päätöksenteolle, siihen tulisi kyetä mallintamaan taistelutila kaikkien merkitsevien muuttujien osalta. Vaikuttavia tekijöitä ovat muun muassa muut sodankäynnin dimensiot kuten pintatorjuntakyvyt ja elektronisen tuen kyvyt, täydennysten huoltoikkunat ja näiden allokointi sekä etäisyydet, muiden alusluokkien erityispiirteet, tehtävät, rajoitteet ja vaatimukset. Kompleksisen tila-avaruuden hahmottamisessa ja hallinnassa hyvin toteutettu tekoälyjärjestelmä voidaan argumentoida kykenevän tukemaan päätöksentekoa avustamalla optimaalisen ratkaisun suunnittelemisessa sekä erityisesti ennakoimattomiin tilannekehityksiin reagoimisessa.

LÄHTEET

[1] Kazimierski W., Sawczak A., Wawrzyniak N., *Analysis of Graph Searchin Algorithms for Route Planning in Inland Navigation*, TransNav the International Journal on Maritime Navigation and Safety of Sea Transportation, 9, kesäkuu 2015, s. 281-286, viitattu 6.3.2022, saatavilla: https://www.researchgate.net/publication/281389664_Analysis_of_Graph_Searching_Algorithms_for_Route_Planning_in_Inland_Navigation

[2] Rich E., *Artificial Intelligence*, McGraw-Hill, 1983, Yhdysvallat, ISBN 0-07-052261-8

[3] Russel S., Norvig P., *Artificial Intelligence – A Modern Approach, Third Edition,* Pearson Education Limited, Harlow, Englanti, 2016, ISBN: 978-1292-1453-96-4