Tampereen yliopisto

Matias Ijäs

VALONLÄHTEEN SUUNNAN ESTIMOINTI KASVOKUVASTA

Informaatioteknologian ja viestinnän tiedekunta
Diplomityö
Lokakuu 2020

TIIVISTELMÄ

Matias Ijäs: Valonlähteen suunnan estimointi kasvokuvasta
Diplomityö
Tampereen yliopisto
Information Technology, Data Engineering and Machine Learning
Lokakuu 2020

Valonlähdettä tai sen suuntaa ei ole helppoa havainnoida kasvokuvasta, etenkään kun valonlähde ei ole näkyvissä kuvassa tai mikäli valonlähteitä on useita. Nykyteknologialla ongelmaan on mahdollista tehdä hyviä estimaattoreita. Työn tarkoituksena on ohjelmoida kokonaisuus, joka pystyy arvioimaan ensisijaisen valonlähteen suunnan kuvan keskipisteen suhteen.

Ohjelman käyttötarkoitus on tarkastella valon suuntaa kaksiulotteisesta (2D) kasvokuvasta. Valon suunnan estimointi tapahtuu pikselidatasta, jolle suuntakulma määrää tunnisteen asteina välillä [0, 360). Käyttökohteita ohjelmalle ovat muun muassa auringon suunnan tunnistaminen ulkona otetuista valokuvista, muokattujen kuvien tunnistaminen sekä useiden valonlähteiden sijaintien tunnistaminen useiden kuvien perusteella.

Tutkimuksen tärkeänä osana tarkastellaan ohjelman kehittämistä. Ohjelma koostuu seuraavita asioista: Opetus- ja testausdatan luomisesta virtuaalisesti. Erilaisten koneoppimismenetelmien soveltamisesta ongelman ratkaisemiseksi. Käytännön toteutuksen tekemisestä käyttäen konvoluutionaalista neuroverkkoa (CNN). Tulosten parantamisesta käyttäen esikäsittelyä, kuten kuvan suodattamista, kasvojen havainnointia etukäteen sekä opetusdatan painottamista eri kertoimin.

Työn tarkimmalle mallille käytettiin ohjattua oppimista, jossa optimointifunktiona käytettiin Adamia ja tulosten tappiofunktiona käytettiin keskineliövirhettä eli MSE:tä. Valittu malli koostui viidestä konvoluutiokerroksesta ja kolmesta täysin yhdistetystä kerroksesta. ReLU:a käytettiin aktivointifunktiona. Parametreja oli yhteensä 490 000. Tarkin malli antoi opetusdatan kaltaisesti generoidulle testidatalle keskimääräiseksi virheeksi 4,2 astetta. Oikeille valokuville, joista kasvot oli havaittu, malli antoi keskimääräiseksi virheeksi 15,8 astetta. Oikeat valokuvat olivat peräisin Yalen datasetistä sekä opinnäytetyön tekijän datasetistä.

Projektissa käytetty ohjelmakoodi on avoimesti saatavilla tutkimuskäyttöön osoitteesta https://github.com/NemoHostem/CNN_Source_of_Light.

Avainsanat: CNN, neuroverkko, valonlähde, datan generoiminen, kasvojen havainnointi

ABSTRACT

Matias Ijäs: Estimation of a Light Source from a Facial Image
Master's Thesis
Tampere University
Information Technology, Data Engineering and Machine Learning
October 2020

It is not an easy task to detect the light source or its direction from a facial image, especially if the light source is not visible in the image or if the image contains several light sources. With current technology, it is possible to create good estimators for the problem. The purpose of this work is to create a software program that can estimate the direction of the primary light source relative to the center of the image.

The purpose of the program is to predict the direction of light from a 2-dimensional (2D) face image. The estimation of the light direction happens from pixel data in which the orientation angle determined the label between [0, 360). Applications for the program include estimating the direction of the sun from external photos, detecting edited pictures, and identifying the locations of multiple light sources based on numerous images.

As the primary part of the study, it considered the development of the software. The software consisted of the following things: The creation of training and testing data virtually. The application of different machine learning methods to solve the problem. The development of practical implementation using a convolutional neural network (CNN). The improvement of results using preprocessing such as image filtering, face detection, and weighting of training data.

The most accurate version of the software used supervised learning, where the optimization function was Adam, and the loss function was MSE. The proposed model consisted of five convolutional layers and three fully connected layers. The total number of parameters was 490 000. The most accurate model gave an average error of 4.2 degrees for the test dataset, which consisted of similar data as the training data. For real photos from which had a face detected, the model gave an average error of 15.8 degrees. The photographs were from Yale's dataset as well as the author's dataset.

The program code of the project is openly available for scientific purposes at https://github.com/NemoHostem/CNN_Source_of_Light.

Keywords: CNN, neural network, light source, data creation, face detection

ALKUSANAT

Työn aihe ”Valonlähteen suunnan estimointi kasvokuvasta” oli erittäin mielenkiintoinen ja työympäristö Puolustusvoimien tutkimuslaitoksella Riihimäellä oli erinomainen. Puolustusvoimien tutkimuslaitokselta haluan kiittää ohjaajaani ja esimiestäni TkT Jukka Ruoskasta sekä AMK Jarkko Karista avusta työn suhteen. Lisäksi haluan kiittää Tampereen yliopistolta pääohjaajaani TkT Esa Rahtua tutkimustyön ohjaamisesta sekä hyvistä neuvoista kirjoittamisprosessissa.

Täysin vastaavaa tutkimusta ei ole aiemmin tehty, vaikka useita tieteellisiä tutkimuksia oli tehty muun muassa valonlähteiden paikallistamisesta sekä valon suunnan estimoinnista tunnetun 3D-kappaleen pinnalle. Opetusdatana käytettävää ulkoista datasettiä ei ollut saatavilla, joten realistisen opetusdatan luominen kuului osaksi työtä. Työ pohjautui vahvasti nykyaikaisiin laskennallisiin menetelmiin, kuten regressiivisiin konvoluutionaalisiin neuroverkkoihin. Nämä asiat tekivät työn tekemisestä vaativan.

Tampereella, 18.10.2020

Matias Ijäs

SISÄLLYSLUETTELO

KUVALUETTELO

KAAVALUETTELO

OHJELMALUETTELO

LYHENTEET JA MERKINNÄT

2D	Kaksiulotteisuus (engl. Two-dimensional space), esimerkiksi harmaasävykuva
3D	Kolmiulotteisuus (engl. Three-dimensional space), esimerkiksi kolmiulotteinen pää
3DMM	Kolmiulotteinen muokattava malli (engl. 3D Morphable model)
Adam	Adaptive Moment Estimation -optimointialgoritmi
AI	Tekoäly (engl. Artificial Intelligence)
ANN	Neuroverkko (engl. Artificial Neural Network)
BHT	Tasapainotetun histogrammin kynnysmenetelmä (engl. Balanced Histogram Thresholding)
BP	Vastavirta-algoritmi (engl. Backpropagation)
CIFAR	Datasetti, jota käytetään objektintunnistuksessa (engl. Canadian Institute For Advanced Research)
CMOS	Kanavatransistoreihin perustuva mikropiiritekniikka (engl. Complementary Metal Oxide Semiconductor)
CMYK	Värimalli (engl. Cyan, Magenta, Yellow, Key (Black))
CNN	Konvoluutionaalinen neuroverkko eli konvoluutioneuroverkko (engl. Convolutional Neural Network)
CPT	Ehdollisen todennäköisyyden taulukko (engl. Conditional Probability Table)
CPU	Suoritin eli prosessori (engl. Central Processing Unit)
CSV	Tiedostomuoto, jossa data erotetaan pilkuilla toisistaan (engl. Comma-separated values)
DARPA	Yhdysvaltojen puolustusministeriön virasto, joka vastaa uusien teknologioiden kehittämisestä armeijan käyttöön (engl. Defense Advanced Research Projects Agency)
DCNN	Dynaaminen konvoluutionaalinen neuroverkko (engl. Dynamic Convolutional Neural Network)
DeCAF	Syvän konvoluutionaalisen neuroverkon aktivointiominaisuuksien visualisointityökalu (engl. Deep Convolutional Activation Feature)
DNN	Syvä neuroverkko (engl. Deep Neural Network)
EXIF	Standardi kuvatiedostojen metadatalle (engl. Exchangable image file format)
FPS	Kuvataajuus (engl. Frames per second)
FractalNet	Lähestymistapa CNN-arkkitehtuuriin hajautetuilla konvoluutiokerroksilla
GAN	Generatiivinen kilpaileva verkosto (engl. Generative Adversarial Network)
GDPR	Euroopan Unionin yleinen tietosuoja-asetus (engl. General Data Protection Regulation)
GMM	Gaussin sekoitemalli (engl. Gaussian mixture model)
GPU	Grafiikkaprosessori (engl. Graphics Processing Unit)
HMM	Piilotettu Markovin malli (engl. Hidden Markov model)
HSL	Väriavaruus (engl. Hue, Saturation, Lightness/Luminance)
IBM	Teknologiayritys (engl. International Business Machines Corporation)
IC	Mikropiiri (engl. Integrated Circuit)
ILSVRC	ImageNetin datasetistä koostuva luokittelukilpailu (engl. ImageNet Large-Scale Visual Recognition Challenge)
JPEG	Kuvaformaatti, standardin mukainen pakkausmetodi (engl. Joint Photographic Experts Group)
LSTM	Pitkäkestoinen lähimuisti (engl. Long short-term memory)

LTP	Kestoherkistyminen (engl. Long-term potentiation)
MATLAB	Tietokoneohjelmisto ja ohjelmointikieli, lyhenne sanoista matrix laboratory
MLP	Monikerroksinen perseptroniverkko (engl. Multilayer perceptron)
MNIST	Datasetti, jota käytetään objektintunnistuksessa (engl. Modified National Institute of Standards and Technology database)
MSE	Keskineliövirhe (engl. Mean squared error)
NAS	Neuronien arkkitehtuurinen haku (engl. Neural Architecture Search)
NLP	Luonnollisen kielen käsittely (engl. Natural Language Processing)
NRC	Yhdysvaltojen tutkimusneuvosto (engl. National Research Council)
PNG	Kuvaformaatti (engl. Portable Network Graphics)
PReLU	Parametrinen, lineaarinen aktivointifunktio (engl. Parametric Rectified Linear Unit)
R-CNN	Aluepohjainen konvoluutionaalinen neuroverkko (engl. Region-based Convolutional Neural Network)
RAM	Keskusmuisti (engl. Random-access memory)
ReLU	Lineaarinen aktivointifunktio (engl. Rectified Linear Unit)
ResNet	Lähestymistapa CNN-arkkitehtuuriin, jossa dataa voidaan siirtää kerroksien ylitse identiteettifunktion avulla (engl. Residual Network)
RGB	Värimalli (engl. Red, Green, Blue)
RNN	Takaisinkytketty neuroverkko (engl. Recurrent Neural Network)
ROI	Mielenkiintoalue (engl. Region Of Interest)
RPN	Alueellinen ehdotusverkko (engl. Regional Proposal Network)
SOM	Itseorganisoituva kartta (engl. Self-organizing map)
SRD	Keskeisten alueiden tunnistaminen (engl. Salient Region Detection)
sRGB	Väriavaruus (engl. Red, Green, Blue)
SSH	Kasvojen havaitsemisessa käytetty malli (engl. Single State Headless)
SVM	Tukivektorikone (engl. Support vector machine)
UV-kartoitus	Renderointitekniikka, jossa 2D-tekstuuri asetetaan 3D-malliin, ”U” ja ”V” ovat koordinaatiston akselien nimet (engl. UV-mapping)
VGG	Oxfordin Yliopiston mallin nimi ImageNet-haasteeseen (engl. Visual Geometry Group)
VRN	Tilavuuteen perustuva regressioverkko (engl. Volumetric regression network)
XAI	Ymmärrettävä tekoäly (engl. Explainable Artificial Intelligence)
XOR	” Poissulkeva tai ” -logiikka (engl. Exclusive OR)
YCbCr	Digitaalinen väriavaruus (myös $Y'C_BC_R$), joka koostuu lumasta Y' tai luminanssista Y, sinisen ja luman erotuksesta C_B sekä punaisen ja luman erotuksesta C_R
YPbPr	Analoginen väriavaruus (myös $Y'P_BP_R$), joka koostuu lumasta Y' tai luminanssista Y, sinisen ja luman erotuksesta P_B sekä punaisen ja luman erotuksesta P_R

b	Vakiotermi (engl. bias)
cd	Kandela, valovoiman yksikön tunnus
m	Metri, matkan yksikön tunnus
w	Painoarvot (engl. weights)
X	Neuroverkkoon sisään menevä data (engl. input)
y	Neuroverkosta ulos tuleva data (engl. output)
λ	Aallonpituus

1. JOHDANTO

Koneoppiminen (engl. Machine Learning) ja neuroverkkojen käyttäminen ovat olleet osana eri alojen ohjelmistoprojekteja. Ne ovat olleet räjähdysmäisessä kasvussa 2010-luvun alusta alkaen (Perrault et al. 2019, s. 6–7, 16–17). Tässä työssä hyödynnetään koneoppimista, jonka avulla 2D-kuvasta havainnoidaan valon ominaisuuksia ja kasvojen piirteitä. Neuroverkkojen avulla kuvasta tunnistetaan ensisijaisen valonlähteen tulosuunta.

Koneoppiminen on tieteellinen tutkimushaara tekoälyn (AI, engl. Artificial Intelligence) saralla. Sen perimmäinen tarkoitus on saada luotettavampia ja parempia tuloksia laskennallisiin ongelmiin hyödyntäen taustatietoja ja ongelmaan soveltuvia matemaattisia malleja. Koneoppiminen on pääasiassa matematiikkaa, tietotekniikkaa sekä tilasto- että tietojenkäsittelytiedettä. Yksinkertainen käytännön esimerkki koneoppimisesta on roskapostisuodatin. Siinä opetusaineiston pohjalta luodaan todennäköisyyksiin pohjautuvia malleja yksittäisille sanoille ja usean sanan ketjuille sekä tilastollisia johtopäätöksiä uudelle aineistolle.

Koneoppimisalgoritmit luokitellaan opetusaineiston luonteen perusteella ohjattuun oppimiseen, vahvistusoppimiseen sekä ohjaamattomaan oppimiseen. Käytetyin koneoppisen haara on ohjattu oppiminen (engl. Supervised Learning), jossa opetusdatasta X tiedetään oikea lopputulos y. Esimerkiksi pelimaailmassa on yleisesti käytössä ohjattua oppimista sekä vahvistusoppimista hyödyntäviä algoritmeja. Vahvistusoppiminen (engl. Reinforcement Learning) tapahtuu ohjelman mallin sekä ohjelman ympäristön jatkuvan vuorovaikutuksen avulla. Siinä ympäristöä havainnoiva agentti tarkastaa ja arvioi mallin tuottamia tuloksia maksimoiden palkinnon (Kaelbling et al. 1996).

Ohjaamaton oppiminen (engl. Unsupervised Learning) perustuu opetusdataan, josta ei tiedetä mitään ennalta. Malli etsii samankaltaisuuksia ja tekee päätelmiä opetusdatan pohjalta (Shalev-Shwartz & Ben-David, 2014, s. 22–24). Esimerkiksi generatiivinen kilpaileva verkosto (GAN, engl. Generative Adversarial Network) on ohjaamattoman oppimisen muoto, jossa kaksi neuroverkkoa on vastakkain (Goodfellow et al. 2014). Näistä ensimmäisen neuroverkon tehtävänä on luoda mahdollisimman aitoja valokuvamaisia kuvia ja toisen tehtävänä on erottaa aidot ja neuroverkon luomat kuvat toisistaan (Goodfellow et al. 2014).

Syväoppiminen (engl. Deep Learning) on koneoppimisen osa-alue, joka perustuu usean kerroksen neuroverkkoihin (Patterson & Gibson, 2017, s. 14–63). Neuroverkot pohjautuvat ihmisten aivoissa olevaan neuroniverkkoon, jossa signaalit kulkevat synapsien välityksellä neuronista toiseen. Tämä tapahtuu syötekerroksen, piilotettujen kerrosten ja ulostulokerroksen avulla (Marcus, 2018, s. 3–5). Algoritmillisesti muodostetut neuroverkot koostuvat keinotekoisista neuroneista, niiden painoarvoista, summaajasta, aktivointifunktiosta sekä vakiotermistä (engl. bias). Neuroverkkoja koulutetaan usein ohjatun oppimisen avulla. Opetusvaiheessa neuroverkolle syötetään opetusdataa X ja haluttuja tuloksia y. Opetusdatan avulla neuroverkko muodostaa painoarvoja neuronien välille, jotta haluttu ulostulo y saadaan muodostettua. Neuroverkon kouluttamisen jälkeen sitä voidaan testata uudella testidatalla ja tarvittaessa kouluttaa lisää. Neuroverkot ovat usein yliparametrisoituja (Denil et al. 2013) eli verkon kokoa voisi pienentää ja parametrien määrää voisi vähentää yhtä tarkan tai tarkemman tuloksen saamiseksi. Yliparametrisointi kasvattaa neuroverkon kokoa ja verkon koulutus kestää pidempään kuin pienemmällä verkolla.

Konvoluutionaalinen neuroverkko (CNN, engl. Convolutional Neural Network) on yksi neuroverkkotyyppi, jossa vähintään yksi kerros on konvoluutionaalinen. CNN sopii kuvantunnistamiseen ja kuvien analysointiin, sillä konvoluutionaaliset kerrokset tunnistavat piirteitä kuvan eri alueista. Piirteiden avulla CNN:ä käyttävät ohjelmistot voivat esimerkiksi havaita objekteja kuvasta sekä erotella niitä toisistaan. Konvoluutioneuroverkko ottaa syötteenä kuvan, jonka sisältämä data säilötään tensorina (engl. tensor). Tensori sisältää matriisimaisen rakenteen, johon kuvan pikselien väriarvot tallennetaan. Konvoluutionaalisen neuroverkon piilotetut kerrokset koostuvat konvoluutiokerroksesta (engl. Convolutional layer), tasasuuntaajan aktivoimisfunktiosta, kuten

ReLU:sta (ReLU, engl. Rectified Linear Unit) ja yhdistämiseen käytetystä kerroksesta (engl. Pooling layer). Piilotettujen kerroksien avulla kuvan osien ominaisuuksia tunnistetaan ja datamäärää pienennetään pikselitasolta osakokonaisuuksiksi. Ominaisuuksien toisistaan erottamisen (engl. feature extraction) ja niiden tunnistaminen tapahtuu erilaisten kuvioiden aktivoitumisten ja neuroverkon seuraavalle kerrokselle vievien linkkien painoarvojen avulla. Piilotettujen kerroksien loppupuolella on yleensä vähintään yksi täysin yhdistetty kerros (engl. dense, fully connected layer). Yhdistetyn kerroksen tarkoituksena on yhdistää data vastaamaan lopputulosta y. Konvoluutionaalisten neuroverkkojen tehtävänä on siis rakentaa mahdollisimman hyvä estimaatti käyttäen painoarvoja, jotta syötteenä käytetystä kuvasta X saadaan mahdollisimman lähelle oikeaa tulosta vastaava lopputulos y.

1.1 Kuvadatan analysoiminen

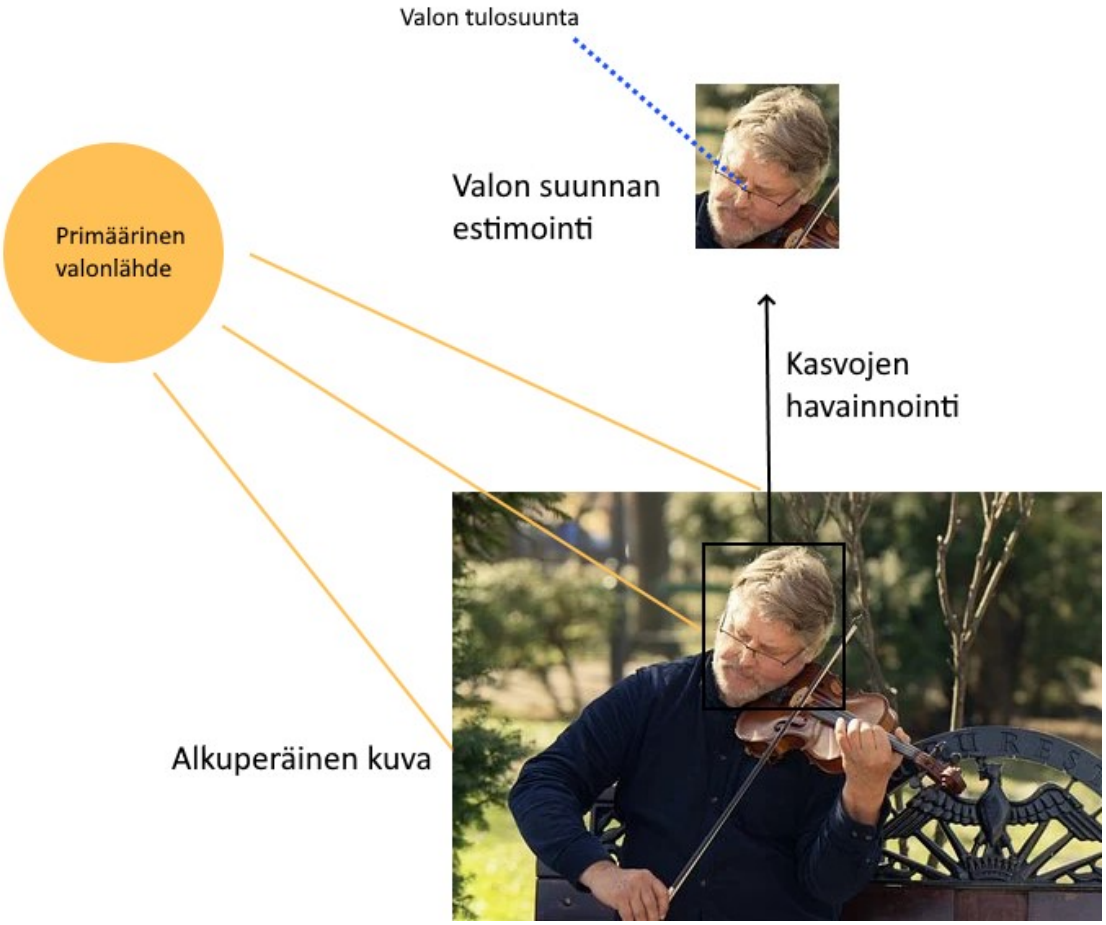
Yksittäinen kuva on digitaalisessa muodossa pakattu, esimerkiksi PNG- tai JPEG-formaattiin. Kuvanpakkaus (engl. Image compression) tehdään digitaalisille kuville tiedonsiirron ja varastoinnin kustannusten pienentämiseksi (Pitas, 2000, s. 3). Kuvassa oleva pakkaamaton data on matriisimaisessa muodossa olevaa numerodataa, joissa numeroiden arvot vastaavat värikanavien pikselikohtaisia arvoja. Värinmittauksen maailma on täynnä erilaisia määritelmiä väreistä (Fairchild, 2005, s. 183). Värien esittämiseksi käytetään värimalleja, kuten RGB tai CMYK, ja väriavaruuksia, kuten HSL ja YPbPr. Väriavaruuksien avulla kuvasta voi painottua erilaisia piirteitä, kuten kirkkautta, luminanssia tai intensiteettiä. Luminanssi on suure, joka kuvastaa pinnan kirkkautta tai pinnasta lähtevää valon voimakkuutta (Selvendy, 2001, s. 1199). Intensiteetti on fysikaalinen suure, joka kuvaa energiamäärän siirtymistä pinta-alayksikön läpi aikayksikössä (Hendee et al. 2003, s. 92). Esimerkiksi YPbPr-avaruudessa luman ja kroman erottamisen myötä videokuvaa voidaan pakata ilman erillisiä värikanavia siten, että kuvan laatu ei kärsi.

Työssä tutkittavaa ongelmaa on tarkasteltu sekä varjojen avulla että kuvan pikselien intensiteetin avulla. Maciej Laskowski muun muassa käyttää intensiteettiä ja Monte Carlo – menetelmää valonlähteiden havainnointiin ja rajoittamiseen (Laskowski, 2007). Korkeat intensiteetit kuvan alueilla kuvaavat usein valonlähteitä, mutta vaaleat pinnat ja heijastukset aiheuttavat virheitä valonlähteiden havainnointiin, mikäli niitä tarkastellaan pelkän intensiteetin avulla. Ruiqi Guon ja hänen työryhmänsä varjojen havainnointiin (engl. shadow detection) ja poistamiseen perustunut työ soveltuisi valonlähteen suunnan tunnistamiseen, sillä siinä paritetaan objektin osia niitä vastaaviin varjoihin (Guo et al. 2011). Useiden kuvassa olevien objektien ja niiden muodostamien varjojen avulla kuvasta olisi mahdollista ennustaa valonlähteen suunta tarkasti.

Peter Nillius ja Jan-Olof Eklundh käyttivät menetelmässään aluksi väri- ja reunatietoja hyödyntäen reunantunnistusta (engl. edge detection), sen jälkeen varjostuksessa käytettävää mallia ja lopuksi Bayesilaisella verkolla (engl. Bayesian network) estimoitiin suurin todennäköisyys valonlähteelle (Nillius & Eklundh, 2001). Vaatimuksena työssä oli, että kuvassa oli objekti, josta oli tunnistettavissa Lambertin pintaheijastuvuus (engl. Lambertian surface reflectance) (Nillius & Eklundh, 2001). Pallomaisille kappaleille varjoja ja valon tulosuuntia oli tarkasteltu Lambertin pallon (engl. Lambertian sphere) avulla.

1.2 Työn tavoitteet ja kuvaus

Työn tekninen osuus tehtiin Puolustusvoimien tutkimuslaitokselle osana laajempaa projektia. Työn päätavoitteena oli kehittää ohjelma, joka konvoluutionaalisia neuroverkkoja käyttäen osaa tunnistaa valonlähteen suunnan kuvasta. Valonlähteen suunnan ja kulman havaitseminen kuvasta mahdollistaa maantieteellisen sijainnin estimoinnin, mikäli kuvan ottamisen aikaleima tiedetään. Lisäksi kuvista, joissa on useita kasvoja, on mahdollista tulkita kuvan aitoutta vertailemalla kasvoihin tulevan pääasiallisen valonlähteen suuntaa. Kuva 1 havainnollistaa työn kontekstia, jossa kasvojen avulla estimoidaan valonlähteen suuntaa.



Kuva 1. *Valonlähteen suunnan estimointi kuvasta.*

Sopivan koulutusaineiston saatavuuden ollessa riittämätöntä, osana projektia tehtiin datan luomisessa käytetty ohjelma. Ohjelman avulla testi- ja opetusdataa voitiin luoda automaattisesti. Opetus- ja testiaineisto tehtiin käyttämällä Face3D-kirjastoa (Feng, 2018) ja Python-ohjelmointikieltä (Van Rossum & Drake Jr, 2020). Face3D-kirjaston käyttäminen ja opinnäytetyön tekijän toteutus datan automaattiseen generoimiseen mahdollistivat valonlähteen määrittämisen 3D-avaruuteen, johon oli keskitetty kolmiulotteinen malli ihmisen kasvoista. Käyttäen UV-kartoitusta (engl. UV-mapping) ja renderointia (engl. rendering), valonlähde asetettiin 3D-avaruuteen, kasvojen pigmentti asetettiin 3D-mallin päälle ja lopuksi kokonaisuudesta muodostettiin kuva JPEG-formaatissa.

Tulevaisuuden kehitystä varten luotiin kasvojen havainnointia (engl. face detection) tekevä automaattinen ohjelma. Sen päätehtävänä oli rajata kuvasta kasvot ja tallentaa kasvokuva tiedostokansioon. Osana työtä tehtiin myös ohjelma mallien tarkkuuksien demonstrointiin. Siinä koulutettuja verkkoja voitiin vertailla ja mallin toimintaa voitiin analysoida visualisointien avulla sekä generoiduilla että aidoilla kasvokuvilla. Lisäksi maantieteellisen tavoitteen takia kehitettiin ohjelma, joka haki halutut parametrit (muun muassa sijaintitiedot) kuvan EXIF-datasta. Yhdessä projektin osa-alueet muodostivat kokonaisuuden, jossa valon suuntaa voitiin estimoida konvoluutionaalista neuroverkkoa hyödyntäen: opetusdatan luomisesta, neuroverkon koulutukseen, testaukseen ja analysointiin visualisointien avulla, jotta käytännön ongelma pystyttiin ratkaisemaan sovelluksen avulla. Tämän diplomityön tarkoituksena oli kuvata ohjelmistoprojektin kehitysprosessi tieteellisestä perspektiivistä ja antaa perustelut päädyttyihin ratkaisuihin hyödyntäen olemassa olevia tieteellisiä tutkimuksia.

1.3 Työn rakenne

Luvussa 2 käsitellään tekoälyn ja koneoppimisen teoriaa sekä käytettyjä teknologioita syväsuuntaavasti perustuen aikaisempiin tutkimuksiin ja julkaisuihin. Alaluvussa 2.1 käydään läpi tekoälyn, koneoppimisen ja syväoppimisen historiaa. Alaluvussa 2.2 perehdytään neuroverkkoihin ja alaluvussa 2.3 keskitytään konvoluutionaalisiin neuroverkkoihin. Luku 3 käsittelee kuvankäsittelyä tutkimuksen näkökulmasta. Alalukuun 3.1 on koottu eri väriavaruuksia ja niiden soveltamista tutkimuksessa. Alaluvussa 3.2 tarkastellaan valoa ja varjoja. Alaluvussa 3.3 perehdytään suotimiin objektintunnistuksen perspektiivistä. Alaluku 3.4 sisältää aiempia tutkimuksia valonlähteen havaitsemisesta kuvista ja siinä keskitytään muihin lähestymistapoihin ongelman ratkaisemiseksi, niiden puutteisiin sekä hyötyihin.

Luvussa 4 käsitellään tutkimuksen toistettavuuteen liittyviä materiaaleja ja menetelmiä. Alaluvut 4.1–4.4 sisältävät datan keräämisen, validoinnin ja esikäsittelyn. Alaluvut 4.5–4.6 sisältävät ohjelman kehittämisen, testaamisen, visualisoimisen sekä vertaamisen vastaavan kaltaisiin ohjelmiin. Luku 5 kertoo yksityiskohtaisesti mallien toiminnasta syötteestä lopputulokseen. Alaluvussa 5.1 tutustutaan syötteeseen eli minkälaista dataa neuroverkko ottaa vastaan. Alaluvussa 5.2 perehdytään verkon rakenteeseen ja mallien kehitykseen projektin aikana. Alaluvussa 5.3 syvennytään valittujen mallien rakenteeseen. Alaluku 5.4 käsittelee koulutuksen lopettamista ja ulostuloa eli milloin neuroverkon koulutus pysäytetään ja minkälaista dataa neuroverkko tuottaa.

Luku 6 käsittelee tuloksia yleisesti. Luvussa verrataan valittujen verkkojen rakenteiden vaikutusta tuloksiin ja annetaan esimerkkejä testidatasta visualisointien avulla. Luvussa 7 analysoidaan tuloksia käyttämällä virhearvioita ja tarkastellaan ohjelman luotettavuutta. Alaluvussa 7.1 tutkitaan virhejakaumia testiseteille. Alaluku 7.2 käsittelee mahdollisia käyttö- ja sovelluskohteita kehitetylle ohjelmalle. Alaluvussa 7.3 analysoidaan koko projektia kriittisesti. Alaluku 7.4 sisältää juridisia rajoitteita, jotka voivat estää ohjelman käytön tilannekohtaisesti. Luku 8 kokoaa yhteenvedon työstä.

2. TEKOÄLY JA KONEOPPIMINEN

Tietojenkäsittelytieteessä (engl. Computer science) tekoäly määritellään ”älykkäiden agenttien” (engl. Intelligent Agents) tutkimisena. Älykkäiden agenttien merkitys tekoälyn kontekstissa on laite, joka maksimoi tavoitteidensa saavuttamisen hyödyntäen ympäristöään ja käyttäen mahdollisia toimintojaan (Poole et al. 1998, s. 1–5).

Koneoppiminen (ML, engl. Machine Learning) mielletään osaksi tekoälyä. Koneoppimista kuvataan tietokonealgoritmien tutkimuksena, jossa algoritmit kehittyvät automaattisesti kokemuksen kautta (Mitchell et al. 1997, s. 1–9). Koneoppimisessa käytetty algoritmi rakentaa matemaattisen mallin opetusdatan pohjalta, jotta algoritmi voi ennustaa lopputulosta tai tehdä päätöksiä ilman tarkkaa ohjelmoinnissa tehtyä käskyä (Bishop, 2006, s. 1–4, 32).

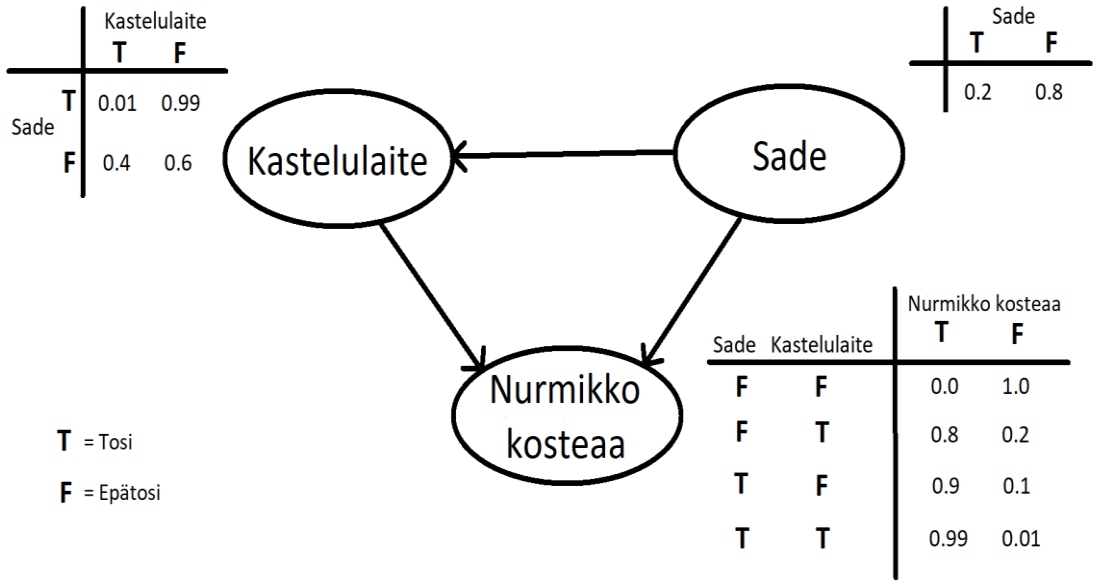
2.1 Tekoälyn historia

Tekoälyn tieteellisenä pohjana voidaan pitää Alan Turingin kehittämää laskettavuusteoriaa (engl. Theory of Computation), joka luotiin 1940-luvun lopussa. Laskettavuusteoria on matematiikan ja tietojenkäsittelyteorian haara, jonka tarkoituksena on tutkia kuinka tehokkaasti ongelman voi ratkaista laskennallisella mallilla algoritmia käyttäen (Sipser, 2012, s. 1–4). Yleisimmin tutkittuna laskennallisen mallin esimerkkinä toimii vuosina 1937–1945 kehitetty Turingin kone (Hodges, 2012, s. 294–297, 324). Vuonna 1950 Alan Turing kehitti Turingin testin (engl. Turing test), jossa testattiin koneen kykyä luoda älykästä käyttäytymistä verrattuna ihmiseen (Turing, 1950, s. 460). Turingin testi on tekoälyn kannalta yksi tärkeimmistä konsepteista, sillä sitä on käytetty testaamaan tekoälyn toimintaa kysymällä ihmisiltä, onko vastapuolella ihminen vai botti. Vuoden 1956 Dartmouthin konferenssia pidetään termin ”tekoäly” syntynä, jossa John McCarthy keksi sen sanana ja määritelmän sille (Crevier, 1993, s. 49). Arthur Samuel kehitti vuonna 1959 koneoppimisen termin. Hän määritteli, että kone ei saa olla erityisesti ohjelmoitu tehtävään (Koza et al. 1996).

1950-luvusta 1990-luvun loppuun tekoälyjen pääasiallinen tehtävä oli esittää tieto tarkoin määritellystä datasta. Rajallinen laskentateho ja tietokoneiden rajoitteisuus vaikuttivat tekoälyn kehityskaareen. Tutkimuksen hitaan kehityksen vuoksi vuosina 1974–1980 oli ensimmäinen tekoälyn kylmä kausi (engl. AI Winter). Tekoälyn kylmään kauteen johti muun muassa Lighthillin raportti (engl. Lighthill report) sekä DARPA:n rahoituksen alasajo (Crevier, 1993, s. 115–117) (Russell & Norvig, 2003, s. 22). Lighthillin raportissa annettiin pessimistinen näkemys tekoälyn tulevaisuudesta muun muassa väittämällä, että suurta vaikutusta ei ole tapahtunut, vaikka niin oli luvattu (Lighthill, 1973). 1980-luvulta lähtien asiantuntijajärjestelmät (engl. Expert Systems) nousivat suosioon onnistuneen markkinointikampanjan vuoksi ja saivat tekoälymarkkinat uuteen nousuun (Crevier, 1993, s. 197–203) (Newquist, 1994, s. 155–183). Toinen tekoälyn kylmä kausi tapahtui vuosina 1987–1993, kun Lisp-koneiden (engl. Lisp machines) markkinat romahtivat, rahoitus peruutettiin tekoälyn saralta Strategic Computing Initiative:n taholta ja asiantuntijajärjestelmät aiheuttivat vastustusta (McCorduck, 2004, s. 430–435) (Crevier, 1993, s. 209–210) (Newquist, 1994, s. 301–318). Virstanpylväitä saavutettiin kuitenkin toisen tekoälyn kylmän kauden aikana, kun 1980-luvun lopussa kanavatransistoreiden kehityksen avulla koneoppiminen yhdistettiin matalalla tasolla neuroverkkoihin ja ensimmäisten toiminnallisten keinotekoisten neuroverkkojen (ANN, engl. Artificial Neural Network) luominen oli mahdollista (Mead & Ismail, 1989, s. 99–100). Kanavatransistoreihin perustuva mikropiiritekniikka CMOS (engl. Complementary Metal Oxide Semiconductor) mahdollisti riittävän laskentatehon. Vuonna 1988 Davari johti IBM:n tutkimusryhmää, joka esitteli tehokasta 250 nanometrin CMOS-prosessoria (Davari et al. 1988). Mikropiirien jatkuvan kehittymisen takia 1990-luvulla tilastollinen koneoppiminen ja Big datan käsittely liitettiin osaksi tekoälyä ja tilastojen tekeminen yleistyi informaatiotekniikan alalla (Mashey, 1997). 1990-luvun puolivälissä tekoälyä aloitettiin käyttämään apuna myös muilla aloilla, kuten datan keräämisessä, logistiikassa ja lääketieteellisessä analyysissä (Newquist, 1994, s. 189–201) (Russell & Norvig, 2003, s. 28). Jatkuvaa kehitystä tekoälyn saralla voidaan pitää mahdollisena Mooren lain myötä, sillä Mooren approksimaatiossa transistorien määrä mikropiirissä (IC, engl. integrated circuit) tuplaantuu kerran kahdessa vuodessa (Moore, 1965).

Tekoälyjen suhteen ensimmäinen kuuluisa työ syntyi vuosina 1947–1959 Arthur L. Samuelin toimesta, kun hän kehitti botin, joka pelasi tammea (engl. Draughts, Checkers). Vuonna 1952 Christopher Stracheyn tammea pelaava botti oli julkaistussa muodossa. Samuelin botti oli vielä kehitysvaiheessa ja hän sanoi kokevansa tappion, sillä hän ei ollut saanut bottiaan ensimmäisenä toimivaksi (Schaeffer, 2009). Samuel kuitenkin jatkoi tekoälyn kehitystä ja vuoden 1954 tulokset olivat hänen mukaansa mielenkiintoisia. Vuonna 1959 Samuel julkaisi työnsä ja väitti, että hänen tammi-bottinsa pelasi paremmin kuin keskivertoihminen (Samuel, 1959). Vuonna 1966 Joseph Weizerbaum esitteli keskustelubotin (engl. chatbot) nimeltä ELIZA (Braun, 2013). Se oli näytölle kirjoittava botti, joka pystyi vastaamaan viesteihin ja näin ollen ”keskusteli” ihmisen kanssa. 1980-luvun alussa asiantuntijajärjestelmät valtasivat tekoälymarkkinoita onnistuneen mainoskampanjan toimesta (McCorduck, 2004, s. 327–335, 427–435). Tekoälyä kohtasi suuri mullistus näkyvyyden kannalta vuonna 1997, kun IBM:n kehittämä tekoäly DeepBlue voitti shakkimestari Garry Kasparovin shakissa (McCorduck, 2004, s. 480–483).

1990-luvun lopulla ja 2000-luvun alussa tekoälyyn perustuvia algoritmeja kehitettiin ympäri maailmaa. Muun muassa Markovin ketjua (engl. Markov chain) ja Monte Carlo -menetelmää (engl. Monte Carlo method) hyödynnettiin älykkäissä agenteissa ja todennäköisyyksiin perustuvissa tekoälysovelluksissa (Russell & Norvig, 2003, s. 28). Asiantuntijajärjestelmät, neuroverkot ja adaptiiviset järjestelmät (engl. Adaptive Systems) nostattivat tekoälyn suosiota teollisuudessa. 1990-luvun lopulla tekoäly ei ollut tutkimuskohteena selkeästi näkyvissä, sillä tekoälyä hyödyntävät osat olivat sulautettuina suurempiin ohjelmistoihin (National Research Council (NRC), 1999, s. 216–220). Yhdysvaltojen tutkimusneuvosto NRC toteaa, että tekoälyä käyttävät ohjelmistot eivät ole osoittaneet ihmismäistä älykkyyttä, mutta käyttökohteita on sovellettu muun muassa mainosteollisuudessa ja sotilaallisiin tarkoituksiin (National Research Council (NRC), 1999, s. 216). Muun muassa Lumiere-projektia kehitettiin vuodesta 1993 Microsoftilla ja se saatiin käyttöön Microsoft Office 97 – ohjelmaan. Projektissa Lumiere - assistentti käytti Bayesilaista verkkoa todennäköisyysjakaumien muodostukseen eri aihealueiden piirissä (National Research Council (NRC), 1999, s. 216–220). Bayesilaisen verkon todennäköisyysfunktio kahdella eri tapahtumalla kuvan 2 tilanteessa lasketaan kaavalla 1.



Kuva 2. *Yksinkertainen Bayesilainen verkko ehdollisen todennäköisyyden taulukoilla. Todennäköisyydet pohjautuvat vasempien lohkojen totuusarvoihin.*

Kuvassa 2 ehdollisen todennäköisyyden taulukko (CPT, engl. conditional probability table) määrittää todennäköisyydet jokaiselle tapahtumalle siten, että jokaisen yksittäisen tapahtuman todennäköisyys on välillä $[0, 1]$. Lisäksi yksittäisten tapahtumien todennäköisyyksien summa on 1 ja kaikkien eri tapahtumien kombinaatioiden todennäköisyyksien summa on 1. Esimerkiksi jos sataa ja kastelulaite on päällä, nurmikko on kosteaa 99 %:n varmuudella. Kokonaistodennäköisyys koko kombinaatiolle on kuitenkin $0{,}2 \times 0{,}01 \times 0{,}99 \times 100\ \% = 0{,}198\ \%$.

$$\Pr(N, K, S) = \Pr(N|K, S)\Pr(K|S)\Pr(S)$$

Kaava 1. *Bayesilaisen verkon ehdollisen todennäköisyyden laskeminen.*

Kaavassa 1 Pr tarkoittaa todennäköisyyttä, N nurmikon kosteutta, K kastelulaitetta ja S sadetta. Ehdollisen todennäköisyyden vuoksi sade vaikuttaa sekä suoraan että välillisesti nurmikon kosteuteen, sillä sade vaikuttaa myös kastelulaitteeseen. Bayesilainen malli on tilastotieteessä käytetty todennäköisyyksiin pohjautuva malli, jonka avulla voidaan tarkastella luotettavuutta. Tilastollinen malli muodostetaan tekemällä todennäköisyyksiin pohjautuvia päätelmiä aineiston pohjalta (Davidson, 2003, s. 163–165).

Naiivi Bayesilainen luokittelu (engl. Naive Bayesian classifier) perustuu Thomas Bayesin lauseeseen ennustajien välisten riippumattomuusoletusten kanssa. Naiivi Bayesilainen malli on helppo rakentaa ilman monimutkaista iteratiivista parametriarviointia, mikä tekee siitä erityisen hyödyllisen erittäin suurille tietojoukoille. Yksinkertaisuudesta huolimatta naiivi Bayesilainen luokittelija toimii usein yllättävän hyvin ja sitä käytetään laajalti, koska sen tarkkuus ylittää usein kehittyneemmät luokittelumenetelmät. Kaavassa 2 $P(c|x)$ on luokan antaman ennustajan posteriori-todennäköisyys (engl. posterior probability), $P(c)$ on luokan aikaisempi todennäköisyys, $P(x|c)$ on todennäköisyys, jonka ennustaja antaa tietylle luokalle ja $P(x)$ on ennustajan aiempi todennäköisyys.

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

Kaava 2. *Naiivi Bayesilainen luokittelu.*

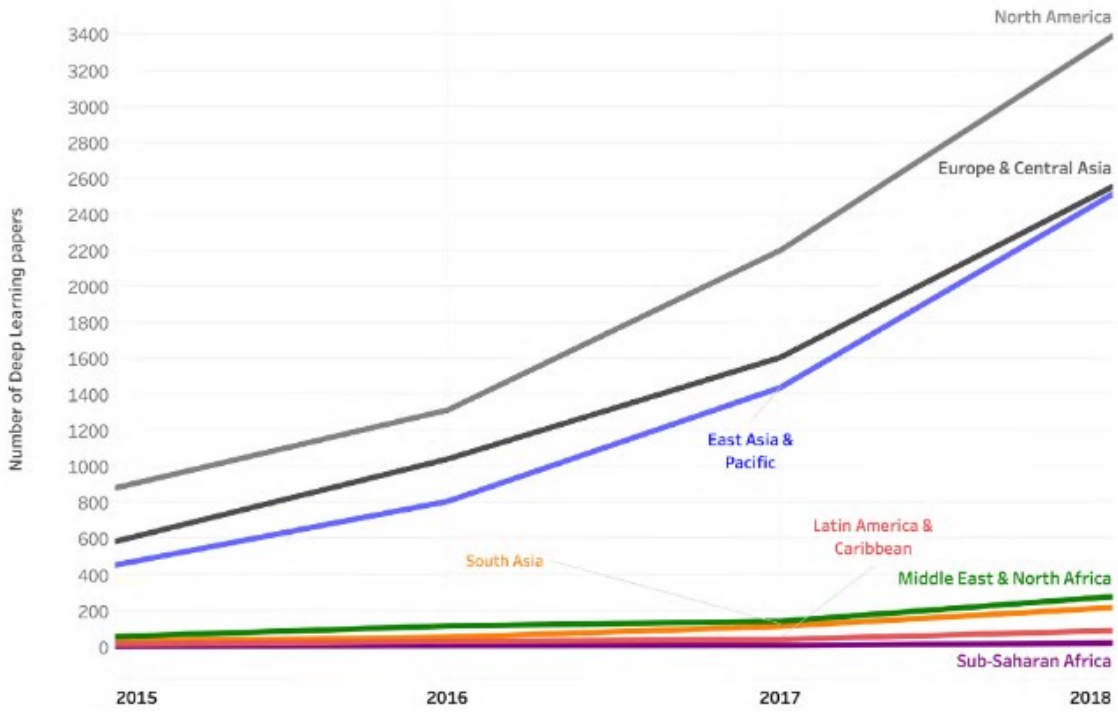
Bayesilainen optimaalinen luokittelija (engl. Bayes optimal classifier) tulee kaavan 3 mukaan. Kaavassa x vastaa luokiteltavaa asiaa, H vastaa hypoteesiavaruutta ja D vastaa luokiteltavien asioiden aiempia esiintymisiä.

$$argmax \sum_{h_i \in H} P(x|h_i) \times P(h_i|D)$$

Kaava 3. *Bayesilainen optimaalinen luokittelija.*

2010-luvulla prosessorien tehokkuuden kasvaminen sekä pääsy valtaviin datamääriin mahdollistivat harppauksen koneoppimisen saralla. Vuonna 2011 IBM:n Watson tietokone voitti Jeopardy! – viihdepelin taitavia ihmispelaajia vastaan. Lopputuloksissa Watsonin tulos oli 77 147 dollaria, kun ihmispelaajilla oli 24 000 ja 21 600 dollaria (Markoff, 2011). Watson oli ensimmäinen tekoäly, joka ymmärsi ihmiskielisen kysymyksen ja osasi antaa oikean ihmiskielisen vastauksen.

Vuonna 2012 syväoppimista hyödyntävät menetelmät alkoivat dominoimaan muita loogisia operaatioita käyttäviä menetelmiä (McKinsey & Company, 2017). Esimerkiksi ImageNet objektintunnistamiskilpailussa ILSVRC syviä neuroverkkoja (engl. Deep Neural Networks) käyttävä menetelmä voitti top-5 kuvantunnistuskilpailun 10,9 prosenttiyksikön erolla toiseksi parhaaseen virheen ollessa 15,3 % verrattuna toisen sijan 26,2 % virheeseen (Krizhevsky et al. 2012). Jack Clarkin mukaan Google kasvatti tekoälyä hyödyntäviä projekteja 2 700:n kappaleeseen, sillä muun muassa virheprosentti on pienempi kuvankäsittelyn tehtävissä (Clark, 2015). Microsoftin Skypen automaattinen kielenkääntäminen ja Facebookin toiminto, jossa kuva selitetään sokealle ihmiselle, olivat myös selkeitä tekoälyn voimannäyttöjä (Clark, 2015). Vuonna 2016 tekoäly AlphaGo voitti neljä ottelua viidestä Go-mestari Lee Se-dolia vastaan samalla ollen ensimmäinen tietokone, joka voitti ammattilaisen Go-pelaajan ilman apua (BBC News, 2016). Vuodesta 2016 Hanson Roboticsin tuotteistama Sophia-niminen humanoidi robotti on esiintynyt julkisuudessa maailman ympäri (Greshko, 2018). Sophia on robottina pilottitutkimuksessa, jonka tarkoituksena on kehittää ohjelmistoa siten, että robotti voi olla vuorovaikutuksessa ihmisen kanssa rakastavalla ja myötätuntoisella tavalla, edistää ihmisten ymmärrystä itsestään ja kannustaa ihmisiä ylittämään itsensä (Goertzel et al. 2017). Syväoppiminen tutkimuskohteena on ollut selvässä kasvussa viime vuosina (Perrault et al. 2019, s. 23–24), kuten kuvasta 3 havaitaan.



Kuva 3. *Syväoppimista koskevien tutkimusten määrät vuosilta 2015–2018 (Perrault et al. 2019, s. 23).*

Syväoppimisessa kritiikin kohteena on ollut mustalaatikkomainen (engl. black box) tapa esittää verkko. Musta laatikko tarkoittaa sitä, että komponentille voidaan antaa syötteitä ja niistä saadaan vain ulostulo, ilman teknistä tietämystä mitä komponentin sisällä tapahtuu (Bunge, 1963). Euroopan Unionin yleinen tietosuoja-asetus (GDPR, engl. General Data Protection Regulation) suojaa henkilökohtaisen datan keräämistä, käyttämistä ja välittämistä (Council Regulation (EU) 2016/679, 2016). GDPR hankaloittaa mustalaatikkomenetelmiä, sillä menetelmä ei pysty selittämään miksi päätös on tehty (Holzinger at al. 2017). Syväoppimiseen luottavien järjestelmien tulee usein generalisoitua nähdyn datan lisäksi sanojen uusiin ääntämisiin tai kuviin, jotka eroavat neuroverkon aiemmin näkemistä kuvista (Marcus, 2018, s. 6–7). Järjestelmässä, missä tietoja on vähemmän kuin ääretön, muodollisten todistusten kyky taata korkealaatuinen suorituskyky on rajallisempi (Marcus, 2018, s. 6–7). Koska syväoppimista hyödyntävää järjestelmää ei voi kouluttaa kaikella datalla, on sen suorituskyky rajallisempi ja varmuutta sen generalisoitumisesta eli sopeutumisesta uuteen dataan ei ole.

1990-luvun puolivälistä alkaen on tutkittu neuroverkkojen läpinäkyvyyttä ja sitä, miten tietoa saisi ulos koulutetusta ANN:sta (Tickle et al. 1998). 2010-luvulla tekoälyn konseptuaalinen päättely (XAI, engl. Explainable AI) nousi julkisesti huolenaiheeksi muun muassa rotuun perustuvan puolueellisuuden (engl. racial bias) takia (Sample, 2017). Julkisuuskohun, GDPR:n ja verkon sisäisen puutteellisen ymmärryksen takia, monet yritykset, kuten Accenture, alkoivat kehittää työkaluja, joiden avulla puolueellisuutta voidaan havaita heidän systeemeistänsä (Kahn, 2018). XAI mahdollistaa käyttäjien ja sisäisten järjestelmien olevan läpinäkyvämpiä, tarjoamalla selityksiä sen päätöksiin tietyllä yksityiskohtaisuuden tasolla (Giplin et al. 2018). Nämä selitykset ovat tärkeitä algoritmillisen oikeudenmukaisuuden varmistamiseksi, puolueellisuuden havaitsemiseksi, ongelmallisen opetusdatan tarkastamiseksi sekä algoritmien odotetun toiminnallisuuden varmistamiseksi (Giplin et al. 2018).

2.2 Neuroverkot

Keinotekoiset neuroverkot ovat laskentajärjestelmiä, joiden luomisessa inspiraatiota on saatu biologisista hermoverkoista. Neuroverkot oppivat suorittamaan tehtäviä esimerkkien avulla, yleensä ilman ohjelmoituja tehtäväkohtaisia sääntöjä. Esimerkiksi kuvantunnistuksessa neuroverkko voidaan opettaa erottamaan autot muista objekteista. Tällaisessa tapauksessa neuroverkolla ei ole tietoa, että autossa pitää olla kori, pyöriä ja moottori, vaan neuroverkko opetetaan kuvilla, jotka merkitään binäärisellä luokittelulla (engl. binary classification) luokaksi ”auto” tai ”ei auto”. Esimerkin tapauksessa kaikki opetusaineiston kuvat eli syötteet $[X_1, \dots, X_n]$ liitetään niitä vastaaviin merkintöihin eli ulostuloihin $[y_1, \dots, y_n]$ siten, että kuva autosta vastaa merkintää ”auto” ja kuva esimerkiksi eläimestä vastaa merkintää ”ei auto”.

Teknisesti keinotekoiset neuroverkot perustuvat kokoelmaan toisiinsa kytkettyjä solmuja, joita kutsutaan keinotekoisiksi neuroneiksi. Keinotekoiset neuronit mallintavat etäisesti biologisten aivojen sisältämiä hermosoluja. Jokainen yhteys, kuten biologisten aivojen synapsit, voi lähettää signaalin muihin neuroneihin. Keinotekoinen neuroni, joka vastaanottaa signaalin, prosessoi signaalin ja voi lähettää sen eteenpäin siihen kytkettyihin neuroneihin. Tiivistettynä neuroverkot koostuvat yksinkertaisista kytketyistä neuroneista, joista jokainen tuottaa reaaliarvoisten aktivointien sekvenssin (Schmidhuber, 2015). Neuroverkoissa oppiminen tapahtuu yhteyksien painoarvoja säätämällä. Painoarvoja säädetään suoraan opetusdatan avulla, ilman oletuksia datan statistisesta jakaumasta tai muista piirteistä (Beale et al. 1996).

Keinotekoisten neuroverkkojen historia alkoi vuonna 1943 Warren McCullochin ja Walter Pittsin toimesta. He tekivät neuroverkoille laskennallisen mallin, joka perustuu kynnyslogiikaksi (engl. threshold logic) kutsuttuun algoritmiin (McCulloch & Pitts, 1943). Laskennallinen malli johti tutkimuksen jakautumisen kahteen lähestymistapaan: biologisiin prosesseihin ja neuroverkkojen soveltamiseen tekoälyn saralla. Vuonna 1951 Kleene julkaisi tutkimuksen, jossa pohdittiin neuroverkkojen reagointia ärsykkeisiin sekä tilakoneen äärellisyyttä (Kleene, 1951). Äärellinen automaatti (engl. Finite-state machine) eli tilakone voi muuttua tilasta toiseen reaktiona sisääntuloihin; muutosta tilasta toiseen kutsutaan siirtymäksi (Wang, 2019, s. 34).

1940-luvun lopulla Hebb esitti oppimishypoteesin, joka perustui hermoston plastiikkamekanismiin (Hebb, 2005). Menetelmä tuli tutuksi nimellä Hebbin teoria (engl. Hebbian learning). Hebbin teoria on koneoppimisessa osa ohjaamatonta oppimista, ja psykologiassa siitä kehitettiin malleja pitkäkestoiseen potentiaatioon eli kestoherkistymiseen (LTP, engl. long-term potentiation). Tutkijat sovelsivat Hebbin teoriaa laskennallisiin malleihin Turingin B-tyypin koneella vuonna 1948. Vuonna 1954 Farley ja Clark käyttivät Hebbin verkon simuloimiseksi ensin laskennallisia koneita, joita kutsuttiin laskimiksi (Farley & Clark, 1954). Muita neuroverkkolaskentakoneita ovat luoneet Rochester, Holland, Habit ja Duda vuonna 1956 (Rochester et al. 1956). Vuonna 1958 Rosenblatt ideoi perseptronin (engl. perceptron), joka on, etenkin kuvantunnistuksessa, binäärisessä luokittelussa käytetty algoritmi (Rosenblatt, 1958). Toisaalta McCulloch ja Walter Pitts kuvasivat perseptronin toiminnan kaltaista kynnyslogiikkaa kirjallisuudessa jo 1940-luvulla (McCulloch & Pitts, 1943). Ensimmäiset toimivat Hebbin teoriaa hyödyntävät monikerroksiset neuroverkot julkaistiin Ivaknenkon ja Lapan toimesta vuonna 1965 (Schmidhuber, 2015). Ivaknenko ja Lapa käyttivät neuroverkkoa deterministisiin prosesseihin sekä satunnaisprosesseihin (Ivaknenko & Lapa, 1967). Neuroverkkoja koskeva tutkimus pysähtyi 1970-luvulla ryhmän Parallel Distributed Processing koneoppimista koskeviin tutkimuksiin, joissa he löysivät kaksi ongelmakohtaa neuroverkkoja käsittelevien laskentakoneiden kanssa (Minsky & Papert, 2017, s. 247–248, 265–266). Ensimmäinen ongelma oli se, että perseptronit eivät kyenneet käsittelemään poissulkevaa taiporttia (XOR, engl. Exclusive or gate). Toinen ongelma oli se, että tietokoneilla ei ollut tarpeeksi laskentatehoa suurten neuroverkkojen käsittelemiseksi. Neuroverkkoja koskeva tutkimus hidastui, kunnes tietokoneet saavuttivat huomattavasti suuremman laskentatehon.

Vastavirta-algoritmi (BP, engl. backpropagation) terminä tuli viralliseksi 1980-luvulla Rumelhartin, Hintonin ja Williamsin toimesta (Rumelhart et al. 1985). Vastavirta-algoritmeissa neuroverkon neuronit antavat palautteen edellisen kerroksen neuroneille muuttaen niiden painoarvoja. BP:n käyttö on yleistä eteenpäin kytketyissä verkoissa (engl. feedforward neural network). Neuroverkon opetusvaiheessa BP laskee tappiofunktion gradientin painoarvojen suhteen yhdelle

”syöte-ulostulo” –esimerkille. Verrattuna gradientin naiiviin suoraan laskentaan jokaiseen painoarvoon erikseen, laskenta on tehokas. Kaavassa 4 annetaan esimerkki tappiofunktiosta, jota voitaisiin käyttää koneoppimisessa.

$$f(w,b) = \frac{1}{N}\sum_{i=1}^{n}\big(y_i - (wx_i + b)\big)^2$$

Kaava 4. *Esimerkkinä toimiva tappiofunktio, jossa w on painoarvot ja b on vakiotermi.*

Kaavassa 5 lasketaan esimerkkifunktion (kaava 4) gradientti käyttämällä gradienttimenetelmää (engl. gradient descent method). BP on tehokkaampi, sillä gradientin ratkaisemiseksi prosessissa iteroidaan datapisteiden läpi käyttäen uusia w ja b arvoja ja lasketaan osittaisderivaatat. Tämä uusi gradientti kertoo tappiofunktion kaltevuuden nykyisessä sijainnissa sekä suunnan, johon tulisi siirtyä parametrien päivityksen optimoimiseksi (Ruder, 2016). Parametrien päivityksen kokoa ohjaa oppimisnopeus (engl. learning rate). Gradienttimenetelmää käytetään usein mustana laatikkona, sillä monet suositut optimointialgoritmit kuten Momentum, Adagrad ja Adam käyttävät sitä (Ruder, 2016).

$$f'(w,b) = \begin{bmatrix}\frac{df}{dw}\\ \frac{df}{db}\end{bmatrix} = \begin{bmatrix}\frac{1}{N}\sum -2x_i\big(y_i - (wx_i + b)\big)\\ \frac{1}{N}\sum -2\big(y_i - (wx_i + b)\big)\end{bmatrix}$$

Kaava 5. *Gradientin laskeminen esimerkille käyttämällä gradienttimenetelmää.*

Adam (Kingma & Ba, 2015) on kehitetty stokastiseen optimointiin ja siinä käytetään gradienttimenetelmää. Adam tallentaa aikaisempien neliögradienttien eksponentiaalisesti hajoavan keskiarvon v_t, kuten Adadelta ja eksponentiaalisesti hajoavan keskiarvon menneistä kaltevuuksista m_t, kuten Momentum (Ruder, 2016). Momentumia voidaan pitää pallona, joka juoksee rinteessä, mutta Adam käyttäytyy kuin raskas pallo kitkalla, mikä mieluummin suosii minimiä virhepinnalla (Heusel et al. 2017). Kaavoissa 6 ja 7 lasketaan hajoavat keskiarvot aikaisemmista gradienteista m_t ja neliöidyistä gradienteista v_t.

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$$

Kaava 6. *Aikaisempien neliöityjen gradienttien eksponentiaalisesti hajoava keskiarvo.*

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$$

Kaava 7. *Aikaisempien gradienttien eksponentiaalisesti hajoava keskiarvo.*

v_t ja m_t ovat estimaatteja gradientin ensimmäisestä ja toisesta momentista eli keskiarvosta ja keskittämättömästä varianssista. Kun v_t ja m_t alustetaan nollavektoreiksi, Adamin kehittäjät huomasivat, että vektorit ovat puolueellisia nollaa kohti, etenkin alkuvaiheiden aikana ja varsinkin hajoamisnopeuksien ollessa pieniä eli β_1 ja β_2 ollessa lähellä lukuarvoa 1 (Ruder, 2016). Adamin kehittäjät torjuivat nämä vinoumat laskemalla oikaistun ensimmäisen ja toisen momentin estimaatit kaavoissa 8 ja 9.

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

Kaava 8. *Oikaistu aikaisempien neliöityjen gradienttien hajoava keskiarvo.*

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$$

Kaava 9. *Oikaistu aikaisempien gradienttien hajoava keskiarvo.*

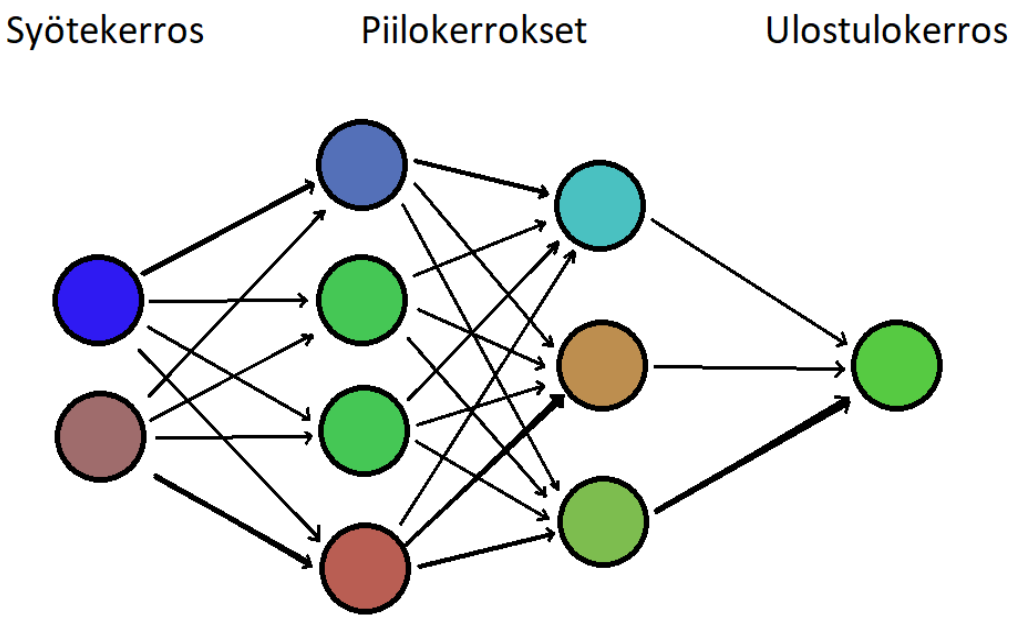
Kaavoja 8 ja 9 käytetään parametrien päivitykseen samalla tavalla kuin Adadeltassa. Adamin päivittäminen tapahtuu kaavalla 10, jossa kokonaiskaava koostuu säädettävistä vakioparametreistä β_1, β_2 ja ϵ.

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t$$

Kaava 10. *Adamin päivityksessä käytettävä laskentakaava.*

Adamin kehittäjät ehdottivat vakioarvoiksi $0{,}9$ β_1:lle, $0{,}999$ β_2:lle ja 10^{-8} ϵ:lle. Adamin toimivuus on osoitettu empiirisesti lukuisissa ohjatun oppimisen töissä (Krizhevsky et al. 2012) (Ioffe & Szegedy, 2015). Adam toimii käytännöllisesti ja Adamin tulokset ovat parempia verrattaessa muihin mukautuvan oppimisen algoritmeihin (Ruder, 2016).

Eteenpäin kytketyt verkot ovat neuroverkkoja, joissa yhteydet solmujen välillä eivät muodosta silmukoita (Zell, 1994, s. 73–75). Sellaisenaan, se eroaa takaisinkytketyistä neuroverkoista (RNN, engl. recurrent neural network), joissa yhteydet muodostavat silmukoita. Eteenpäin kytketty verkko oli ensimmäinen ja yksinkertaisin suunniteltu ANN (Schmidhuber, 2015). Yksinkertaisin neuroverkko on yksikerroksinen perseptroni verkko, joka koostuu yhdestä kerroksesta ulostulon solmuja. Verkossa sisään menevä data eli syötekerros linkittyy suoraan ulostuloon siten, että ulostuloon vaikuttaa ainoastaan sisään menevän datan lisäksi painoarvot syötekerroksen ja ulostulokerroksen välillä. Painoarvojen ja syötteiden tulojen summa lasketaan kussakin solmussa ja jos arvo on valitun kynnysarvon yläpuolella, neuroni aktivoituu ja ottaa aktivoidun arvon. Muuten neuroni ottaa deaktivoidun arvon. Tällaisen aktivointitoiminnan omaavia neuroneita kutsutaan keinotekoisiksi neuroneiksi tai lineaarisiksi kynnysyksiköiksi (McCulloch & Pitts, 1943). Kuvassa 4 on esimerkki eteenpäin kytketystä verkosta, jossa syötekerroksen kaksi arvoa välittyy piilokerroksen neuroneille eri tavoin painoarvojen takia.



Kuva 4. *Eteenpäin kytketty neuroverkko. Syötteistä muodostetaan painoarvojen ja piilokerrosten neuronien avulla ulostulo.*

Perseptroni voidaan luoda käyttämällä mitä tahansa aktivoitujen ja deaktivoitujen tilojen arvoja, kun valittu kynnysarvo on näiden kahden arvon välillä. Usein kynnysarvona käytetään nollaa, aktivoituna arvona lukua 1 ja deaktivoituna arvona lukua -1. Perseptroneja voidaan kouluttaa yksinkertaisella oppimisalgoritmilla, jota kutsutaan deltasäännöksi (engl. delta rule). Delta-sääntö

laskee virheet lasketun ulostulon ja näytteen ulostulodatan välillä käyttäen sitä painoarvojen säätämiseen. Algoritmi implementoituna vastaa gradienttimenetelmää eli kaltevuuskulman laskemista ja optimoimista virheen minimoimiseksi.

Koneoppimisessa deltasääntö on gradienttimenetelmän oppimissääntö, jolla päivitetään keinotekoisten neuroneiden painoarvot yksikerroksisessa neuroverkossa (Russell, 2012). Se on erityistapaus yleisemmästä BP-algoritmista. Neuronille j ja aktivointifunktiolle $g(x)$, deltasääntö j:n i:nnelle painoarvolle w_{ji} tulee kaavan 11 mukaan.

$$\Delta w_{ji} = \alpha(t_j - y_j)g'(h_j)x_i$$

Kaava 11. *Deltasääntö gradienttimenetelmässä.*

Kaavassa 11 α vastaa oppimisnopeutta, $g(x)$ neuronin aktivointifunktiota, g' on g:n derivaatta, t_j on tavoitteellinen ulostulo, h_j on painotettu summa neuroneiden syötteistä, y_j on todellinen ulostulo ja x_i on i:s syöte. Vaikka deltasääntö on samanlainen kuin perseptronin päivityssääntö, derivaatta on erilainen. Perseptroni käyttää porrasfunktiota (engl. Heaviside step function, unit step function) aktivointifunktiona $g(h)$. Porrasfunktion derivaatta $g'(x)$ ei ole olemassa nollassa ja muilla x:n arvoilla derivaatta on nolla. Derivaatan puuttuminen tekee deltasäännön suorasta soveltamisesta mahdotonta.

Gradienttimenetelmässä ennenaikainen pysäyttäminen (engl. early stopping) mahdollistaa menetelmän koulutusnopeuden tehostumisen. Gradienttimenetelmässä, boostausalgoritmeissa, kuten AdaBoostissa ja MLP:ssä ongelmaksi muodostuu ylisovittaminen (engl. overfitting), joka estää optimaaliseen tulokseen päätymisen käytännön aineistolla (Yao et al. 2007). Boostausalgoritmit, kuten AdaBoost parantavat heikkojen oppijoiden tarkkuutta (Shalev-Shwartz & Ben-David, 2014, s. 134–141). Ennenaikainen pysäyttäminen mahdollistaa koulutus- ja validointiaineiston tarkastelun perusteella kouluttamisen lopettamisen, jotta malli ei ylisovittuisi vain koulutusaineistolle (Yao et al. 2007). Gradienttimenetelmän optimointialgoritmit ovat mukautuvia gradienttien stokastisuuden ja neuroverkkoarkkitehtuurin muutoksille (Li & Malik, 2017).

Eldan ja Shamir tarjoavat funktioita, jotka ovat esitettävissä 3-syvyisillä neuroverkolla ja approksimoitavissa 2-syvyisellä neuroverkolla vain, mikäli 2-syvyisen neuroverkon leveys on eksponentiaalinen (Eldan & Shamir, 2016). Tutkimuksen tulos toimii sekä ReLU-aktivoinnille että Sigmoid-aktivointifunktiolle ja yleisimmin millä tahansa aktivointifunktiolla (Eldan & Shamir, 2016). ReLU-aktivointi on esitetty kaavassa 12:

$$y = \max(0, x)$$

Kaava 12. *ReLU-aktivointifunktio matemaattisesti.*

ReLU ei ole derivoituva nollassa, mikä voi aiheuttaa ongelmia derivaattoihin perustuvilla menetelmillä. Lisäksi ReLU ja sen derivaatta ovat nollia negatiivisilla arvoilla. Oppimisen aikana neuronit saattavat ”kuolla”, mikäli neuronien painoarvot päätyvät nolliksi. Neuronien kuolemisen välttämiseksi ReLUn aktivointifunktiota muutetaan hieman.

$$\mathrm{y} = \begin{cases} \mathrm{ax}, & \mathrm{x} < 0 \\ \mathrm{x}, & \mathrm{x} \geq 0 \end{cases}$$

Kaava 13. *Parametrinen ReLU-aktivointifunktio.*

Kaavassa 13 parametrinen ReLU-aktivointifunktio (PReLU) saadaan vastaamaan vuotavaa ReLU:a (engl. leaky ReLU), kun parametri a asetetaan arvoon 0,01. PReLU:n tapauksessa neuroverkko pystyy itse määrittämään parametrin a arvon. Molemmat PReLU ja vuotava ReLU poistavat ”kuolevan ReLU:n” ongelman, sillä niillä ei ole vakiotermisiä funktion arvoja, kuten ReLU:ssa oleva $\mathrm{y} = 0,\ \mathrm{x} \leq 0$ (Liu, 2017).

Kaavassa 14 oleva yksinapainen Sigmoid-aktivointifunktio on erityisen hyödyllinen neuroverkkojen BP-algoritmeissa (Karlik & Olgac, 2011). Kaavassa 15 oleva kaksinapainen Sigmoid-aktivointifunktio vastaa yksinapaista muuten, mutta kaksinapainen saa arvoväliksi $[-1, 1]$, kun yksinapainen saa arvoja väliltä $(0, 1)$.

$$\mathrm{y} = \frac{1}{1 + e^{-x}}$$

Kaava 14. *Yksinapainen Sigmoid-aktivointifunktio.*

$$\mathrm{y} = \frac{1 - e^{-x}}{1 + e^{-x}}$$

Kaava 15. *Kaksinapainen Sigmoid-aktivointifunktio.*

Sigmoid-funktio on binäärisen porrasfunktion tasoitettu versio. Sigmoid-funktio on hidas muutoksille, eli sen arvo kasvaa todella hitaasti, kun x kasvaa ja sen arvo vähenee todella hitaasti, kun x vähenee (Glorot & Bengio, 2010). Sigmoid-funktion derivaatta on erittäin lähellä nollaa x:n ollessa pieni tai suuri. Mikäli neuroverkkoa opetetaan derivaattoihin perustuvilla menetelmillä, Sigmoid-funktion derivaatan arvosta tulee ongelma. Sigmoid-funktio ei myöskään ole symmetrinen nollan suhteen, mikä on ongelmallista (Sharma, 2017). Glorotin ja Bengion mukaan logistinen Sigmoid-aktivointifunktio on epäsopiva syviin neuroverkkoihin, joissa painoarvot alustetaan satunnaisesti, sillä Sigmoid-funktion keskiarvo voi ajaa varsinkin ylimmän piilokerroksen saturaatioon (Glorot & Bengio, 2010).

Hyperbolinen tangentti on skaalattu Sigmoid-funktio (Sharma, 2017). Hyperbolinen tangentti on epälineaarinen luonnostaan, joten funktion toiminta ei estä kerroksien kasaamista. Lisäksi funktion arvot ovat väliltä $[-1, 1]$. Hyperbolinen tangentti on suosittu, sillä se on symmetrinen nollan suhteen ja sen derivaatta on suurempi kuin Sigmoid-funktiolla nollan läheisyydessä, koska funktio yksinkertaisesti kasvaa nopeammin. Gradientti on kuitenkin erittäin pieni todella pienillä ja suurilla arvoilla, joten sen käyttäminen aktivointifunktiona voi johtaa neuroverkon oppimisen hitauteen. Hyperbolisen tangentin aktivointifunktio on esitetty kaavassa 16.

$$\tanh(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

Kaava 16. *Hyperbolisen tangentin aktivointifunktio.*

Kaavat 17 ja 18 eivät ole yhden taitoksen aktivointifunktioita, vaan Softmax on yleistys logististen funktioiden useille ulottuvuuksille ja Maxout on itsessään koulutettu mallin pohjalta. Softmaxia käytetään moniosaisessa logistisessa regressiossa ja usein neuroverkon viimeisenä aktivointifunktiona neuroverkon ulostulon normalisoimiseksi todennäköisyysjakaumaksi ennustettujen luokkien yli (Bishop, 2006, s. 198). Maxoutin tapauksessa ulostulo on maksimi joukosta syötteitä. Se on suunniteltu sekä helpottamaan optimointia yhdessä satunnaiskarsinnan (engl. dropout) kanssa että parantamaan satunnaiskarsinnan nopean likimääräisen mallin keskimääräistä laskennallista tarkkuutta (Goodfellow et al. 2013). Satunnaiskarsinnalla tarkoitetaan neuroneiden osittaista poistamista verkosta, mikä tapahtuu asettamalla niiden painoarvot nolliksi, eli kuolettamalla ne. Satunnaiskarsinnan tarkoituksena on estää neuroverkkoa ylisovittumasta koulutusaineiston mukaiseksi (Srivastava et al. 2014).

$$f_i(\vec{x}) = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}}, \qquad for\ i = 1, \dots, J$$

Kaava 17. *Softmax-aktivointifunktio.*

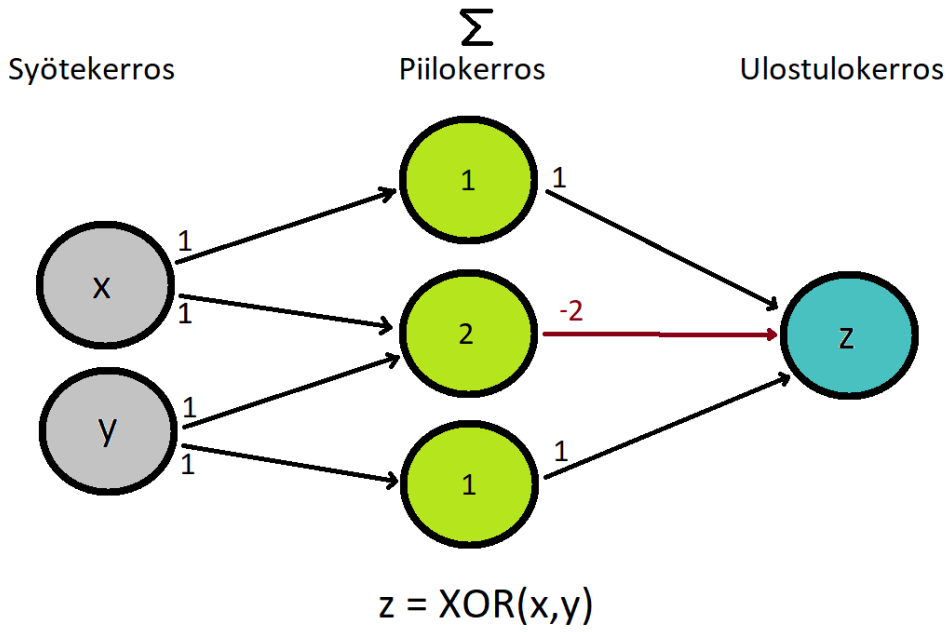
$$f(\vec{x}) = \max_{i} x_i$$

Kaava 18. *Maxout-aktivointifunktio.*

Monikerroksinen perseptroniverkko (MLP, engl. multilayer perceptron) koostuu vähintään kolmesta neuronikerroksesta, syötekerroksesta, vähintään yhdestä piilokerroksesta ja ulostulokerroksesta, jotka ovat yhdistetty toisiinsa eteenpäin kytkeytyvällä tavalla. Jokaisella neuronilla yhdessä kerroksessa on vähintään yksi yhteys seuraavan kerroksen neuroneihin. Mhaskar todisti, että MLP:n ja Sigmoid-aktivointifunktioiden avulla voidaan approksimoida mitä tahansa jatkuvaa

funktiota millä tahansa euklidisen avaruuden (engl. Euclidean space) pienikokoisella osajoukolla (Mhaskar, 1993).

MLP:n etu yksikerroksiseen perseptroniin on sen mukautuvuus monimutkaisempiin funktioihin. Yhtenä yleisenä esimerkkinä on XOR-ongelma, joka on mahdoton ratkaista yksittäisellä neuronilla, mutta yksinkertainen ratkaista MLP:n avulla, kuten kuva 5 osoittaa. Piilokerroksessa oleva luku kertoo laukaisijana toimivan neuronien minimimäärän, eli vain ehto $x = 1 \,\&\, y = 1$ lähettää -2 arvon ulostulokerrokselle. MLP:t ovat universaalisia funktion approksimaatioita, kuten Cybenkon teoreema osoittaa (Cybenko, 1989). Niitä voidaan käyttää matemaattisten mallien luomiseen regressioanalyysillä. Koska luokittelu on erityinen regressiotapa silloin, kun muuttujat on jaoteltu kategorioihin, MLP:t soveltuvat luokittelualgoritmien malleiksi. Lisäksi MLP:t ovat hyödyllisiä tutkimuksessa, sillä ne kykenevät ratkaisemaan ongelmat stokastisesti, mikä mahdollistaa usein hyvin approksimoidut ratkaisut erittäin monimutkaisiin ongelmiin.



Kuva 5. *XOR-ongelman ratkaisu MLP:n avulla. Painoarvojen ja piilokerroksen neuronien aktivoitumisien avulla määräytyy z:n arvoksi 0 tai 1.*

Monikerroksisten perseptroniverkkojen kultainen aikakausi oli 1980–1990-luvulla, jolloin niiden avulla tehtiin sovelluksia monille aloille, kuten puheentunnistukseen, kuvantunnistukseen ja konekääntämiseen (Wasserman & Schwartz, 1988). Tämän jälkeen MLP:iden suosio laski, sillä ne joutuivat kilpailemaan paljon yksinkertaisempien tukivektorikoneiden (SVM, engl. support vector machine) kanssa (Collobert & Bengio, 2004). Kiinnostus BP-algoritmeja kohtaan palasi 2010-luvulla syväoppimisen potentiaalin vuoksi.

Vahvistusoppimista ja BP-algoritmeja voitiin yhdistää Williamsin toimesta jo 1980-luvun lopulla, jolloin neuroverkkojen approksimointikyky parani vahvistusoppimisen alueella. Williams suunnitteli vahvistusoppimista hyödyntäville algoritmeille neuronin kaltaisia yksiköitä, joilla oli jatkuva eikä binäärinen ulostulo (Williams, 1992). Williams suunnitteli REINFORCE-algoritmeja, joiden perusideana on painoarvojen säätäminen vahvistuskertoimen r saamisen jälkeen ja jokaisen testikierroksen jälkeen säädetään painoarvoa w_{ij} kaavan 19 mukaisesti.

$$\Delta \mathrm{w_{ij}} = \alpha_{ij}(r - b_{ij})e_{ij}$$

Kaava 19. *Williamsin (1992) REINFORCE-algoritmien painoarvojen muutoskaava.*

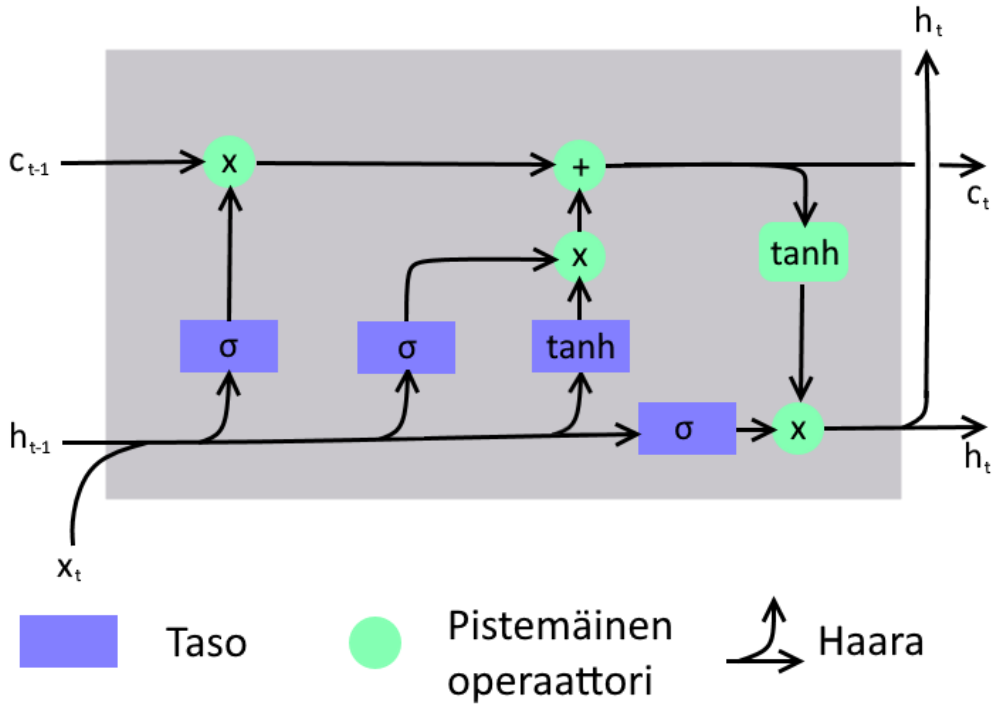
Kaavassa 19 α_{ij} on oppimisnopeuden kerroin, b_{ij} on vahvistuksen pohjataso ja e_{ij} on w_{ij}:n ominaiskelpoisuutta (engl. charasteristic eligibility) kuvaava termi. Ominaiskelpoisuus w_{ij} lasketaan kaavalla 20:

$$e_{ij} = \partial \ln g_i / \partial w_{ij}$$

Kaava 20. *Painoarvon w_{ij} ominaiskelpoisuuden kaava (Williams, 1992).*

Lisäksi oletetaan, että vahvistuksen pohjataso b_{ij} on ehdollisesti riippumaton y_i:stä, kun otetaan huomioon, että W ja x^i sekä oppimisnopeuden kerroin α_{ij} eivät ole negatiivisia ja riippuvat korkeintaan w^i:stä ja t:stä (Williams, 1992). Kaikkia vastaavan muodon omaavia oppimisalgoritmeja kutsutaan REINFORCE-algoritmeiksi. Nimi on lyhenne sanoista "REward Increment = Non-negative Factor x Offset Reinforcement x Characteristic Eligibility", joka kuvaa algoritmin muotoa kaavana (Williams, 1992).

Pitkäkestoinen lähimuisti (LSTM, engl. Long short-term memory) esiteltiin vuonna 1997. LSTM:n tarkoituksena on varastoida informaatiota leikkaamalla gradientti silloin, kun siitä ei ole haittaa (Hochreiter & Schmidhuber, 1997). LSTM voi oppia ylittämään minimaaliset viiveet, jotka ylittävät 1000 diskreettiaikavaihetta toteuttamalla jatkuvan virhevirran jatkuvien virhearvojen läpi erikoisyksiköitä käyttäen (Hochreiter & Schmidhuber, 1997). LSTM ratkaisi ongelman, jossa BP-algoritmeja pyrittiin käyttämään jatkuva-aikaisesti. BP-algoritmin käyttäminen jatkuva-aikaisesti aiheutti painoarvojen värähtelyn. Tästä syystä koulutus joko kesti erittäin kauan tai ei toiminut. LSTM esitettiin takaisinkytketyssä neuroverkossa, joka mahdollisti verkon arkkitehtuuriin perehtymisen (Hochreiter & Schmidhuber, 1997).



Kuva 6. *Pitkäkestoinen lähimuisti (LSTM). LSTM:ssä edellisen solun tilasta ja ulostulosta sekä nykyisen solun syötteestä lasketaan nykyiselle solulle tila ja ulostulo.*

Kuvassa 6 visualisoidaan LSTM:n yhtä solua, jossa syötteelle x_t lasketaan ulostulo h_t ja solun tila c_t. Solun syötteinä toimivat lisäksi edellisen solun ulostulo h_{t-1} ja edellisen solun tila c_{t-1}. Laskennassa käytetään tasoja, kuten Sigmoid-kerrosta tai hyperbolista tangenttia ja pistemäisiä operaattoreita, kuten kertolaskua tai yhteenlaskua. Sigmoid-kerros antaa lukuarvoja väliltä (0, 1) määrittäen kuinka paljon komponentit x_t ja h_{t-1} vaikuttavat solun ulostuloon tai tilaan. Hyperbolinen tangentti palauttaa lukuarvoksi luvun väliltä (-1, 1).

Neuroverkkojen tulokset paranivat, kun verkkojen koko kasvoi. 1990-luvun lopussa Bartlett todisti, että painoarvojen koolla on suurempi merkitys kuin verkon koolla (Bartlett, 1997). Neuroverkkojen osalta ei voida varmentaa, että ne generalisoituisivat hyvin. Jotta generalisointia voitaisiin kontrolloida, statistisen oppimisteorian (engl. statistical learning theory) pohjalta pitäisi pystyä kontrolloimaan kahta tekijää: empiirisen riskin määrää ja luottamusväliä (engl. confidence

interval) (Vapnik, 2013). Neuroverkkojen tapauksessa ei pystytä kontrolloimaan kumpaakaan. Empiirisen riskin minimointia varten neuroverkossa pitäisi minimoida funktionaalisuus, jossa on monta paikallista minimiä – Oppimisteorian pohjalta ei ole olemassa rakenteellista tapaa, jolla estettäisiin hyväksymättömien paikallisten minimien ilmaantuminen (Vapnik, 2013).

Bartlett ja Mendelson esittelivät datasta riippuvia riskirajoja neuroverkoille, päätöspuille (engl. decision tree) ja tukivektorikoneille (Bartlett & Mendelson, 2002). Kuvioiden luokittelussa ja regressiossa hyvien virherajojen löytäminen on ollut aikaa vievää. Yleisesti virherajat ovat kahden tekijän summa: otospohjainen estimaatti suorituskyvystä ja rangaistustermi, joka on suuri monimutkaisemmilla malleilla (Bartlett & Mendelson, 2002).

Neuroverkkojen optimointiin tärkeitä ominaisuuksia ovat neuronien osittainen poistaminen eli satunnaiskarsinta (Srivastava et al. 2014), ennenaikainen pysäyttäminen (Yao et al. 2007) ja muut hyperparametrit (Bergsta et al. 2011). Hyperparametreja ovat muuttujat, jotka määrittävät verkon rakenteen ja muuttujat, jotka määrittelevät miten verkko on koulutettu. Ne asetetaan ennen opetuksen aloittamista. Verkon rakenteeseen vaikuttavia hyperparametreja ovat satunnaiskarsinnan prosenttiosuuden lisäksi painoarvojen alustaminen, käytetyt aktivointifunktiot ja piilotettujen neuronien määrä. Oppimisalgoritmiin vaikuttavia hyperparametreja ovat muun muassa oppimisnopeus, eräkoko (engl. batch size), koulutuskausien (engl. epoch) määrä sekä momentti eli liikemäärä gradienttimenetelmän optimointialgoritmissa.

Etenkin syvien neuroverkkojen (DNN, engl. deep neural network) tapauksessa hyperparametrien optimointi on erityisen tärkeää. Hyperparametrien optimoinnissa satunnaishaut (engl. random search) eivät toimineet syväoppimisessa, vaan sekä Gaussin prosessi (engl. Gaussian Process) että puurakenteinen Parzen estimaattori (engl. tree-structured Parzen estimator) suoriutuivat huomattavasti paremmin (Bergsta et al. 2011). DNN:t ja RNN:t ovat tehokkaita malleja, mutta niiden kouluttaminen stokastisella gradienttimenetelmällä momentin kanssa oli mahdottomuus ennen hyvin suunniteltua satunnaisalustajaa ja hitaasti kasvavaa aikataulutusta momentille (Sutskever et al. 2013). Vuonna 2014 RNN:lle tehtiin regularisointitekniikka, jossa käytettiin LSTM:ä (Zaremba et al. 2014). Tutkimuksessa tulee esiin, että satunnaiskarsinta ei sovellu suoraan ylisovittamisen estämiseksi RNN:lle ja LSTM:lle (Zaremba et al. 2014). Regularisoinnilla tarkoitetaan ylisovittamisen estämistä verkon hyperparametrien avulla, kuten pienentämällä kertoimia (engl. coefficient) kohti nollaa. Regularisointi siis mahdollistaa joustavampien mallien kehittämistä monimutkaisempiin ongelmiin ilman pelkoa ylisovittamisesta.

Neuroverkkojen kontrolloitavuuteen ja paikallisten minimien aiheuttamaan epäluotettavuuteen (Vapnik, 2013) löytyy potentiaalisia ratkaisuja, joissa selvitetään ongelman lähdettä. Suurikokoisille eriytetyille eteenpäin kytketyille verkoille satunnaishäviöfunktion matalimmat kriittiset arvot muodostavat kerrostetun rakenteen ja ne sijaitsevat hyvin määritellyssä kaistassa, jota globaali minimi rajoittaa alhaalta päin (Chromanska et al. 2015). Tämän kaistan ulkopuolella olevien paikallisten minimien lukumäärä vähenee eksponentiaalisesti verkon koon mukaan (Chromanska et al. 2015). Vaikka paikalliset minimit aiheuttavat epäluotettavuutta neuroverkkojen toiminnassa tilastotieteellisestä näkökulmasta, globaali minimi ei lähes koskaan ole optimaalisin ratkaisu käytännössä. Gradientin globaali minimi ohjaa tehokkaimmin kohti opetusdatan optimaalista ratkaisua. Globaali minimi johtaa usein ylisovittamiseen (Chromanska et al. 2015).

Luottamusvälin laskenta on mahdotonta neuroverkkojen tapauksessa (Vapnik, 2013). Koulutusvirhe ei vastaa käytännössä validointivirhettä tai todellista virhettä. Syvät neuroverkot sovittavat satunnaisesti arvotut nimet tietoaineistoon eli datasettiin siten, että kouluttaminen onnistuu koulutusvirheen ollessa 0 (Zhang et al. 2016). Tähän johtavia syitä on muun muassa se, että neuroverkon tehokas kapasiteetti on tarpeeksi suuri muistamaan koko datasetin (Zhang et al. 2016). Neuroverkon pystyy myös kouluttamaan korvaamalla kuvan täysin satunnaisilla pikseleillä eli Gaussilaisella kohinalla (engl. Gaussian noise) siten, että koulutuksen virhe on 0 (Zhang et al. 2016). Riittävän suuri neuroverkko on kykenevä muistamaan koko opetusaineiston, jolloin neuroverkko ei opi haluttuja ominaisuuksia, vaan käytännössä opettelee ulkoa aineiston. Tällöin koulutuksen virhe on 0, mutta testauksen virhe on usein liian suuri käytännön sovelluksiin. Ratkaisuna regularisoinnin osalta on käytössä muun muassa validointidata, jonka virhearvio toimii koulutusvaiheessa estimaattina testausdatan virheelle.

Syvien neuroverkkojen kouluttamisessa yksi ongelmista on kerrosten syötteiden muuttuminen koulutusvaiheen aikana, sillä myös edellisten kerrosten parametrit ja syöte muuttuvat. Ongelma hidastaa kouluttamista vaatimalla matalampia oppimisnopeuksia ja huolellista hyperparametrien alustamista samalla tehden epälineaaristen mallien kouluttamisesta vaikeaa (Ioffe & Szegedy, 2015). Erän normalisointi (engl. batch normalization) mahdollistaa korkeamman oppimisnopeuden käyttämisen ja hyperparametrien alustamisen suhteen ei tarvitse olla äärimmäisen tarkka – samalla regularisoiden verkon, joka mahdollistaa tarkkuuden, jossa jopa satunnaiskarsinta voidaan poistaa (Ioffe & Szegedy, 2015). Erän normalisointi käyttää summattujen syötteiden jakaumaa pienerällä (engl. mini batch), jotta keskiarvo ja varianssi voidaan laskea. Näitä käytetään normalisoimaan summattu syöte jokaisessa koulutustapauksessa. Erän normalisointi on riippuvainen eräkoosta, joten sen soveltaminen RNN:ään ei ole yksinkertaista (Ba et al. 2016). Kerrosten normalisoinnissa (engl. layer normalization) lasketaan keskiarvo ja varianssi kaikista summatuista syötteistä. Jokaiselle neuronille annetaan mukautuva vakiotermi ja vahvuus, jotka lasketaan normalisoinnin jälkeen, mutta ennen epälineaarisuutta (Ba et al. 2016). Toisin kuin erän normalisoinnissa, kerrosten normalisoinnissa tehdään täsmälleen sama laskenta koulutuksessa ja testauksessa (Ba et al. 2016).

Neuroverkkoja voi soveltaa hyvin laajalti moniin eri ongelmiin, kuten kuvantunnistukseen, puheentunnistukseen tai konekääntämiseen (Wasserman & Schwartz, 1988). Neuroverkkojen käyttäminen kuvantunnistamisessa on pakottanut tutkijoita kehittämään innovatiivisia menetelmiä arkkitehtuurin suunnitteluun. Kasvojen havainnointi harmaasävykuvista (engl. grayscale image) perustui 20x20-kokoisiin otoksiin, valaistuksen sovittamiseen ja histogrammin tasaukseen (engl. histogram equalization) potentiaalisista havainnoista (Rowley et al. 1996). Käsin kirjoitettujen symbolien tunnistuksessa konvoluutionaaliset neuroverkot dominoivat yksinkertaisempiin menetelmiin nähden tarkkuuden osalta (LeCun et al. 1998). Objektintunnistukseen Lowe ehdotti likimääräisen LoG:in (engl. Laplacian of Gaussians) laskemista käyttämällä DoG-suodatinta (engl. Difference of Gaussians), joka keskittyy ominaisuuksien parantamiseen (Lowe, 1999). 2010-luvulla kuvantunnistuksen huipulla on siirrytty yhä suurempiin ja syvempiin konvoluutionaalisiin neuroverkkoihin. Kuvantunnistuksen keulakuvana tunnetussa ImageNetin datasetissä on satoja kategorioita, tuhansia luokkia ja miljoonia kuvia (Russakovsky et al. 2015). ImageNetin objektien luokitteluhaasteessa (ILSVRC) tuhatta luokkaa tunnistava malli sisälsi 60 miljoonaa parametria ja tunnisti yli 62 prosenttia kuvista oikein (Krizhevsky et al. 2012).

Puheentunnistuksessa neuroverkot helpottavat tunnistamaan sekä sanoja äänteiden ja tavujen avulla, että sanaketjuja. Sanaketjujen ja sanojen merkityksien ymmärtäminen on osa luonnollisen kielen käsittelyä (NLP, engl. Natural language processing). Luonnollisen kielen käsittely on 2000-luvulla ollut tärkeä osa koneoppimista ja yksi neuroverkkojen yleisimmistä käyttökohteista. Bengion työryhmän neuroverkkoja hyödyntävä todennäköisyyksiin perustuva malli pohjautuu sanojen peräkkäisyyksiin ja samankaltaisten sanojen tunnistamiseen (Bengio et al. 2003). Tutkimuksessa neuroverkko suoriutui 20–35 % paremmin kuin tasoitettu trigrammi (Bengio et al. 2003). Trigrammi on N-grammin (engl. N-gram) muoto, jossa kolmea peräkkäistä sanaa hyödyntäen rakennetaan todennäköisyyksiin perustuva kielimalli.

Useimmat puheentunnistusta hyödyntävät järjestelmät käyttävät piilotettuja Markovin malleja (HMM, engl. hidden Markov model) käsitellessään puheen ajallista vaihtelua (Hinton et al. 2012). Lisäksi järjestelmät käyttävät Gaussin sekoitemallia (engl. Gaussian mixture model) määritelläkseen kuinka hyvin jokaisen HMM mallin jokainen tila sopii koeffisienttikehykseen, joka edustaa akustista syötettä (Hinton et al. 2012). DNN:t suoriutuvat paremmin kuin GMM:ään perustuvat menetelmät puheentunnistuksessa (Hinton et al. 2012). Puheentunnistusta hyödyntävän funktionaalisuuden liittäminen NLP-ohjelmistoon mahdollistaa tekoälyjen kommunikoinnin ihmisen kanssa.

Konekääntämisessä monet menetelmät perustuvat enkooderi-dekooderi -arkkitehtuuriin (engl. encoder-decoder-architecture). Esimerkiksi Googlella kehitetty Transformer on arkkitehtuuri, jossa enkooderi alustaa sisääntulosekvenssin $(x_1 \dots x_n)$ sekvenssiksi jatkuvista esityksistä $(z_1 \dots z_n)$ ja dekooderi generoi ulostulosekvenssin $(y_1 \dots y_m)$ symboleista regressiivisesti yksi elementti kerrallaan (Vaswani et al. 2017). Usein syöte asetetaan määrätyn kokoiseksi vektoriksi, jolle dekooderi muodostaa käännöksen. Määrätyn kokoinen vektori on pullonkaula, joka estää arkkitehtuurin tarkkuuden kehityksen (Bahdanau et al. 2014). Bahdanaun tutkimusryhmä mah-

dollisti mallille oleellisten sanojen etsimisen syötteestä (Bahdanau et al. 2014). Chon tutkimusryhmä keskittyi käyttämään kahta RNN:ää, jotka ovat yhdessä koulutettuja (Cho et al. 2014). Sutskeverin tutkimusryhmä keskittyi monikerroksisen LSTM:n tekemiseen, jotta syötteitä pystyi säätämään erikokoisille vektoreille (Sutskever et al. 2014). Kalchbrennerin tutkimusryhmä käytti dynaamista konvoluutionaalista neuroverkkoa (DCNN, engl. dynamic convolutional neural network) ja dynaamista maksimien yhdistämistä (engl. max pooling), jotta verkko pystyy käsittelemään eri pituisia syötteitä mahdollisimman hyvin (Kalchbrenner et al. 2014).

2.3 Konvoluutionaaliset neuroverkot

Konvoluutionaaliset neuroverkot ovat neuroverkkoja, joissa on vähintään yksi konvoluutionaalinen kerros, sekä lähes kaikissa CNN:issä yhdistämiskerroksia (engl. pooling layer) on vähintään yksi (Goodfellow et al. 2016, s. 345). CNN:t ovat regularisoituja versioita monikerroksisista perseptroniverkoista (MLP). Konvoluutionaalisten kerrosten tarkoituksena on löytää ominaisuuksia, jotta syötteet voidaan yhdistää mahdollisimman hyvin vastaamaan haluttua ulostuloa. Regularisoinnilla pyritään estämään ylisovittamista eli verkon mukautumista vain koulutusaineistoon. Tyypillisiä regularisointitapoja ovat esimerkiksi painoarvojen suuruuksien rajoittaminen tappiofunktion avulla, verkosta solmujen poistaminen sekä ennenaikainen pysäyttäminen. CNN:t lähestyvät regularisointia hyödyntämällä datan hierarkkisia kuvioita ja kokoamalla monimutkaisempia malleja käyttämällä pienempiä ja yksinkertaisempia malleja. Kuvantunnistuksessa neuroverkot huomioivat ominaisuuksien sijainnin vaihtelevuuden syötekuvissa. Tämän vuoksi se on yksi parhaista esimerkeistä, miten alan asiantuntemus voidaan sisällyttää tehokkaasti verkon arkkitehtuuriin (Goodfellow et al. 2016, s. 269).

Yksi CNN:n tärkeistä konsepteista on yhdistämiskerros, jonka tarkoituksena on pienentää datan kokoa epälineaarisesti. Vuonna 1992 kehitettiin menetelmä maksimien yhdistäminen, joka toi suvaitsevaisuutta pieniin siirtymiin ja muodonmuutoksiin 3D-objektien tunnistamisen helpottamiseksi. Maksimien yhdistäminen on yhdistämiskerroksista käytetyin (Krizhevsky et al. 2012). Menetelmässä yhdistämiskerroksen syöte jaetaan suotimen kokoisiin ei-päällekkäisiin osiin, joista jokaisesta valitaan maksimi kerroksen ulostuloksi (Ciresan et al. 2011). Täten datan koko pienenee suotimen koon mukaan eli esimerkiksi 2x2-suodinta käytettäessä datan koko pienenee neljännekseen. Kuvantunnistuksessa maksimien yhdistämisen osoitettiin toimivan paremmin kuin muiden yhdistämisessä käytettyjen vaihtoehtojen (Scherer et al. 2010).

Konvoluutionaalisten neuroverkkojen tapauksessa opetusaineistoa tarvitaan paljon, kuten neuroverkkojen tapauksessa aina. Datasettien koot ovat kasvaneet paljon (Goodfellow et al. 2016, s. 36). 1990-luvun lopulla käsin kirjoitettujen numeroiden mustavalkoinen datasetti MNIST (engl. Modified National Institute of Standards and Technology database) koostui kymmenistä tuhansista esimerkeistä (LeCun et al. 1998). 2000-luvun alussa monimutkaisempia datasettejä tehtiin julkisiksi, kuten CIFAR-10 (engl. Canadian Institute For Advanced Research) datasetti, joka koostui kymmenestä eri objektiluokasta värikuvia (Krizhevsky, 2009). 2010-luvulla luotiin ImageNet tietokanta, joka koostuu sadoista tuhansista kuvista yli tuhannelle eri luokalle (Russakovsky et al. 2015). 2010-luvulla tietokoneiden laskennallinen teho kasvoi riittävän suureksi, jotta syviä konvoluutionaalisia neuroverkkoja voitiin kouluttaa. Graafisten prosessointiyksiköiden (GPU, engl. graphical processing unit) käyttö mahdollisti moninkertaisesti nopeamman laskennan verrattuna keskussuorittimeen (CPU, engl. central processing unit). Yhtenä suurena harppauksena kuvantunnistuksessa voidaan pitää Krizhevskyn tutkimusryhmän työtä (Krizhevsky et al. 2012), jossa ImageNet:in tapauksessa tunnistettiin 61,9 % suurimmalla todennäköisyydellä ja 83,6 % viiden parhaan todennäköisyyksillä.

Konvoluutionaalisten neuroverkkojen optimointi tapahtuu pitkälti samalla tavalla kuin MLP:n tapauksessa, eli satunnaiskarsinta (Srivastava et al. 2014), ennenaikainen pysäyttäminen (Yao et al. 2007), eräkoon normalisointi (Ioffe & Szegedy, 2015) sekä muut hyperparametrit ovat tärkeässä roolissa. Hyperparametrien hienosäädöllä on mahdollista parantaa verkon suorituskykyä (He et al. 2019). Datan esikäsittelyssä on sovellettu muun muassa poistomenetelmää (engl. cutout) ja kohinan lisäämistä (engl. noisification). Poistomenetelmässä valitaan kuvasta tietynkokoinen suorakulmainen alue, joka täytetään yhdellä värillä, jotta neuroverkko oppisi tunnistamaan esineen vain sen osasta (DeVries & Taylor, 2017). Kohinan lisäämisen tarkoituksena on hämärtää muotojen selvyyttä ja parantaa verkon sietokykyä erilaatuisille kuville.

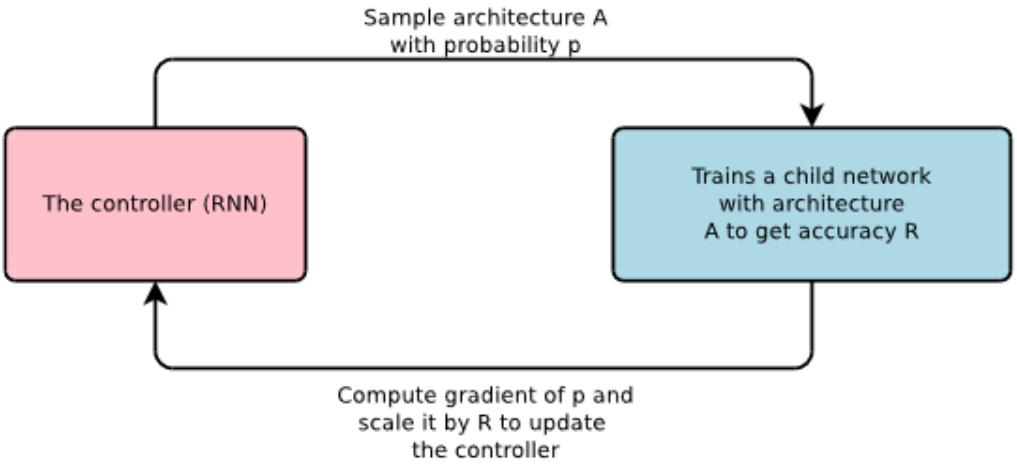
Jotta esimerkiksi gradienttien katoamista voidaan estää ja hidasta koulutusaikaa voidaan nopeuttaa, tulee CNN:n rakenteeseen tehdä optimointeja. Stokastinen syvyys (engl. stochastic depth) on menetelmä, jonka tarkoituksena on käyttää lyhyitä verkkoja koulutuksessa ja syviä verkkoja testaamisessa (Huang et al. 2016). Stokastinen syvyys toimii siten, että koulutuksen alussa verkko on syvä, mutta koulutuksen aikana jokaiselle erälle poistetaan osajoukko kerroksia ohittamalla ne identiteettifunktiolla (engl. identity function) ja skaalaamalla syöte oikean kokoiseksi (Huang et al. 2016).

Aluepohjaiset konvoluutionaaliset neuroverkot (R-CNN, engl. Region-based convolutional neural network) sopivat objektintunnistukseen. R-CNN:t ovat tarkempia kuin perinteiset menetelmät ja ”Nopea R-CNN” on 9 kertaa nopeampi koulutusvaiheessa ja 213 kertaa nopeampi testivaiheessa kuin perinteinen R-CNN (Girshick, 2015). R-CNN:n haittoja ovat koulutuksen monivaiheisuus ja hitaus sekä objektintunnistuksen kuluttama aika, joka on lähes minuutin GPU:lla (Girshick, 2015). ”Nopeampi R-CNN” koostuu CNN:stä, alueellisesta ehdotusverkosta (RPN, engl. Regional proposal network) ja mielenkiintoalueista (ROI, engl. Region of interest). Oletuksena CNN:nä toimii ResNet50, joka perustuu ResNet (engl. Residual Network) -arkkitehtuuriin. ”Nopeampi R-CNN” pystyy havaitsemaan useita objekteja syötekuvasta tehokkaasti, jopa 5–17 kuvaa sekunnissa (FPS, engl. frames per second) GPU:lla (Ren et al. 2015). Aluepohjaiset konvoluutionaaliset verkot vaativat tehokkaan GPU:n ja paljon muistia, jotta koulutus menisi sujuvasti.

Kasvojentunnistus (engl. face recognition) on yksi konvoluutionaalisten neuroverkkojen sovelluskohde. Lawrencen työryhmän ohjelmakokonaisuus yhdistää paikallisen näytteenoton (engl. local image sampling), itseorganisoituviin karttoihin (SOM, engl. self-organizing map) pohjautuvan neuroverkon ja konvoluutionaalisen neuroverkon (Lawrence et al. 1997). Kasvojentunnistuksessa konvoluutionaalisten neuroverkkojen hyöty on se, että ne pystyvät tunnistamaan kasvojen eri osia, vertaamaan opetettujen henkilöiden piirteitä syötekuvan piirteisiin ja muodostamaan todennäköisyyksiin perustuvan johtopäätöksen vastaako syötekuvan henkilö ketään opetettua henkilöä.

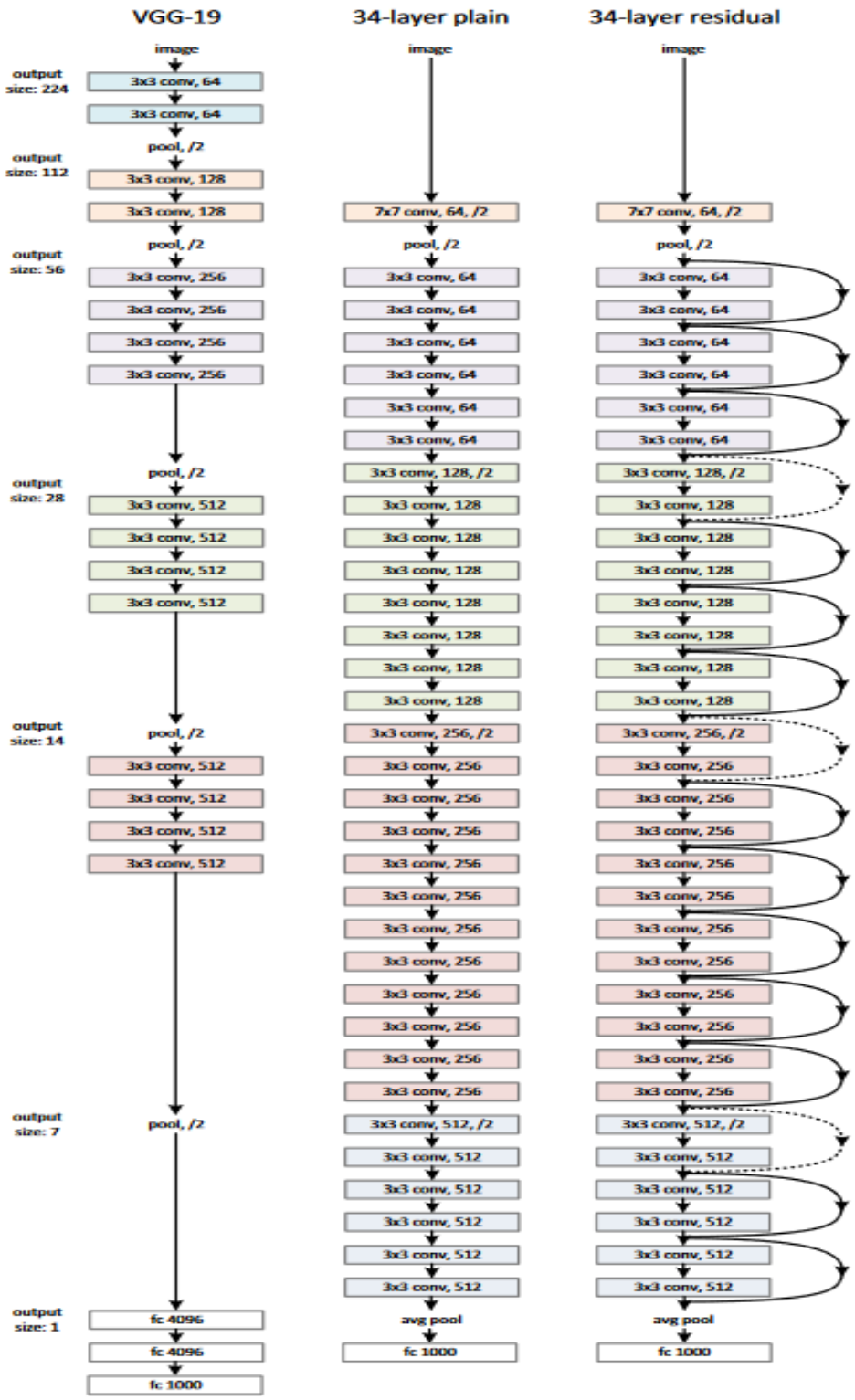
Koulutettujen neuroverkkojen soveltuvuutta on tutkittu monissa eri alojen käyttökohteissa. Neuroverkkojen soveltamiseen on käytetty muun muassa syvien konvoluutionaalisten aktivointiominaisuuksien (DeCAF, engl. Deep convolutional activation feature) tarkastelua ja visualisointia (Donahue et al. 2014). DeCAF on avoimen lähdekoodin toteutus, joka visualisoi verkkojen aktivointeja eri tunnistukseen liittyvissä ongelmissa. Donahuen työryhmän tutkimuksesta huomataan, että eri tarkoituksiin koulutetuilla neuroverkoilla on hyvin erilaiset aktivointiominaisuudet. Konvoluutionaalisten neuroverkkojen yhtenä huonona ominaisuutena on koulutuksen päteminen ainoastaan tiettyyn tehtävään, jolloin neuroverkkoa ei voida käyttää luotettavasti sellaisessa tehtävässä, johon sitä ei ole koulutettu.

Neuroverkkojen arkkitehtuuriin ja verkkojen koon pienentämiseen on keskitytty tuoreissa tutkimuksissa. Esimerkiksi geneettisillä verkoilla (GeNet, engl. Genetic network) ja muilla innovaatioilla verkon rakenteen suhteen pyritään pienentämään verkon kokoa ja säilyttämään mahdollisuus kehittää vieläkin tarkempi verkko. GeNetin tapauksessa todennäköisimmän vaihtoehdon virheprosentti oli ILSVRC:ssä 27,87, mutta kooltaan se on pienempi kuin monet kilpailijansa (Xie & Yuille, 2017). NASNet-arkkitehtuurissa etsitään arkkitehtuurinen rakennuspalikka pienestä datasetistä ja otetaan se käyttöön suuremmalle datasetille (Zoph et al. 2018). NASNet perustuu neuronien arkkitehtuuriseen hakuun (NAS, engl. Neural Architecture Search), joka käyttää vahvistusoppimista (Zoph et al. 2018). NAS perustuu takaisinkytkettyyn neuroverkkoon ja vahvistusoppimiseen. Takaisinkytketyn neuroverkon avulla generoidaan mallin kuvaukset ja vahvistusoppimisessa validointisetin avulla maksimoidaan generoitujen arkkitehtuurien odotettu tarkkuus (Zoph & Le, 2016). NASNet-arkkitehtuuri noudattaa Williamsin (1992) REINFORCE-algoritmia, sillä palkintosignaali ei ole differentioituva (Zoph & Le, 2016). NASNetin yleinen toimintamalli esitetään kuvassa 7.



Kuva 7. *NASNetin yleiskuva (Zoph & Le, 2016, s. 1). NASNetissä suurempi RNN koulutetaan pienemmistä verkoista saaduilla arkkitehtuureilla* A.

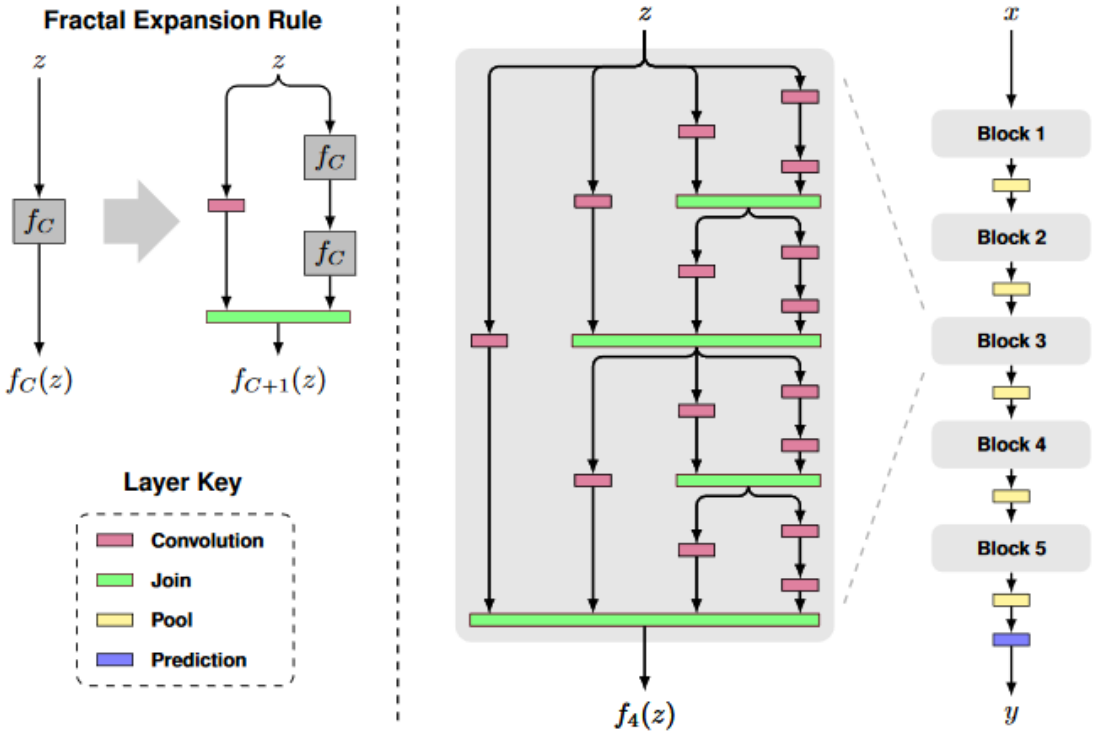
ResNetit johtivat dramaattiseen kasvuun neuroverkkojen syvyyden ja tarkkuuden osalta, jolloin tulokset nousivat monimutkaisissa ongelmissa omaan luokkaansa verrattuna muihin menetelmiin. ResNetit ovat konvoluutionaalisia neuroverkkoja, joiden toimintaa kuvastaa verkon rajoittaminen oppimaan edellisten kerrosten jäännöksistä (He et al. 2016). ResNetin tapa ohittaa rinnakkaiskerroksessa konvoluutionaalisia kerroksia kehittää verkon ominaisuuksien tunnistusta laajemmin. ResNet käyttää stokastista syvyyttä, jossa osa konvoluutionaalisista kerroksista ohitetaan ja korvataan identiteettifunktiolla (Huang et al. 2016). ResNetin suorituskyky verrattuna VGG-19 (engl. Visual Geometry Group) malliin (Simonyan & Zisserman, 2014) ImageNetin datasetin ILSVRC:n osalta on selkeä, sillä ResNet-152:lla virheprosentti oli 19,38 %, kun VGG:llä virheprosentti oli 24,40 %. ResNet on kehitettävissä paremmaksi, mikäli siihen sovelletaan parannuksia, kuten väriavaruuden säätöä ja rakenteellisia muutoksia (He et al. 2019). Kuvassa 8 vertaillaan VGG:tä, yleismallia ja ResNetin rakenteita.



Kuva 8. *Vertailu VGG-19 mallin, yleismallin ja ResNetin rakenteiden välillä (He et al. 2016, s. 4). Malleissa konvoluutiokerrokset ja täysin yhdistetyt kerrokset esitetään laatikoina ja siirtymiä kuvataan nuolilla.*

DenseNet-arkkitehtuurissa tavoitteena on säilyttää verkon kerrosten määrä pienenä, sillä kerrosten välillä on paljon yhteyksiä. DenseNet pohjautuu eteenpäin suuntautuvaan neuroverkkoon, jossa konvoluutionaaliset kerrokset ovat yhdistetty eteenpäin kaikkiin muihin konvoluutionaalisiin kerroksiin (Huang et al. 2017). DenseNet hyödyntää ResNetin hyperparametrien optimointia ja skaalautuu syvempiin verkkoihin ilman ylisovittamista (Huang et al. 2017).

FractalNetin tapauksessa huomattavasti pienemmällä verkon koolla saadaan yhtä hyviä tuloksia kuin ResNetillä (Larsson et al. 2016). FractalNet koostuu konvoluutiokerrosten osittaisesta jakamisesta pieniin osiin ja yhdistämiskerroksista. Täten koko malli koostuu vähintään parista yhdistämiskerroksesta ja useasta konvoluutionaalisesta kerroksesta, jotka ovat rinnakkain kuvan 9 mukaisesti.



Kuva 9. *FractalNetin arkkitehtuuri (Larsson et al. 2016, s. 2). Verkon rakenne z koostuu rinnakkaisista konvoluutiokerroksista sekä rinnakkaisten ja peräkkäisten konvoluutiokerrosten yhdistämisistä. Syöte x lasketaan vuorotellen rinnakkaisrakenteen z ja yhdistämiskerroksen avulla, kunnes siitä saadaan ennuste y.*

GoogLeNetin tapauksessa (Szegedy et al. 2015) vuoden 2014 ILSVRC-haaste antoi viiden todennäköisimmän virheeksi 6,67 %. Työryhmän mukaan pelkästään verkon koko ei vaikuta tulosparannukseen, vaan siihen vaikuttaa myös heidän kehittämänsä Inception-moduuli, joka koostuu rinnakkaisista erikokoisista konvoluutiokerroksista, liitoskerroksista ja maksimien yhdistämiskerroksista (Szegedy et al. 2015).

Konvoluutionaaliset neuroverkot vastaavat pitkälti MLP-arkkitehtuuria, jonka ensimmäiset kerrokset sisältävät usein konvoluutionaalisia kerroksia sekä yhdistämiskerroksia. Konvoluutionaaliset neuroverkot ovat moniulotteiselle datalle hyvin soveltuvia, mikäli datasta halutaan löytää yhteneviä kuvioita, kuten erinäisiä objekteja. CNN:ien tehokkuus kuvantunnistuksessa johtuu pääosin ominaisuuksien löytämisestä sijainnista riippumatta. CNN:ien huonoja puolia ovat gradienttien mahdollinen katoaminen, eteenpäin suuntaavan virran väheneminen ja koulutusaikojen tuskallinen hitaus (Huang et al. 2016). CNN:n, kuten yksinkertaisempien neuroverkkojen, tapauksessa pitää olla huolellinen, ettei ylisovittamista tapahdu opetusdatan suhteen.

3. KUVANKÄSITTELY

Digitaalinen kuvankäsittely (engl. digital image processing) eli tässä työssä kuvankäsittely on signaalinkäsittelyn (engl. digital signal processing) alakategoria. Kuvankäsittelyssä käytetään tietokonetta, jotta digitaalisessa muodossa olevia kuvia voidaan prosessoida algoritmin avulla (Gonzales & Woods, 2007, s. 23–24). Digitaalisten kuvien yleinen esitystapa on bittikartta, joka koostuu pikseliriveistä (Patin, 2003, s. 5). Pikseli voi koostua yhdestä tai useammasta värikanavasta, joka määrittää pikselin värin. Yleisesti kuvankäsittelyyn kuuluvat kuvan laadun parantaminen (engl. image enhancement), kunnostaminen (engl. image restoration), koodaus (engl. image encoding) ja pakkaus (engl. image compression).

Digitaalisten kuvien alkuhetki oli 1920-luvulla, jolloin Lontoon ja New Yorkin välillä lähetettiin kaapelia pitkin kuva, joka koostettiin lennättimen kaltaisella laitteella (Gonzales & Woods, 2007, s. 25–26). 1960-luvulla kuvankäsittelyn perimmäisenä tarkoituksena oli parantaa kuvan laatua. Nykyään kuvankäsittelyllä on lukuisia käyttötarkoituksia, kuten kuvan laadun parantaminen, objektien tunnistaminen ja kuvanmuokkaus. Ne auttavat teknologiaa kehittymään tarkemmaksi ja tehokkaammaksi lukuisilla alueilla, kuten lääketieteessä, autoteollisuudessa ja koneoppimisessa.

Tietotekniikan teknisen puolen innovaatioiden avulla mahdollistettiin kuvankäsittely. Näitä innovaatioita olivat erityisesti: transistorien keksiminen 1948, COBOL- ja FORTRAN- ohjelmointikielien kehittäminen 1950–1960-luvuilla, mikropiirien (IC) keksiminen 1958, käyttöjärjestelmien (OS) kehittäminen 1960-luvulla, mikroprosessorien kehittäminen 1970-luvun alussa, henkilökohtainen tietokone (PC) vuonna 1981 ja muistin sekä näyttöpäätteiden kehittäminen riittävälle tasolle (Gonzales & Woods, 2007, s. 27). Nykyään tietokoneet ovat niin tehokkaita, että niillä pystytään käsittelemään tarkkoja kuvia tai videoita reaaliaikaisesti. Kuvankäsittelyssä rajoitteina on silti CPU:n tehokkuuden lisäksi keskusmuistin (RAM, engl. random-access memory) tiedonsiirtonopeus ja muistin riittävyys suurilla datamäärillä – erityisesti videoiden käsittelyssä.

Kuvankäsittely perustuu muun muassa luokitteluun (engl. classification), ominaisuuksien erottamiseen, laaja-alaiseen signaalianalyysiin (engl. multi-scale signal analysis), projektioon sekä hahmontunnistukseen (engl. pattern recognition). Kuvankäsittelyssä voidaan käyttää esimerkiksi HMM:a, neuroverkkoja, suotimia tai itseorganisoituvia karttoja. Suodattaminen on yksi tärkeimpiä menetelmiä, jonka avulla kuvasta voidaan poistaa kohinaa tai terävöittää kuvan reunoja. Esimerkiksi reunojen havainnointi on mahdollista tehdä värikuvalle käyttämällä kaavaa 21:

$$D(p_1, p_2) = \sqrt{(R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2}$$

Kaava 21. *Reunojen havainnoinnissa käytettävä viereisten pikselien vertailukaava.*

Kaavassa 21 D kuvaa etäisyyttä kahden pikselin p_1 ja p_2 välillä. Pikselit koostuvat värikomponenteista punainen R, vihreä G ja sininen B. Yhtä pikseliä verrataan oikeanpuoleiseen ja alempana olevaan pikseliin, ja mikäli pikselien väriero D ylittää kynnysarvon, asetetaan ulostulokuvan pikselin arvoksi joko valkoista väriä vastaava arvo, tai harmaan sävy. Mikäli kumpikaan vertailukohta ei ylitä kynnysarvoa, asetetaan ulostulokuvan pikselin arvoksi mustaa väriä vastaava arvo.

Signaalinkäsittelyssä suoritettavat operaatiot riippuvat voimakkaasti tehtävistä, joita pyritään ratkaisemaan. Äänisignaali on koodattu aika- tai taajuusalueeseen, kun taas digitaalinen kuva on koodattu kaksiulotteiseen spatiaaliseen alueeseen. Lukuisat menetelmät, kuten hahmontunnistus, ominaisuuksien erottelu tai luokittelu ovat työkaluja tavoitteen saavuttamiseksi. Suotimien avulla kohinaa voi vähentää, mutta sitä pystyy myös lisäämään. Väriavaruuksien avulla piirteitä voidaan havaita eri tavalla, sillä eri värimallit koostuvat eri komponenteista, kuten kirkkaudesta, luminanssista tai yksittäisistä värikanavien lukuarvoista. Väriavaruuden vaihtaminen voi parantaa verkon suorituskykyä (He et al. 2019). Esimerkiksi ominaisuuksien erottamisessa histogrammin tasoitus on hyvä vaihtoehto, sillä se vahvistaa harmaasävykuvan pikselien jakauman tasaiseksi. Histogrammin tasoittamista käytetään kuvan laadun parantamiseen (Gonzales & Woods, 2007, s. 142). Myös värikuvien histogrammi voi olla hyödyllinen esitysmuoto esimerkiksi objektintunnistusta varten (Shapiro & Stockman, 2001, s. 221).

3.1 Värimallit ja -avaruudet

RGB (engl. Red, Green, Blue) on yleisimmin käytetty värimalli valon sekoittamiseen perustuvissa laitteissa, kuten näytöissä (Eskelinen, 2002, s. 11). Sen pohjalta on luotu väriavaruuksia, kuten HSL ja sRGB. Värimalli on matemaattinen tapa kuvata värit, kuten RGB tai CMYK (engl. Cyan, Magenta, Yellow, Key (Black)). Väriavaruus sen sijaan on joukko värejä, jotka voidaan näyttää tai toistaa. Esimerkiksi sRGB on erityinen joukko punaisen, vihreän ja sinisen intensiteettejä. Väri muodostuu sekoittamalla keskenään näiden kolmen värin valoa, jolloin värimallia kutsutaan myös additiiviseksi.

HSL (engl. Hue, Saturation, Lightness) - väriavaruus tulee sanoista (väri)sävy, värikylläisyys ja valoisuus. Se on RGB-värimallin esitysmuoto, joka luotiin vastaamaan ihmisen näkökykyä ja värien hahmottamista. Sävyä on mahdoton määrittää ilman esimerkkiä, mutta se on ikään kuin yksi näkyvistä väreistä; punainen, keltainen, vihreä ja sininen, tai kahden näistä kombinaatio (Fairchild, 2005, s. 85). Värikylläisyydellä tarkoitetaan ärsykkeen värikkyyttä suhteessa sen kirkkauteen (Fairchild, 2005, s. 88). Valoisuus voidaan määrittää ärsykkeen kirkkaudella suhteessa vastaavasti valaistun valkoisen kirkkauteen (Fairchild, 2005, s. 90).

sRGB-väriavaruudessa väri on luokiteltu kolmella numeroarvolla, joista jokainen vastaa yhtä kolmesta pääväristä: punainen, vihreä ja sininen (Patin, 2003, s. 5). Väriarvoina $(\mathrm{R}, \mathrm{G}, \mathrm{B}) = (0, 0, 0)$ vastaa mustaa ja $(\mathrm{R}, \mathrm{G}, \mathrm{B}) = (255, 255, 255)$ vastaa valkoista. Yhteensä värikanavien kombinaatioita on 16,8 miljoonaa, sillä jokainen värikanava voi saada 256 eri arvoa. RGB:n värien näyttäminen ruudulla on kuitenkin laiteriippuvaista, mikä voi johtaa erilaisiin realisoituviin värimääriin laitteiden välillä. Lisäksi väri on ongelmallinen käsite, sillä ihmissilmä ja erilaiset laitteet näkevät ja tuottavat värejä eri tavalla (Eskelinen, 2002, s. 10).

Kirjallisuudessa on määritetty, että värit koostuvat additiivisesti hyödyntäen esimerkiksi päävärien valoa (Patin, 2003, s. 4–5). Väri on määritelty kirjallisuudessa myös värisävyn ja värikylläisyyden yhdistelmänä, jossa väri on riippumaton valoisuudesta (Watkinson, 2008, s. 232–234). Valoisuudesta riippumattomuuden vuoksi kaikki RGB-arvot, joilla on sama suhde toisiinsa tuottavat saman värin, kuten esimerkiksi $R = G = B$ tuottaa saman värin oli pikseliarvo 0 tai 255 (Watkinson, 2008, s. 232–234). Tämän vuoksi värimallit ja -avaruudet ovat tärkeitä määrittää, sillä sRGB-väriavaruudessa musta, harmaa ja valkoinen eivät ole sama väri, vaikka $R = G = B$. Kuvankäsittelyssä ja tietokonegrafiikassa valoisuudella sekä värisävyillä on suuri merkitys esimerkiksi valonlähteen suunnan havainnoinnissa.

YPbPr-väriavaruus eli $Y'P_BP_R$ sisältää komponentin Y' eli luma, joka tarkoittaa gamma-korjattua luminanssia (Eskelinen, 2002, s. 10). YPbPr-väriavaruutta käytetään muun muassa JPEG-kuvien pakkauksessa. Y' muuntuu RGB-värimallista kaavan 22 mukaan:

$$Y = 0.2126\,R + 0.7152\,G + 0.0722\,B$$

Kaava 22. *Luman muunnos RGB-värimallista.*

P_B on erotus sinisen B ja Y':n välillä ja P_R on erotus punaisen R ja Y':n välillä. YPbPr on analoginen versio digitaalisesta YCbCr-väriavaruudesta. YCbCr tai $Y'C_BC_R$ -väriavaruudessa Y vastaa luminanssia, mutta Y' vastaa lumaa, joka tarkoittaa, että valon intensiteetti koodataan epälineaarisesti gammakorjattujen RGB-primääriarvojen perusteella. C_B vastaa erotus sinisen B ja kroman (engl. chroma) välillä ja P_R on erotus punaisen R ja kroman välillä. Kroma tarkoittaa pikselin värikkyyttä verrattuna valkoisen viitekohteen kirkkauteen (Eskelinen, 2002, s. 10). Fysikaalisesti luminanssilla tarkoitetaan säteilyä, joka on painotettu kunkin aallonpituuden vaikutuksella tyypilliseen ihmistarkkailijaan, ja se mitataan SI-yksikköinä kandela / neliömetri (cd/m^2). Kandela on SI-järjestelmän mittayksikkö valovoimalle, jota kutsutaan myös valon intensiteetiksi (Suomen standardoimisliitto, 2002, s. 3, 22).

Harmaasävykuva on kuva, jossa jokaisen pikselin arvo on yksittäinen näyte, joka edustaa valon määrää eli intensiteettiä. Harmaasävykuvat koostuvat yksinomaan harmaan sävyistä kontrastin vaihdellessa mustasta valkoiseen. Harmaasävykuvia käytetään kuvankäsittelyssä, sillä kanavien määrän vuoksi syötteen koko on moninkertaisesti pienempi kuin värikuvissa; RGB-värimal-

lissa kolminkertainen ja CMYK-värimallissa nelinkertainen. Muunnos RGB-värimallista harmaasävykuvaksi voidaan toteuttaa kaavan 23 avulla, joka on yleisesti käytössä muun muassa MATLAB-ohjelmointikielen rgb2gray-funktiossa:

$$H = 0.2989\ R + 0.5870\ G + 0.1140\ B$$

Kaava 23. *Muunnos RGB-värimallista harmaasävykuvaksi.*

Kaavassa H vastaa harmaasävykuvan pikselin arvoa ja komponentin R, G ja B ovat sRGB-väriavaruuden värikanavien komponentit. H saa saman arvovälin [0, 255] kuin 8-bittisen sRGB-väriavaruuden komponentit. 8-bittinen arvoväli riittää ihmisen tapauksessa, sillä ihmissilmä ei kykene erottamaan 256 eri valon intensiteettiä. Valoisuusasteiden tarkastelussa on hyödyllistä käyttää myös laajempaa arvoväliä, kuten 16-bittistä arvoväliä, sillä kirkkaiden ja tummien alueiden yksityiskohtia on mahdollista tarkastella yksityiskohtaisemmin laajemmalla arvovälillä.

3.2 Valo ja varjot

Valo eli näkyvä valo on sähkömagneettista säteilyä sähkömagneettisen spektrin siinä osassa, jonka ihmissilmä voi havaita. Näkyvän valon aallonpituus λ on noin 400–700 nm (Shapiro & Stockman, 2001, s. 210). Alapuolelta spektriä rajoittaa ultraviolettisäteily, jonka aallonpituus on välillä 100–400 nm. Yläpuolelta spektriä rajoittaa infrapunasäteily, jonka aallonpituus on välillä 700 nm – 1 mm ja täten se on pidempiaaltoista kuin näkyvä valo. Ihmisen näköaistin avulla pystytään hahmottamaan 3D-ympäristö. Syvyyden hahmottaminen muun muassa perustuu silmien ja ihmisaivojen ympäristöä käsittelevien mallien avulla, joihin ihminen on oppinut jo lapsena. Tärkeitä komponentteja ovat objektiympäristö, valaistus ja objekteista heijastuvan valaistuksen aistiminen (Shapiro & Stockman, 2001, s. 33).

Varjo on ympäristöään tummempi alue, joka muodostuu alueelle, jonka edessä valonlähteeseen nähden on läpäisemätön este. Läpäisemätön este voi päästää läpi valonsäteet osittain tai ei lainkaan. Pistemäisen valonlähteen aiheuttama varjo muodostaa projektion valolähteen edessä olevasta esineestä. Laaja-alainen valonlähde aiheuttaa varjon laidoille puolivarjon, joka ei ole yhtä tumma kuin keskellä oleva sydänvarjo, sillä läpäisemätön este ei estä kaikkea valoa puolivarjon alueelle. Varjojen avulla kuvasta voi tunnistaa ulkoisen valonlähteen parittamalla varjoalueet niitä vastaaviin objekteihin ja projisoimalla varjon reunat objektin vastaaviin reunoihin.

Valonlähteen tunnistuksessa on käytetty muun muassa kuvan pikseleiden intensiteettien laskemista ja mahdollisten valonlähteiden havainnointia yhdistämällä korkeita intensiteettialueita (Laskowski, 2007). Intensiteettien laskemiseksi menetelmä hyödyntää kuvan esikäsittelyssä kaavaa 22, jossa RGB-kuvasta lasketaan luminanssiarvot. Algoritmi koostuu paikallisten maksimien etsimisestä Monte Carlo -menetelmällä, paikallisten maksimien määrän vähentämisestä ryhmiin jakamisen vuoksi sekä ryhmien maksimien valinnasta (Laskowski, 2007). Menetelmä toimii yhdelle tai usealle valonlähteelle, kunhan kaikki valonlähteet ovat näkyvissä kuvassa ja heijastavia pintoja olisi mahdollisimman vähän.

Usean valonlähteen tapauksessa ongelmasta tulee monimutkainen, sillä valon intensiteettitasot nousevat molempien valonlähteiden läheisyydessä ja varjojen havainnoinnista tulee epätarkempaa. Varjot ovat usein puolivarjoja ja ne eivät muodostu tasaisesti, mikäli yksi valonlähde on muita kirkkaampi. Kaksi tärkeää efektiä molemminpuolisessa valaistuksessa ovat pintojen välisen kontrastin väheneminen sekä varjostuksen tai valon porrastuminen tasopintakuvissa (Horn, 1977). Varjojen porrastuminen kasvokuvissa muodostaa vaikeuksia pintojen normaalien tunnistukseen sekä ulkoisen valonlähteen havaitsemiseen.

Varjojen tunnistaminen ja poistaminen ovat prosesseja, joilla parannetaan konenäön (engl. machine vision) sovellusten suorituskykyä, luotettavuutta ja tarkkuutta, esimerkiksi kuvan segmentoinnissa ja objektintunnistuksessa. Varjojen poisto on kriittinen askel kohteen havaitsemisen ja seurannan parantamiseksi (Sanin et al. 2012). Esimerkiksi Guon työryhmän algoritmi perustuu objektin ja sitä vastaavan varjon parittamiseen (Guo et al. 2011). Ensisijaisen valonlähteen suunta on silloin mahdollista estimoida objektin reunan ja varjon reunan projisoinnilla.

Objektintunnistuksessa valon heijastuvuuteen vaikuttavat kovuus, tekstuurit ja reflektanssi eli valon heijastuvuus jostakin pinnasta. Konenäössä käytetään paikallisia tekstuurikuvaajia, sillä niiden avulla voidaan havaita pintoja ja tekstuureja erittäin tarkasti. Esimerkiksi riippumattoman komponenttianalyysin ja skalaarikvantisointirakenteen avulla voidaan tuottaa binäärikoodi jokaiselle pikselille (Kannala & Rahtu, 2012). Reflektanssin laskeminen kuvasta on mahdollista esimerkiksi alueelle vertaamalla sitä sen taustaan (Nayar & Bolle, 1996). Objektintunnistuksessa ulkoisen valonlähteen suuntaa on myös tutkittu tunnetuille objekteille. Menetelmässä käytetään heuristista algoritmia, joka yhdistää esineiden reunojen suunnat ja varjoalueen käyttäen Lambertin pintaheijastuvuutta (Nillius & Eklundh, 2001).

3.3 Suotimet ja objektien havainnointi

Kuvien suodattaminen tapahtuu erilaisten suotimien avulla, kuten liukuvan keskiarvosuotimen (engl. moving average filter), Gaussin suotimen (engl. Gaussian filter) tai kuvan segmentoinnin (engl. image segmentation) avulla. Suotimia käytetään muun muassa kohinanpoistoon, superresoluution tekemiseen eli kuvan terävöittämiseen ja reunantunnistukseen. Kuvankäsittelyssä suotimia käytetään pitkälti kuvan esikäsittelyssä, jossa kuvan ominaisuuksia, kuten objektien reunoja tai kasvonpiirteitä yritetään korostaa. Tällöin konvoluutionaaliselle neuroverkolle syötetyissä kuvissa on ihmissilmälle helpommin havaittavia piirteitä, ja myös neuroverkko voi oppia havaitsemaan halutut piirteet paremmin.

Liukuva keskiarvosuodin toimii laskemalla pikselien väriarvojen keskiarvo tietynkokoiselle suotimelle (usein 3x3-alueelle). Kuvan tarkkuus huononee, mutta kohina poistuu tehokkaasti, sillä kuva tasoittuu. Suotimen käyttö sellaisenaan joko pienentää kuvan kokoa tai kuvan reunoista tulee epämääräisiä. Tämän vuoksi alkuperäiseen kuvaan lisätään reunoihin täyte joko nollista tai reuna-arvoista. Kaavassa 24 esitetään 3x3-kokoinen liukuva keskiarvosuodin, jossa g vastaa lopputulosta, f syötekuvaa ja i sekä j ovat vaaka- ja pystyakselin indeksejä.

$$g[i,j] = \frac{1}{9} * \begin{pmatrix} f[i-1,j-1] + f[i-1,j] + f[i-1,j+1] + \\ f[i,j-1] + f[i,j] + f[i,j+1] + \\ f[i+1,j-1] + f[i+1,j] + f[i+1,j+1] \end{pmatrix}$$

Kaava 24. *3x3-kokoinen liukuva keskiarvosuodin.*

Kuvan segmentoinnissa valitaan raja-arvo, joka yksinkertaistaa kuvaa jakamalla kuvan alueisiin, joissa pikseleillä on samanlaiset ominaisuudet. Kuvasta voi täten tunnistaa objektit ja rajat helpommin. Kuvan segmentointia käytetään konenäön yhteydessä muun muassa objektin- ja kasvontunnistuksessa sekä lääketieteessä (Pham et al. 2000). Kynnysmenetelmällä valitaan arvo, jonka yläpuolella olevat arvot saavat maksimiarvon ja muut arvot saavat minimiarvon, esimerkiksi harmaasävykuvan tapauksessa kaavan 25 mukaan:

$$g[i,j] = \begin{cases} 255, & f[i,j] > 100 \\ 0, & f[i,j] \leq 100 \end{cases}$$

Kaava 25. *Harmaasävykuvan segmentoinnin kynnysmenetelmä.*

Kaavassa 25 g vastaa lopputulosta binäärisestä harmaasävykuvasta, kun f on syötekuva ja i sekä j ovat vaaka- ja pystyakselin indeksejä. Kynnysmenetelmästä kehittyneempi menetelmä on tasapainotetun histogrammin kynnysmenetelmä (BHT, engl. balanced histogram thresholding method), jossa histogrammi tasoitetaan ennen kynnysarvon valitsemista, jotta pikselien arvoväli sijoittuu tasaisemmin kuvaan. BHT:ssa oletetaan, että kuva on jaettu kahteen pääluokkaan: tausta ja etuala. BHT-menetelmä yrittää löytää optimaalisen kynnystason, joka jakaa histogrammin kahteen luokkaan.

Muita menetelmiä kuvan segmentoinnissa ovat muun muassa k:n keskiarvon klusterointi (engl. k-means clustering) ja Otsun menetelmä (engl. Otsu's method). Otsun menetelmä on Fisherin diskriminanttianalyysin (engl. Fisher's Discriminant Analysis) yksidimensionaalinen diskreetti analogi, joka liittyy Jenksin optimointimenetelmään (Liu & Yu, 2009). Täten se vastaa globaalisti optimaalista k:n keskiarvoa, joka on tehty intensiteettihistogrammilla (Liu & Yu, 2009).

Alun perin signaalinkäsittelystä peräisin oleva k:n keskiarvon klusterointi on kvantisointimenetelmä, jonka tarkoituksena on jakaa n havaintoa klustereihin K_0 ... K_k. Jokainen havainto n_i kuuluu lähimmän keskiarvon avulla yhteen klusteriin K_j. Klusterointimenetelmän avulla värien määrää voidaan vähentää haluttuun määrään k.

Kuvankäsittelykontekstissa Gaussin suotimessa jokainen matriisin elementti edustaa pikselin ominaisuutta, kuten kirkkautta tai värin intensiteettiä, ja kokonaisvaikutusta kutsutaan Gaussin sumennukseksi (engl. Gaussian blur). Gaussin sumennus on kuvansumennuksessa käytetty alipäästösuodin, joka käyttää Gaussin funktiota muunnoksen laskemiseksi jokaiselle kuvan pikselille. Sitä käytetään muun muassa esikäsittelyvaiheessa konenäön algoritmeissa, jotta kuvan rakenteet paranisivat eri mittakaavoissa. Gaussin funktio yksiulotteisena on esitetty kaavassa 26 ja kaksiulotteisena kaavassa 27.

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$

Kaava 26. *Yksiulotteinen Gaussin funktio.*

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Kaava 27. *Kaksiulotteinen Gaussin funktio.*

Kaavoissa 26 ja 27 x ilmaisee etäisyyttä origosta vaaka-akselilla, y etäisyyttä origosta pystyakselilla ja σ on Gaussin jakauman keskihajonta. Gaussin funktio ilmaisee myös tilastotieteessä Gaussin jakaumaa eli normaalijakaumaa (engl. normal distribution). Signaalinkäsittelyssä yksiulotteista Gaussin funktiota käytetään Gaussin suotimessa. Kuvankäsittelyssä kaksiulotteista Gaussin funktiota käytetään Gaussin sumennuksessa.

Objektintunnistuksessa on myös käytetty keskeisten alueiden tunnistusta (SRD, engl. salient region detection). SRD:n tarkoituksena on rajoittaa kuvaa ihmisen tekemien huomioiden mukaan säilyttämällä kuvan ydinasia. Sitä voidaan pitää kehittyneempänä, ihmisen näköä imitoivana tapana kuvan segmentoinnissa. Kuvalle keskeisyyden mittaamisessa käytetään muun muassa paikallisten piirteiden kontrastia valaistukselle ja väritiedoille, kuten vuoden 2010 tutkimuksessa (Rahtu et al. 2010). Myös globaalia kontrastia on hyödynnetty, jotta korkean resoluution keskeisyyskarttoja voidaan laskea (Cheng et al. 2014). Keskeisten alueiden tunnistuksesta on myös tehty kontekstitietoista, jossa kuvasta tunnistetaan syötettä vastaavat kuva-alueet (Goferman et al. 2011).

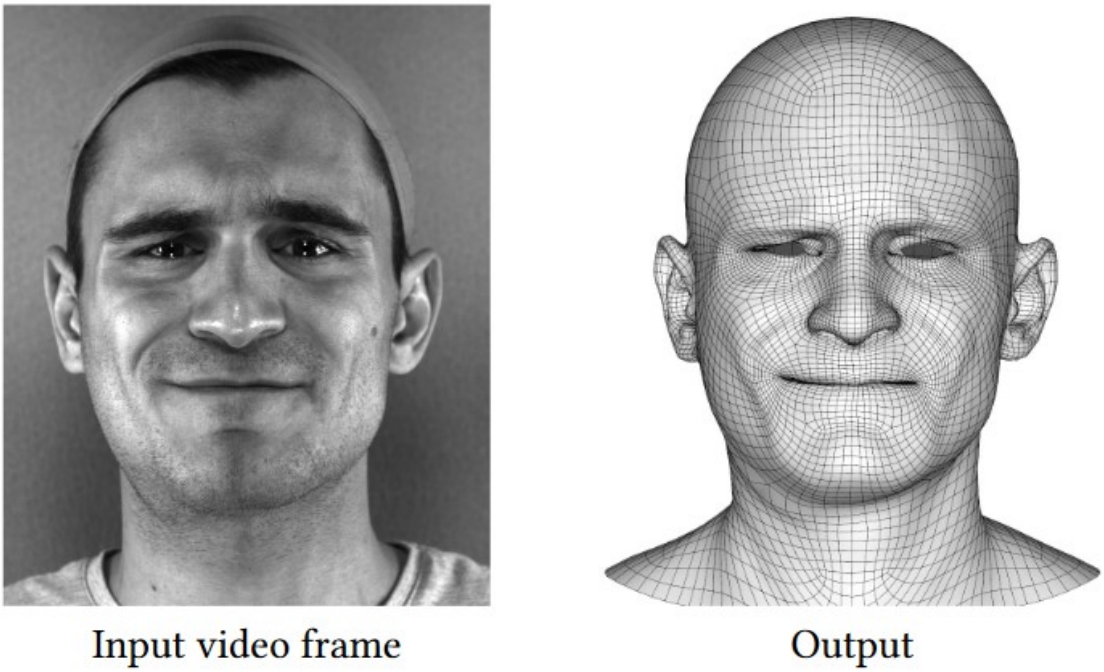
3.4 Valonlähteen havaitseminen ja suunnan estimointi

Valonlähteen voi havaita kuvasta, mikäli se on näkyvä ja kirkas eli sen intensiteetti- ja luminanssiarvot ovat korkeat. Kuitenkaan useimmissa tapauksissa kasvokuvissa ei valonlähdettä ole kuvassa. Ulkoisen valonlähteen tapauksessa valonlähteen suuntaa voidaan estimoida käyttämällä monenlaisia menetelmiä, kuten kasvojen normaaleja, 3D-mallia kasvoista, valon intensiteetin arvoja tai näiden yhdistelmiä. Kasvokuvien tunnistaminen eri valaistuksessa ja eri pään asennoilla on ollut mahdollista jo 2000-luvun alussa. Generatiivisella mallilla on mahdollista luoda olemassa olevan kuvan perusteella generoituja kuvia erilaisilla kombinaatioilla valaistuksen ja pään asennon suhteen (Georghiades et al. 2001).

Kasvojen normaalien ja valon intensiteettien laskeminen kuvasta on potentiaalinen menetelmä valonlähteen havaitsemiseksi kuvasta tai valonlähteen suunnan tunnistamiseksi. Kasvojen normaalien laskeminen perustuu syvyyden hahmottamiseen 2D-kuvasta. Kuvasta syvyys on mahdollista estimoida käyttäen syvää neuroverkkoa ja syvyyden estimointia pikseleittäin, jolloin viereiset pikselit muodostavat tason pinnan normaalien laskemiseksi (Chen et al. 2017). Esimerkiksi ihmiskasvojen pinnan normaalien arvioinnissa on käytetty konvoluutionaalisia neuroverkkoja, jotta tuloksista on saatu tarkkuudeltaan huippuluokkaa (Trigeorgis et al. 2017).

Intensiteettien laskeminen sopii sellaisiin tilanteisiin, joissa valonlähde on jo näkyvissä kuvassa. Intensiteetin avulla valonlähteen havaitseminen perustuu usein heijastuksiin ja kirkkaisiin pintoihin, jolloin virheellisiä positiivisia (engl. false positive) havaintoja tapahtuu kuvissa, joissa valonlähdettä ei ole kuvassa. Laskowski muun muassa käyttää algoritmissaan intensiteetin paikallisia maksimeita potentiaalisten valonlähteiden havainnointiin ja tämän jälkeen vähentää niiden määrää iteroimalla ja ryhmittelemällä kuvasta valittuja pikseleitä (Laskowski, 2007). Menetelmä ei kuitenkaan sovellu käyttökohteeseen, jossa valonlähde on harvoin kuvassa.

Kasvojen ilmeiden tunnistuksessa käytetään kasvoista 3D-mallin luomista, joka muodostaa kuvan ihmiskasvon pikseleistä vektorimaisen 3D-verkon (Laine et al. 2017). Reaaliaikaista kasvonilmeiden havainnointia on tehty samankaltaisella tavalla jo aiemmin. Esimerkiksi pään muotoa on simuloitu videokuvasta 3D-malliksi kasvonkohtien avulla (Cao et al. 2014). Laineen työryhmän algoritmissa käytetään ohjattua oppimista, jossa konvoluutionaalisen neuroverkon ja regression avulla luodaan 3D-vektorimalli syötekuvan ihmiskasvoista (Laine et al. 2017). Kuvassa 10 on esimerkki algoritmin syötteestä ja ulostulosta.

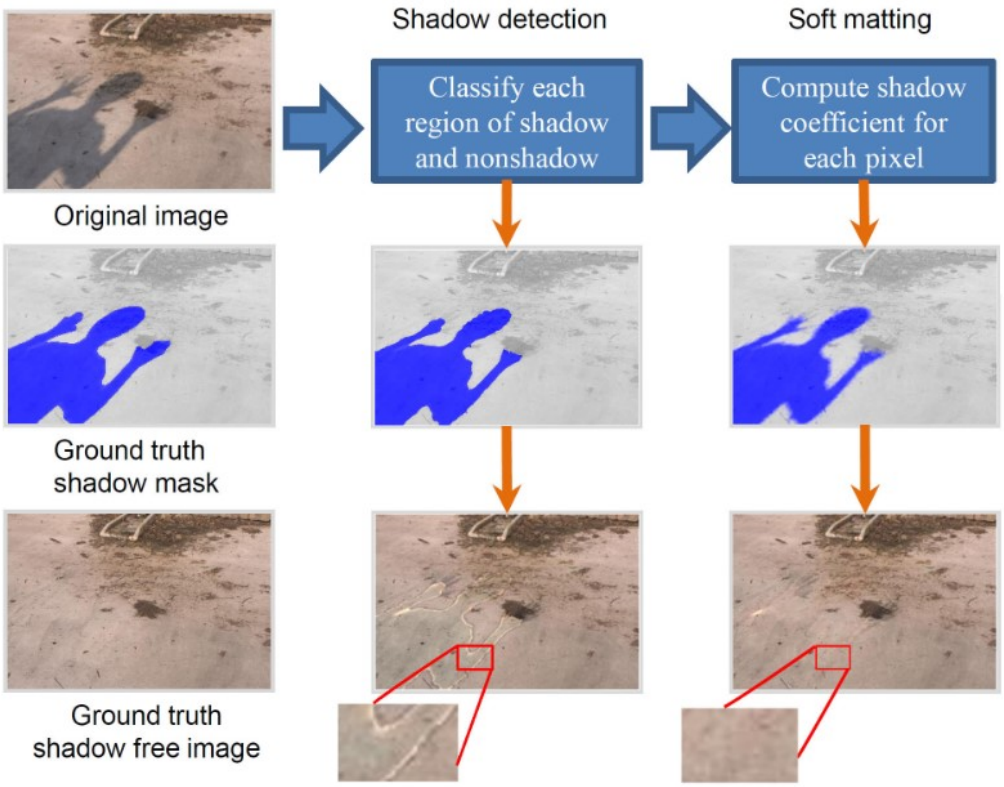


Kuva 10. *Videokuvan syöte ja ulostulo vektoreista koostuvassa 3D-mallissa (Laine et al. 2017, s. 2).*

Tämänkaltaisia menetelmiä pystyttäisiin käyttämään osana valonlähteen havainnointia, sillä yhdistämällä syötekuvasta tehdyn intensiteettikartoituksen 3D-vektorimalliin, voidaan laskea valon tulokulma pinnan normaalien ja kohtisuoruuden avulla. Valonlähteen voi myös sovittaa 3D-verkkoon (engl. 3D mesh), jossa kasvoista on rakennettu verkko sen pintoja mukaillen (Jackson et al. 2017). 3D-mallit ovat tällä hetkellä raskaita laskennallisesti, ja niiden virheherkkyys on kohtalaisen suuri pinnan normaalien suhteen.

Varjojen avulla ulkoisen valonlähteen havainnointi on mahdollista varsinkin, kun kuvassa on objekteja ja niitä vastaavia varjoja. Esimerkiksi Guon työryhmän algoritmissa segmentoidaan kuva käyttämällä keskiarvon siirtoalgoritmia (engl. mean shift algorithm) ja arvioidaan koulutetun luokittelijan avulla jokaisen alueen varjoisuutta kuvaava luottamusväli (Guo et al. 2011). Lopuksi algoritmissa etsitään valaistusparit eri materiaaleille, rakennetaan graafi vastaamaan valaistuspareja ja materiaaleja sekä luokitellaan varjoalueet kuvasta (Guo et al. 2011). Menetelmä toimii kuvissa, joissa objekti ja varjo ovat erotettavissa kuvan muista elementeistä, kuten taustasta. Mikäli kuvassa ei ole selkeää varjoa tai objektia, menetelmä ei ole luotettava. Kasvokuvissa vain kasvoihin nähden sivusta (45–90 astetta) tuleva valo aiheuttaa selkeästi havaittavan varjon, joka on yhdistettävissä sitä vastaavaan kasvonmuotoon. Lisäksi jokaisella ihmisellä on uniikit kasvot, jolloin toisistaan eroavat kasvonpiirteet voivat aiheuttaa virhealttiutta menetelmälle, jossa valon-

lähteestä ja kasvonmuodoista projisoituneiden varjojen avulla yritetään etsiä alkuperäisen valonlähteen sijainti. Kuvassa 11 on esimerkki varjojen havainnoinnista ja poistosta Guon menetelmällä (Guo et al. 2011).



Kuva 11. *Varjojen tunnistaminen kuvasta (Guo et al. 2011, s. 2). Alkuperäisestä kuvasta tunnistetaan varjoalueet ja lopputuloksena muodostuu kuva, josta on poistettu sekä varjoalue että sen reunat.*

Kuvassa 11 havaitaan varjot, jonka jälkeen varjon muodostamat reunat silotetaan vastaamaan ympäristöä. Alkuperäisestä kuvasta on nähtävissä ihmisen varjo, joka havaitaan keskimmäisessä kuvassa lähes kokonaan. Keskellä alareunassa olevassa kuvassa varjon aiheuttamat ääriviivat poistetaan oikean alakulman kuvaan käyttämällä varjon kertoimia jokaiselle pikselille.

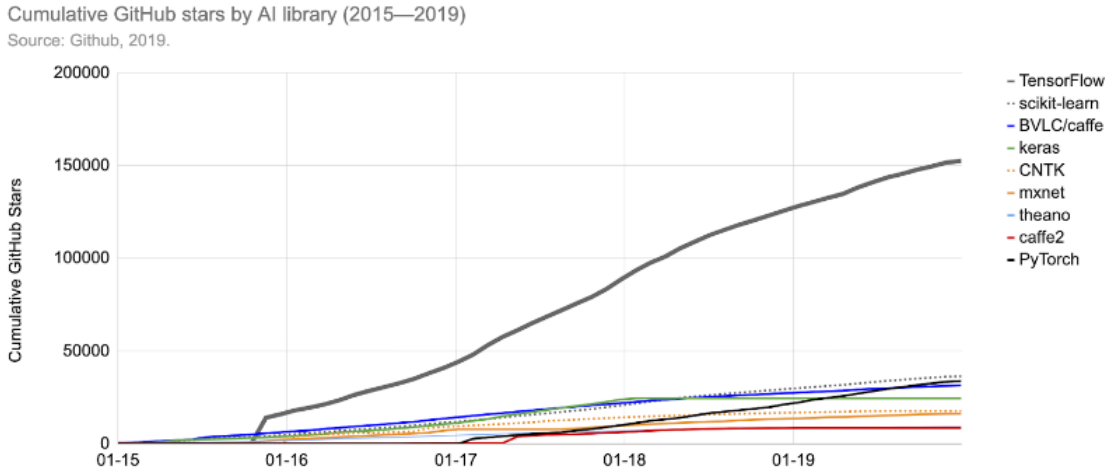
Lineaarinen regressio ja tukivektorikone on myös mahdollinen lähestymistapa ongelmaan. Niiden avulla kuvasta voidaan luokitella valaistut alueet ja varjoalueet. Mikäli menetelmän avulla tunnistettaisiin sekä varjot että kirkkaimmat alueet, joihin valo osuu, voidaan tietoja hyödyntää esimerkiksi pinnan normaalien laskemisessa. Tällaisessa menetelmässä kirkkaista alueista lasketaan pinnan normaalit ja niiden avulla estimoidaan pääasiallisen valonlähteen suunta. Varjoalueilta voidaan myös laskea pinnan normaalit, ja niiden avulla voidaan estimoida tehdyn ennustuksen luotettavuutta. Mikäli varjoalueiden pinnan normaalit osoittavat vastakkaiseen suuntaan kirkkaiden alueiden pinnan normaaleihin verrattuna, on ennustus luotettavampi kuin kulmien välisen eron ollessa pienempi.

4. MATERIAALIT JA MENETELMÄT

Tämän työn päätarkoitus oli tehdä demonstraatio valon suunnan tunnistamisesta kasvokuvasta. Neuroverkolla valon suuntaa tunnistettavaa ohjelmaa tulisi pystyä käyttämään monenlaisilla laitteilla ja haastavissakin ympäristöissä. Suurimpana rajoitteena oli laskentateho, joka rajoittui CPU:hun laitteiston vuoksi. Lisäksi ohjelmiston piti pystyä visualisoimaan tuloksia selkeästi, jotta teknisesti taitamattomampi henkilö pystyy sekä käyttämään että ymmärtämään ohjelmaa. Työssä tehdään seuraavat olettamukset: 1. Kasvojen tulee olla havaittavissa kuvasta. 2. Valonlähteelle on aina olemassa suuntakulma. 3. Ainoastaan ensisijaista valonlähdettä etsitään. Olettamukset ovat työn käyttötarkoitukseen sopivia, vaikka ne eivät yleisesti ole täysin realistisia, sillä kuvat eivät aina sisällä kasvoja ja valonlähteitä voi olla useita.

Työn olettamuksien lisäksi on olemassa rajoitteita. Laitteiston vuoksi työssä käytettiin tietokonetta, jossa oli vain CPU käytössä. Datasettiin kohdistuvia rajoitteita oli datasettien saatavuus ja käyttöoikeudet, joiden vuoksi työssä generoitiin dataa. Generoidulle datalle kasvojen pinnan ei tarvinnut olla sulava, vaikka lähes aina näin kuitenkin oli. Lisäksi kasvojen pigmentille tai valon värille ei asetettu rajoitteita. Pääasiallisen valonlähteen suuntaa kasvojen keskikohtaan estimoitiin, vaikka kuvassa tai kuvan ulkopuolella olisi lukuisia valonlähteitä.

Ohjelmisto on tehty Python-ohjelmointikielen versiolla 3.8. Python on helppokäyttöinen ja monipuolinen ohjelmointikieli, jossa on tehokkaat korkean tason datastruktuurit sekä elegantti syntaksi (Van Rossum & Drake Jr, 2020, s. 1). Neuroverkkojen luonnissa on käytetty Keras-kirjastoa (Chollet et al. 2015), joka on yksi suosituimmista koneoppimisen kirjastoista (Perrault et al. 2019, s. 33–34). Keras pohjautuu lisäksi suosituimpaan tekoälykirjastoon Tensorflow:iin (Abadi et al. 2016), joka on ohjelmoitu tehokkaalla ohjelmointikielellä C++:lla. Suosituimpia koneoppimisessa käytettäviä kirjastoja on esitetty kuvassa 12.



Kuva 12. *Suosituimmat koneoppimisen kirjastot GitHubissa vuosilta 2015–2019 kumulatiivisen tähtimäärän mukaan (Perrault et al. 2019, s. 34).*

Muita tärkeitä työssä käytettyjä kirjastoja ovat numpy, matplotlib, imageio, OpenCV, scikit-image, math ja csv. Numpya (Oliphant, 2006) käytettiin harmaasävymuunnokseen sekä datan alustamiseen koulutuksessa, validoinnissa ja testauksessa. Matplotlib-kirjastoa (Hunter, 2007) hyödynnettiin datan sekä testitulosten visualisoimiseen. Imageio-kirjastoa (Klein et al. 2019) sovellettiin kuvien lukemiseen sekä koulutus- että testivaiheessa. OpenCV-kirjastoa (Bradski, 2000) sekä scikit-imagea (Van der Walt et al. 2014) käytettiin kuvien skaalaamiseen. Math-kirjastoa (Python Software Foundation, 2001) sovellettiin kulmien laskemisessa sekä datan generoimisessa että testaamisessa. Csv-kirjastoa (Python Software Foundation, 2009) hyödynnettiin datan generoinnin yhteydessä varastoimaan kuvien sijainnit ja kulmatiedot CSV-tiedostoon.

4.1 Datasettien muodostaminen

Realististen kuvien käyttö on tärkeää, etenkin testauksessa, jotta ohjelman validointi olisi mahdollista. Kasvokuvista koostuvia datasettejä on olemassa, kuten kasvontunnistuksessa käytetty Photoface (Zafeiriou et al. 2011) tai CAS-PEAL (Gao et al. 2007). Lisäksi menetelmänä sosiaalisten verkkojen kuvadatan ja metadatan käyttäminen olisi ollut potentiaalinen vaihtoehto, kuten kuvanluokittelussa (McAuley & Leskovec, 2012). Niiden käyttöä kuitenkin rajoittaa GDPR ja datasettien saatavuus, sillä kasvokuvien käyttäminen ilman käyttäjän suostumusta rikkoo yksityisyydensuojaa (Council Regulation (EU) 2016/679, 2016).

Opetusdataksi suunniteltiin Yalen yliopiston datasettiä ”Extended Yale Face Database B” (Georghiades et al. 2001). Datasetistä ei kuitenkaan ollut saatavissa käyttökelpoista kulmaa kasvojen ja valonlähteen suhteen, jotta sitä olisi voinut käyttää opetusmateriaalina. Datasetti pohjautui suun ja silmien xy-koordinaatteihin sekä elevaatio- ja atsimuuttikulmiin, jotka laskettiin kameraan nähden. Datasetistä tehtiin approksimaatio, jossa valonlähteen suuntaa estimoitiin kasvojen keskipisteen suhteen. Approksimaatio ei ollut täysin tarkka, sillä kasvojen sijainti kuvassa ja etäisyys kamerasta vaihtelivat kohdehenkilöiden välillä. Datasettiä käytettiin kuitenkin testauksessa, jotta aidoilla valokuvilla saatiin suuntaa antava arvio ohjelman tarkkuudesta. Approksimaatiossa käytettiin kaavaa:

$$\theta' = tan^{-1}(\ (tan(\epsilon)\ /\ tan(\alpha))$$

Kaava 28. *Yalen datasetin kulman muunnos.*

Kaavassa 28 muunnetaan Yalen datasetin käyttämä kameran suhteen laskettu elevaatio- ja atsimuuttikulma kasvojen keskipisteestä laskettuun valon suuntakulmaan. ϵ vastaa elevaatiota ja α atsimuuttia. Tämän jälkeen kulma θ lasketaan kulmasta θ' seuraavasti:

$$\theta = \begin{cases} 2\pi - \theta', & \epsilon \geq 0\ \&\ \alpha \geq 0 \\ \pi - \theta', & \epsilon \geq 0\ \&\ \alpha < 0 \\ \pi - \theta', & \epsilon < 0\ \&\ \alpha < 0 \\ -\theta', & \epsilon < 0\ \&\ \alpha \geq 0 \end{cases}$$

Kaava 29. *Kulman laskeminen eri elevaation ja atsimuutin ehdoilla.*

Kaavassa 29 θ vastaa neuroverkossa käytettyä kulmaa ja muut parametrit vastaavat kaavan 28 merkintöjä. Lisäksi kulmaan vaikuttaa kasvojen sijainti kuvassa, sillä Yalen datasetissä atsimuutti ja elevaatio on laskettu kameran suhteen, jolloin koordinaatisto voidaan normalisoida kasvojen keskipisteeseen hyödyntäen kaavaa 30:

$$O(x, y, z) = \frac{x_{E1} + x_{E2}}{2}, \frac{y_{E1} + y_{E2} + 4 * y_M}{6}, d$$

Kaava 30. *Kuvan koordinaatiston normalisointi kasvojen keskipisteen suhteen.*

Kaavassa 30 lasketaan kuvan koordinaatistolle uusi origo, jolloin vanha koordinaatisto voidaan normalisoida vähentämällä x-, y- ja z-akseleiden arvosta uuden origon akselia vastaavan arvon. $E1$ ja $E2$ vastaavat silmien koordinaatteja, M suun koordinaatteja ja d etäisyyttä kameraan. Datasetistä ei kuitenkaan ole saatavissa etäisyyttä kameraan, jolloin kaavaa ei käytetä approksimoidun kulman korjaamisessa.

Opetusdatan tekemisessä käytettiin Face3D-kirjastoa, jonka on kehittänyt Yao Feng (Feng, 2018). Kirjastoa käyttämällä projektin osana tehtiin dataa generoiva ohjelma, jossa käyttäjä pystyi määrittämään muun muassa pään rotaatiokulmien minimin ja maksimin, valonlähteen x-, y- ja z-koordinaatit sekä valonlähteen värin RGB-muodossa. Face3D-kirjaston toteutuksessa 3D-malli kasvoista asetetaan koordinaatiston keskipisteeksi. Sen päälle renderöidään UV-kartoitus, joka käytännössä asettaa venytetyn 2D-kasvokuvan 3D-mallin päälle koordinaattipisteitä hyödyntäen. Tämän jälkeen projektissa kehitetty ohjelma asettaa valonlähteen koordinaattipisteeseen ja renderöi kasvokuvan ohjelmaan syötetyn valonlähteen värin mukaan. Kasvokuva tallennettiin automaattisesti JPEG-muodossa valittuun kansioon.

Fengin (Feng, 2018) kirjastototeutus simuloi 3D-mallin päälle asetettavaa valaistustekniikkaa. Se oli kehittyneempi versio aiemmista menetelmistä. Esimerkiksi Brunellin menetelmässä 3D-mallin valaistusta kuvasta siirrettiin valaistu alue valokuvassa olevien kasvojen päälle intensiteettiarvoja hyödyntäen (Brunelli, 1997). Kasvokuvan pohjalta renderointia on mahdollista tehdä eri pään asennoilla ja erilaisessa valaistuksessa hyödyntäen 3D-mallia (Georghiades et al. 2001) ja ilman sitä (Choi et al. 2011). 3D-mallin luominen aidoista kasvoista onnistuu kolmiulotteisen muokattavan mallin (3DMM, engl. 3D Morphable model) ja GAN:in avulla (Yin et al. 2017). Fengin Face3D käyttää myös 3DMM-menetelmää, jossa 3D-malli kasvoista on luotu käyttämällä ihmiskasvoja. Ohjelma 1 vastaa sovellettua funktiota yksittäisen kuvan generoimisesta:

```
def light_trans_test(vertices, obj, camera, light_positions, light_intensities, h = 256, w = 256):

    R = mesh.transform.angle2matrix(obj['angles'])
    transformed_vertices = mesh.transform.similarity_transform(vertices, obj['s'], R, obj['t'])

    if camera['proj_type'] == 'orthographic':
        projected_vertices = transformed_vertices
        image_vertices = mesh.transform.to_image(projected_vertices, h, w)
    else:

        ## world space to camera space. (Look at camera.)
        camera_vertices = mesh.transform.lookat_camera(transformed_vertices, camera['eye'],
        camera['at'], camera['up'])
        ## camera space to image space. (Projection) if orth project, omit
        projected_vertices = mesh.transform.perspective_project(camera_vertices, camera['fovy'],
        near = camera['near'], far = camera['far'])
        ## to image coords(position in image)
        image_vertices = mesh.transform.to_image(projected_vertices, h, w, True)

    lit_colors = mesh.light.add_light(vertices, triangles, colors, light_positions, light_intensities)
    rendering = mesh.render.render_colors(image_vertices, triangles, lit_colors, h, w)
    rendering = np.minimum((np.maximum(rendering, 0)), 1)
    return rendering
```

Ohjelma 1. Datan generoiminen Face3D-kirjaston avulla.

Kuvia generoitiin yhteensä 100 000 kappaletta, jotka jaoteltiin kahteen eri koulutusaineistoon: 50 000 kumpaankin. Generoidut kuvat olivat 256x256x3-kokoisia JPEG-kuvia. Molemmissa aineistoissa tehtiin 80:10:10 jaottelu opetusdatan, validointidatan ja testidatan suhteen. Lisäksi Yalen datasetistä havaittiin kasvot 11 105 kuvasta, josta käytettiin 1 000 kuvan satunnaisotoksia validointi- ja testidatassa. Kuvassa 13 on esimerkki yksittäisestä generoidusta kuvasta.

Kuva 13. *Esimerkki Face3D-kirjaston avulla generoidusta kasvokuvasta.*

Todellinen kulma Face3D-kirjaston kuvista saatiin käyttämällä ohjelmia 2 ja 3. Face3D-kirjastossa käytettiin koordinaatistoa, jossa nenänpään koordinaatti oli $(0, 0, 100)$, jolloin valonsuunta laskettiin kasvojen keskipisteen suhteen. Kuitenkin päätä käännettäessä valonsuunta 2D-kuvaan muuttui, sillä kameran sijainti ei muuttunut, mutta pää kääntyi ja valonlähde siirtyi. Tämän takia ohjelmassa 2 esitetään koordinaattien uudelleen laskeminen, kun päätä käännetään x-, y- tai z-akselin suhteen.

```
def rotateX3D(theta, x, y, z):

    theta = theta * (math.pi / 180)
    cos_t = math.cos(theta)
    sin_t = math.sin(theta)

    new_x = x
    new_y = y * cos_t - z * sin_t
    new_z = z * cos_t + y * sin_t

    return new_x, new_y, new_z
```

```
def rotateY3D(theta, x, y, z):

    theta = theta * (math.pi / 180)
    cos_t = math.cos(theta)
    sin_t = math.sin(theta)

    new_x = x * cos_t - z * sin_t
    new_y = y
    new_z = z * cos_t + x * sin_t

    return new_x, new_y, new_z
```

```
def rotateZ3D(theta, x, y, z):

    theta = theta * (math.pi / 180)
    cos_t = math.cos(theta)
    sin_t = math.sin(theta)

    new_x = x * cos_t - y * sin_t
    new_y = y * cos_t + x * sin_t
    new_z = z

    return new_x, new_y, new_z
```

Ohjelma 2. Uusien koordinaattien laskeminen kulman kääntyessä.

Ohjelmassa 3 laskettiin uudet koordinaatit valonlähteelle ohjelman 2 funktioiden avulla, jotta valonlähteen suunta voitiin laskea kameran ja kasvojen keskipisteen suhteen. Tämän jälkeen uusien koordinaattien perusteella laskettiin etäisyys valonlähteeseen, suuntakulma kuvan 2D-tasossa sekä valon tulokulma 3D-avaruudesta kasvojen keskipisteen 2D-tasoon nähden.

```
def compute_complicated_face_angle(x, y, z, pitch, yaw, roll):

    # 3D space with object rotation
    x, y, z = rotateX3D(pitch, x, y, z)
    x, y, z = rotateY3D(yaw, x, y, z)
    x, y, z = rotateZ3D(roll, x, y, z)

    radius2 = math.sqrt(x**2 + y**2 + z**2)
    phi2 = math.atan2(y, x) * (180 / math.pi)
    theta2 = math.acos(z / radius2) * (180 / math.pi)

    return radius2, phi2, theta2
```

Ohjelma 3. Kulmien ja etäisyyden uudelleenlaskenta, jossa päätä on käännetty x-, y- ja z-akseleilla Face3D-kirjaston avulla.

Koska työn alussa ei löytynyt käyttökelpoista datasettiä, otettiin realististen kasvokuvien generointi osaksi työn tavoitteita. Face3D-kirjaston avulla taustaa ei ollut mahdollista generoida mukaan, joten generoitu data ei täysin vastaa realistista kuvaa. Realistisempi ympäristö ja

kasvonpiirteet olisivat vaatineet useampia 3D-malleja kasvoista ja tietokoneen, jolla GPU-kiihdytys on mahdollista ohjelman suorittamiseen. 3D-malleja kasvoista olisi mahdollista generoida tilavuuteen perustuvalla regressioverkolla (VRN, engl. Volumetric regression network), jossa yhden kuvan perusteella luodaan 3D-malli kasvoista (Jackson et al. 2017). Kuitenkin generoidun kuvadatan avulla neuroverkko koulutettiin oppimaan myös todellisten kasvokuvien piirteitä, mikä osoitettiin Yalen datasetin avulla.

4.2 Datan värimuunnokset

Face3D:llä generoituja kasvokuvia oli luomisen jälkeen yhteensä 100 000 kappaletta. Nämä olivat RGB-kuvia, joissa kasvot olivat keskitettynä ja tausta oli musta. Osana toteutusta muokattiin muun muassa datan muotoa, jotta kasvokuvat saatiin vastaamaan erilaisia väriavaruuksia. Lisäksi dataa esikäsiteltiin, jotta tärkeimmät piirteet olisivat havaittavissa helpommin kuvasta. Datamuodon muuttamista tehtiin sRGB-, YCbCr- ja HSL-väriavaruuksiin 5 000 kuvan testierissä. Testitulosten pohjalta päädyttiin sRGB-väriavaruuteen. Päätös tehtiin sRGB-väriavaruuden helppouden ja muunnoksiin kuluvan ajan takia. Lisäksi sRGB-väriavaruudesta muodostettiin suodatettu RGB-muoto, jotta valon suunta olisi nähtävissä mahdollisimman hyvin. Opetusdatan koon pienentämisen takia sRGB-väriavaruudesta muunnettiin kuvat myös harmaasävykuviksi.

Suodatetun RGB-muodon tarkoituksena oli helpottaa ihmissilmää ja konetta havaitsemaan valon tulosuunta. Datan muuntaminen suodatettuun RGB-muotoon tapahtui ohjelman 4 avulla, jossa raja-arvon alittavat lukuarvot asetettiin nolliksi kynnysmenetelmän mukaan.

```
def mask_img(img, mmin, mmax=256):

    n_img = img.copy()
    xm, ym, zm = n_img.shape
    for x in range(0,xm):
        for y in range(0,ym):
            for z in range(0,zm):
                if n_img[x,y,z] < mmin or n_img[x,y,z] > mmax:
                    n_img[x,y,z] = 0

    return n_img
```

Ohjelma 4. Suodatetun RGB-kuvan tapauksessa käytetty kynnysmenetelmä.

Suodatettua RGB-dataa generoitiin yhteensä 50 000 kappaletta, ja ne jakautuivat vastaavasti 80:10:10-jaottelulla opetusdataksi, validointidataksi ja testidataksi. Minimaaliseksi kynnysarvoksi asetettiin 150 ja maksimaalista kynnysarvoa ei asetettu. Tällöin jokaisessa värikanavassa pikselin arvot [150, 255] säilyivät ja niitä pienemmät arvot asetettiin nolliksi. Kuvassa 14 on esimerkki Face3D:n generoimasta RGB-kuvasta, suodatetusta kuvasta sekä harmaasävykuvasta.

Kuva 14. *RGB-kuva, suodatettu kuva ja harmaasävykuva suorasta kasvokuvasta.*

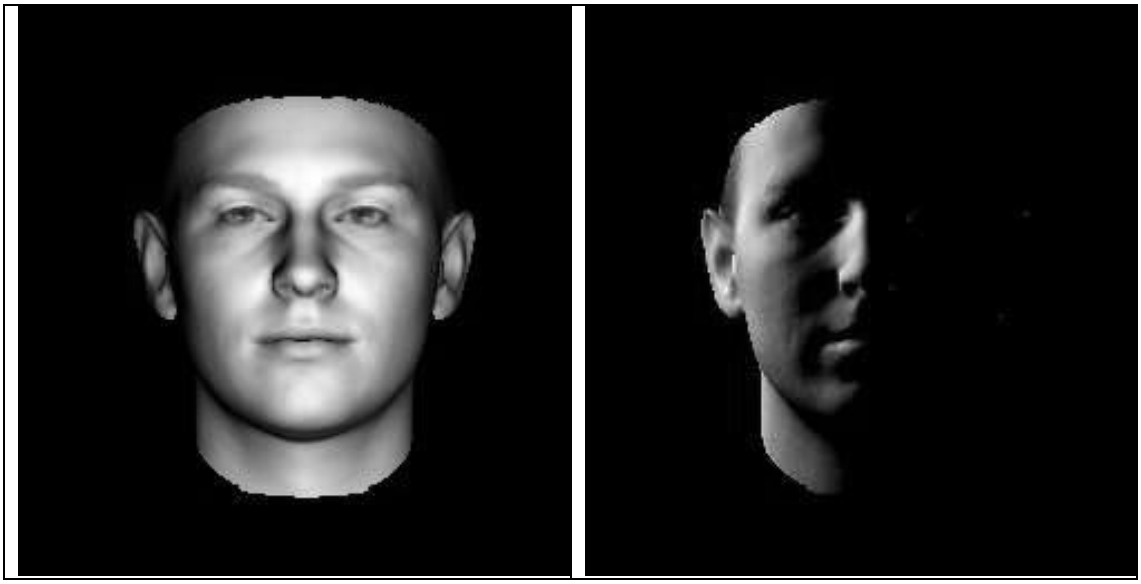
Kuvista on nähtävissä valonsuunta ihmissilmällä vertailemalla esimerkiksi molempien kulmakarvojen ulkosyrjiä, korvia ja nenän varjoa. On havaittavissa, että kynnysmenetelmällä suodatettu RGB-kuva antaa selkeimmän indikaation valonlähteen suunnasta.

Kokeellisessa mielessä dataa muunnettiin sekä YCbCr- että HSL-muotoon. Dataa muunnettiin sRGB-väriavaruudesta digitaaliseen YCbCr-väriavaruuteen, sillä YPbPr-väriavaruus on usein käytetty kuvaa tai videota käyttävissä analogisissa järjestelmissä. Videoteknologiassa YPbPr tai YCbCr erottavat signaalin riittävän hyvin ilman värien erillistä kanavointia. Dataa muunnettiin HSL-väriavaruuteen, sillä valoisuutta kuvaava kanava olisi todennäköisesti havainnut oikeista valokuvista valon suunnan optimaalisesti. Otoksen perusteella konvoluutionaalinen neuroverkko suoriutui paremmin sRGB-väriavaruudessa generoiduilla testikuvilla.

Kaikki 100 000 generoitua kuvaa muunnettiin myös harmaasävykuviksi. Muunnos tapahtui käyttämällä kaavaa 23. Harmaasävykuvissa käytettiin 8-bittistä arvoväliä, sillä funktionaaliset toteutukset tukivat sen muunnosta. Konvoluutiokerrosten ominaisuuksien tunnistamisessa havaittiin samankaltaisia värialueita, jolloin pikselin värien tarkkuudeksi riitti väli [0, 255]. Harmaasävykuvamuunnosten tarkoituksena oli muun muassa vähentää neuroverkkojen opetuksessa kuluvaa aikaa. Harmaasävykuva muutti kuvan kolmesta värikanavasta yhteen, jolloin konvoluutionaalisten neuroverkkojen ei tarvinnut yrittää yhdistää värikanavia keskenään, vaan koko kuva oli yhdessä kanavassa kolmen sijasta. Samalla datan koko pieneni kolmannekseen alkuperäisestä, jolloin myös koulutusaika lyheni noin kolmannekseen.

4.3 Datan sopivuus käyttötarkoitukseen

Ohjelman käyttötarkoituksessa kasvokuvat ovat edestäpäin otettuja, joissa kasvot ovat kohtisuorassa kameraan nähden tai pää on vähän kääntynyt 3D-avaruuden akselien mukaan. Valaistus voi olla mistä suunnasta tahansa 3D-avaruudessa. Opetus-, validointi- ja testidataa generoitiin käyttämällä neljää eri kombinaatiota: valo edestä tai takaa; kasvot suorassa tai vinossa. Kuvassa 15 esitetään kasvokuvia, joissa kasvot ovat suorassa. Kuvassa 16 kasvokuvat ovat kääntyneet joko vähän tai kohtalaisesti.



Kuva 15. *Kasvot suorassa - valo edestä ja takaviistosta.*

Kuva 16. *Kasvot kallellaan - valo kasvojen etupuolelta ja takaa.*

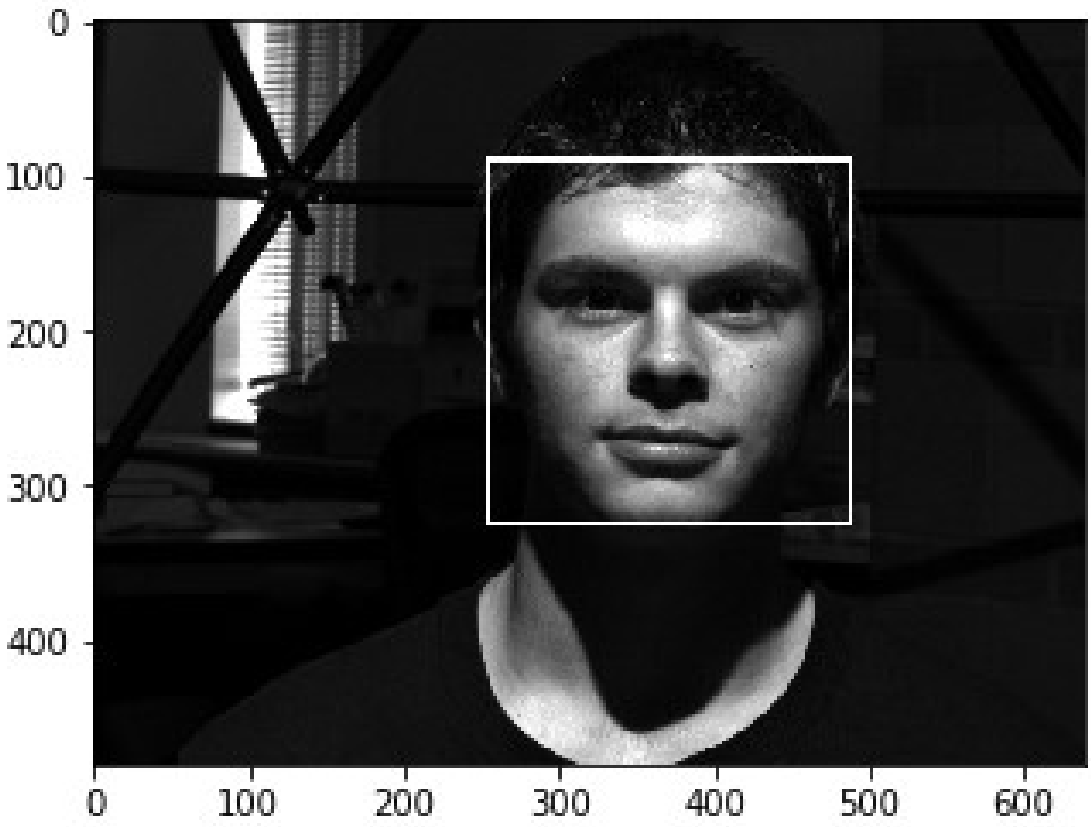
Kuvien generoinnissa kuvat jaettiin kahteen 50 000 kuvan datasettiin. Molemmat datasetit jakautuivat seuraavasti: 10 000 kasvot suorassa valo edestä, 10 000 kasvot suorassa valo sivulta tai takaa, 30 000 kasvot kallellaan valo edestä tai sivulta. Ensimmäisessä datasetissä kallellaan olevat kasvot olivat [-90, 90] astetta jokaisen akselinsa ympäri ja toisessa datasetissä vastaava arvoväli oli [-15, 15] astetta. Ensimmäisessä datasetissä noin kaksi prosenttia pään asennoista oli luonnottomia, jonka vuoksi toinen datasetti oli realistisempi.

Data soveltuu opetusdataksi, vaikka se on generoitua. Kuvien kasvojen piirteet ovat realistiset ja kuvat on tarkastettu erikseen, jolloin generoidut datasetit soveltuvat tällaiseen kokeilulliseen tutkimukseen. Generoitu data tuo uuden perspektiivin neuroverkkojen kouluttamiseen.

4.4 Datan esikäsittely

Käyttämällä sekä verkon koulutusvaiheessa että testivaiheessa samanlaista esikäsittelyprosessia, verkko on tarkempi oikeille kasvokuville. Esimerkiksi kasvojen havainnointi ja suotimen asettaminen ovat käytännöllisiä datan esikäsittelyn muotoja tässä tapauksessa. Kasvojen havainnointi kuvasta voisi tapahtua esimerkiksi käyttämällä Matsugun työryhmän (Matsugu et al. 2003) avoimen lähdekoodin algoritmia, jossa kuvasta haettiin kasvoille tyypillisiä muotoja, kuten silmiä, nenää ja suuta. Kasvojen havainnoinnissa päädyttiin havaitsemaan yksinkertaisemmin koko päätä, jolloin FaceDetect-algoritmi (Tiwari, 2017) soveltui ongelmaan tehokkuusrajoitusten takia paremmin.

Datan esikäsittelyssä käytetty FaceDetect-algoritmi (Tiwari, 2017) pohjautuu OpenCV-kirjastoon. FaceDetect on monivaiheinen prosessi, jossa aluksi kuva jaotellaan alueisiin. Alueille tehdään kasvoanalyysi, joka koostuu kymmenistä pienistä testeistä, joista jokainen on edellistä tarkempi. Mikäli kaikki testit menevät läpi, on alueella havaittu kasvot. Suurin osa alueista palauttaa negatiivisen tuloksen ensimmäisissä testeissä, jolloin säästetään aikaa ja voidaan siirtyä seuraavan alueen tarkasteluun. Tässä työssä Yalen datasetille ajettiin FaceDetect-algoritmi. Algoritmin avulla kuvasta havaittiin kasvot, asetettiin kasvot itsenäiseksi kuvakseen ja kuvan koko skaalattiin verkkoon sopivaksi. Kuvassa 17 esitetään FaceDetect-algoritmin havaitsema alue, josta tehtiin erillinen kasvokuva datasettiin.



Kuva 17. *Kasvojen havaitsemisessa rajattu alue.*

Vastaavia kasvojen havainnointiin sopivia algoritmeja on olemassa. Esimerkiksi SSH (engl. Single State Headless) on kasvon havaitsemiseen soveltuva algoritmi, joka havaitsee lukuisat kasvot yhdessä vaiheessa verrattuna paljon monimutkaisempaan VGG-16-malliin (Simonyan & Zisserman, 2014) parametrien määrän suhteen (Najibi et al. 2017). Myös kasvojen osia on mahdollista havaita, jolloin kasvot voitaisiin koota kasvojen osista (Matsugu et al. 2003). Kasvojen havainnoinnissa päädyttiin FaceDetect-algoritmiin, sillä algoritmille löytyi Python-ohjelmointikielelle soveltuva esimerkki. Täten algoritmille piti vain säätää havainnointiparametreja.

Tässä työssä tarkimmat neuroverkot on koulutettu vain vähän kääntyneillä pään asennoilla. Mikäli kuvassa oleva pää olisi kääntynyt 30–90 astetta kohtisuorasta, olisi mahdollista käyttää kuvan esikäsittelyssä kasvojen kääntämistä kohtisuoraan kameraa kohti 3DMM-pohjaista GAN-algoritmia (Yin et al. 2017). Pään kääntämisellä olisi kuitenkin ollut epävarmuutta luovia piirteitä, sillä valon suunnan tunnistaminen olisi haastavampaa ja korjauskulma olisi pitänyt laskea erikseen. Vaikka kasvoista luotaisiin 3D-malli esimerkiksi VRN:n avulla (Jackson et al. 2017) ja pää käännettäisiin kohtisuoraan kameraan nähden, valon tulosuunta muuttuisi kuvaan nähden ja kasvonpiirteet saattaisivat muuttua todellisuudesta poikkeaviksi.

OpenCV-kirjastoa käytettiin kasvokuvien skaalaamisessa testikuville, jotta ne olivat verkkoon syötettäessä samankokoisia kuin opetusvaiheessa käytetyt kasvokuvat. Kuvan koon muuttamisessa käytettiin pikselialueiden interpolointia. Yleensä halutaan, että kuvan koon muuttaminen syötekuvasta muokattuun kohdekuvaan on mahdollisimman sujuvaa (Bradski & Kaehler, 2008, s. 129–130). Interpoloinnin avulla minimoitiin kohinaa ja mahdollistettiin erikokoisten kuvien käyttäminen osana neuroverkon kouluttamista ja testaamista.

Objektintunnistusta testattiin laajennettavuuden kannalta osana projektin esikäsittelyä, sillä eri objekteista ja niiden varjoista voisi myös päätellä ensisijaisen valonlähteen suunnan. Sekä ”Nopea R-CNN” (Girshick, 2015) että ”Nopeampi R-CNN” (Ren et al. 2015) tarjosivat avoimen lähdekoodin, mutta R-CNN tekniikkana oli aivan liian hidas CPU:lla laskettavaksi. Aihe kohdennettiin keskittymään pelkkiin kasvoihin, sillä kasvokuvat ovat tärkeitä monilla aloilla.

4.5 Ohjelmakokonaisuuden kehittäminen

Ohjelmakokonaisuuden kehittämisessä käytettiin Python-ohjelmointikieltä, joka on helppokäyttöinen syntaksiltaan ja ilmainen. Pythonista ja vaadituista kirjastoista on mahdollista tehdä erillinen paketti, joka ohjelman lisäksi voidaan siirtää eri laitteille ja käyttöympäristöihin. Lisäksi Python tukee lukuisia koneoppimiskirjastoja, kuten Keras ja Tensorflow. Projektin tarkoituksena oli demonstroida, että valon suuntaa on mahdollista havaita kuvasta käyttämällä konvoluutionaalisia neuroverkkoja. Keras on yksinkertaisempi kirjasto havainnollistamiseen kuin Tensorflow, joten projektin aikataulutuksen vuoksi päädyttiin Keras-kirjaston käyttöön. Tällöin kaikkia verkon komponentteja ei tarvitse rakentaa tyhjästä, vaan voidaan käyttää valmiita konvoluutiokerrosten toteutuksia, jotka pohjautuvat Tensorflow:n arkkitehtuuriin. Ohjelmakokonaisuus on avoimesti saatavissa osoitteessa [1] (Ijäs, 2019).

Ohjelmakokonaisuuteen kuuluu konvoluutionaalisten neuroverkkojen koulutuksessa käytetty ohjelma ja datan generoinnissa käytetty Face3D, johon on tehty muutoksia kasvojen asennon ja valaistuksen säädön yhdistämisen takia. Kokonaisuuteen kuuluu lisäksi kuvan esikäsittelyssä käytetty FaceDetect, jonka parametreja on säädetty. Ohjelmakokonaisuuteen kuuluu myös kasvokuvien suunnan estimoinnissa käytetty ennustustyökalu, joka pohjautuu koulutettuun malliin. Pienempiä skriptejä on lisäksi väriavaruuden tarkastamisessa sekä muunnoksissa käytetty skripti, EXIF-dataa (engl. Exchangable image file format) lukeva skripti sosiaalisen median kuviin sekä visualisointityökalu, jonka avulla saadaan tilastotietoja yksittäisille kuville tai generoidulle testisetille.

Neuroverkkojen opetuksessa käytetty ohjelma sekä muut kehitetyt skriptit muun muassa testaukseen ja visualisointiin perustuvat imperatiiviseen ohjelmointiin. Siinä osakokonaisuudet jaotellaan funktioihin ja pääfunktio ajaa tiedoston vaihe vaiheelta. Muuttujien tilat vaihtuvat ohjelman suorituksen aikana, jonka vuoksi kyseessä ei ole funktionaalinen ohjelmointi. Kyseessä ei ole myöskään olio-ohjelmointi, sillä funktiot eivät ole luokkapohjaisia, eikä perintää tapahdu. Imperatiivisella ohjelmoinnilla on mahdollista saada aikaiseksi nopeasti hyvälaatuista ohjelmakoodia. Empiirisesti voidaan todentaa, että tämän työn ohjelmakoodin taso on vähintään kohtalaista.

Ohjelmakokonaisuuden tärkein komponentti oli neuroverkkojen opetuksessa käytetty ohjelma ja sen avulla luotu malli. Konvoluutiokerroksiin perustuva malli koostui viidestä konvoluutiokerroksesta, joiden piirteiden määrä oli ensimmäisellä kerroksella 16 ja viimeisellä 80. Konvoluutiokerrosten jälkeen malli litistettiin yksiulotteiseksi ja kolme täysin yhdistettyä kerrosta muodosti painoarvot verkolle MLP:n tavoin. Optimoijana käytettiin Adamia ja aktivointifunktioina ReLU:a viimeistä kerrosta lukuun ottamatta. Mallin toimintaa selitetään tarkemmin luvussa 5.

[1] https://github.com/NemoHostem/CNN_Source_of_Light

4.6 Ohjelman testaus ja vertaaminen

Ohjelmaa testattiin automaattisesti käyttämällä testisettiä, joka oli generoitu samalla tavalla kuin opetusdata. Testisetti koostui 10 000 kuvasta, joiden jakauma vastasi koko generoidun datasetin jakaumaa pään asennon suhteen. Testauksessa vertailtiin virhealueita, jotka jaoteltiin väleihin [0, 1), [1, 5), [5, 10), [10, 30), [30, 90) ja [90, 180] astetta. Virhealueita tarkasteleva ohjelmakoodi on esitetty ohjelmassa 5.

```
def evaluate_accuracies(acc_count, sum_angle, est_val, real_val):

    if real_val < -90 and est_val > 90:
        err_val = abs(real_val + 360 - est_val)
    elif real_val > 90 and est_val < -90:
        err_val = abs(est_val + 360 - real_val)
    else:
        err_val = abs(real_val - est_val)
    sum_angle += err_val

    # Accuracy (in degrees) <=1, <=5, <=10, <=30, <=90, >90
    if err_val < 1:
        acc_count[0] += 1
    elif err_val < 5:
        acc_count[1] += 1
    elif err_val < 10:
        acc_count[2] += 1
    elif err_val < 30:
        acc_count[3] += 1
    elif err_val < 90:
        acc_count[4] += 1
    else:
        acc_count[5] += 1

    return acc_count, sum_angle
```

Ohjelma 5. Testauksessa käytetty virhealueiden jaottelu.

Ohjelmaa testattiin myös käyttämällä laajennettua Yalen yliopiston datasettiä (Georghiades et al. 2001), josta havaittiin kasvot FaceDetect-algoritmin (Tiwari, 2017) avulla. Datasetissä olevat valon suuntakulmat laskettiin eri tavalla kuin tässä työssä, joten tarkkaa kulmaa valon suunnasta kasvojen suhteen estimoitiin käyttämällä atsimuuttia ja elevaatiota kaavojen 28 ja 29 mukaisesti. Datasetistä käytettiin 1 000 kuvaa testauksessa, jotta neuroverkon soveltuvuutta realistisiin kuviin voitiin analysoida. Testikuvat skaalattiin verkkoon sopivan kokoisiksi käyttämällä OpenCV-kirjastoa.

Lisäksi noin kymmentä kuvaa opinnäytetyön tekijän datasetistä käytettiin testitarkoituksessa, sillä verkolle täysin vieraat kuvat voisivat aiheuttaa virheellistä toimintaa. Kuville ei ole mitattu tarkkaa valonsuuntaa, mutta kuvista on silmämääräisesti arvioitavissa valon suuntakulma.

Ohjelman sisäisessä vertailussa keskityttiin neuroverkon rakenteeseen sekä toimintaan. Tutkimuksen aikana tehtiin rakenteellinen kokeilu, jossa neuroverkko perustui luokitteluun, eikä regressioon. Neuroverkon koulutuksessa käytetyt kuvat olivat samoja, mutta haluttu ulostulo oli regressiivisen liukuluvun sijasta lähimpänä oikeaa kulmaa oleva kokonaisluku. Ongelma on regressiivinen, mutta luokittelun avulla olisi ollut mahdollista antaa estimoitu luottamusväli, esimerkiksi 25 astetta ± 10 astetta 75 %:n todennäköisyydellä.

Projektissa keskityttiin regressiiviseen lähestymistapaan. Regressiiviset mallit olivat tarkempia kuin luokitteluun perustuvat mallit, sillä asteen tarkkuudella regressiivisen mallin Gray 4 tarkkuus 12,1 % oli merkittävästi tarkempi kuin luokkapohjaisen mallin tarkkuus 2,9 %. Regressiivisistä malleista Gray 4 ei ollut tarkin, vaan sen pohjalta jatkokehitettiin tarkempia malleja, kuten Gray 6 ja Gray 8.

Ohjelmaa ei vertailtu muihin vastaavanlaisiin tutkimuksiin, sillä niiden lähdekoodia ei ollut saatavissa niissä toteutuksissa, jotka vastasivat valon suuntakulman tunnistamista kuvasta. Kasvokuvasta valon suuntakulmaa ei ole aiemmin tutkittu vastaavalla menetelmällä.

Ohjelmaa visualisoitiin käyttämällä testiaineistolle matplotlib-kirjastoa. Kuvien päälle visualisoitiin estimoitu valonlähteen suuntakulma. Lisäksi generoituihin kuviin visualisoitiin todellinen valonlähteen suunta. Virhejakaumista tehtiin tilasto, joka osoittaa sekä koko testisetin virhejakauman että pään eri asentojen ja valaistuksen kombinaatioiden virhejakauman. Lisäksi verkon aktivointikanavia ja verkon koulutushistoriaa visualisoitiin. Visualisointeja esitellään luvuissa 5, 6 ja 7.

5. MALLIEN TOIMINTA

Mallit luotiin kouluttamalla konvoluutionaalinen neuroverkko generoiduilla kasvokuvilla. Generoituja kasvokuvia käytettiin mallien kouluttamisessa välillä [15 000, 80 000]. Lisäksi testaamisessa ja validoinnissa käytettiin generoituja kasvokuvia väliltä [2 500, 10 000]. Projektin tarkoituksena oli kehittää ohjelma CNN-teknologialla, kouluttaa sen avulla malli ja arvioida sen soveltuvuutta valonlähteen suunnan estimointiin. Rajoitetun laskentatehon ja monipuolisen käyttöympäristön takia mallista ei saanut tulla laskennallisesti liian raskas.

Mallit koostuivat pääosin konvoluutiokerroksista ja täysin yhdistetyistä kerroksista, kuten kuvista 18, 19 ja 21 on nähtävissä. Konvoluutiokerroksien välissä tapahtui maksimien yhdistäminen, eräkoon normalisointi ja ReLU-aktivointi. Viimeisen konvoluutiokerroksen jälkeen parametrit litistettiin yksiulotteiseksi, jotta neuroneita voitiin käyttää täysin yhdistetyissä kerroksissa. Täysin yhdistetyissä kerroksissa jokainen edellisen kerroksen neuroni oli yhdistetty seuraavan kerroksen neuroniin. Täysin yhdistetyissä kerroksissa käytettiin myös ReLU-aktivointifunktiota ja satunnaiskarsintaa. Ulostulona kuvien 19 ja 21 regressiivisissä malleissa oli yksi liukuluku välillä [0, 360). Kuvan 18 luokkapohjaisessa mallissa ulostulona toimi kerros, jossa oli 360 neuronia. Jokainen neuroni vastasi yhtä luokkaa, jotka määräytyivät suuntakulman mukaan yhden asteen välein.

Malleja koulutettiin RGB-kuville, suodatetuille RGB-kuville ja harmaasävykuville. Jokaisessa mallissa syötteeksi valittiin kasvokuva, jossa kasvojen alue koko kuvasta oli yli 50 %. Tämä mahdollistettiin käyttämällä FaceDetect-algoritmia sellaiselle aineistolle, jossa kuvassa oli muutakin kuin kasvot. Mallit pohjautuivat konvoluutionaalisiin neuroverkkoihin, joissa parametrien määrä vaihteli välillä [100 000, 1 300 000]. Mallin ulostulona oli luokkapohjaisissa malleissa todennäköisin suuntakulma ja regressiivisissä malleissa estimoitu suuntakulma. Ulostuloa visualisoitiin piirtämällä kuvan päälle arviota vastaavaan suuntaan kulkeva jana. Mikäli todellinen kulma tiedettiin valonsuunnalle, piirrettiin myös se kuvaan.

5.1 Syöte ja kuvien esikäsittely

Mallin syötteenä käytettiin kasvokuvia, joissa kasvojen osuus kuvasta oli yli puolet. Käytännössä opetusdata koostui generoiduista kuvista, joissa kasvojen osuus kuvasta oli noin 50–60 %. Aidoista testikuvista rajattiin FaceDetect-algoritmilla kasvokuva, joka skaalattiin vastaamaan verkon syötteen kokoa. Skaalaus tapahtui käyttämällä pikselialueiden interpolointia, joka minimoi kohinaa ja säilytti tarkkuuden. Verkkoon syötettyjen kuvien koko vaihteli eri malleilla välillä [64x64, 256x256] ja värikanavia syötteillä oli 1 tai 3. Lisäksi syötteen pikseliarvot normalisoitiin välille [0, 255]. Syötteenä käytetyissä generoiduissa kasvokuvissa valonsuunnan asteluku oli satunnaisesti generoitua, mutta suuntakulmien jakauma oli suunnilleen tasajaottunut opetusaineistossa välille [0, 360) astetta.

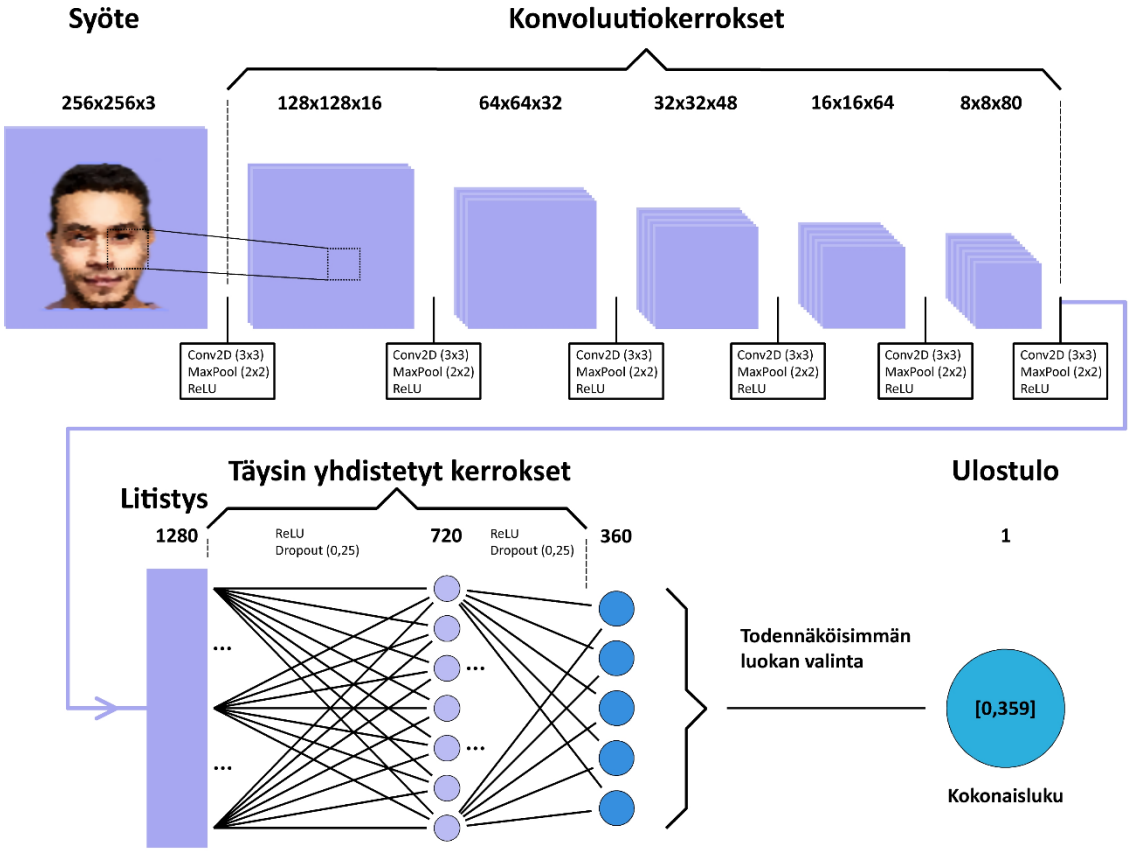
RGB-kuvien tapauksessa muuta esikäsittelyä kuville ei tehty. RGB-kuvat suodatettiin käyttämällä ohjelmaa 4, joka hyödynsi kynnysmenetelmää jokaiselle värikanavalle erikseen. Harmaasävykuvien tapauksessa esikäsittelyssä käytettiin kaavaa 23, joka muutti kuvan RGB-muodosta harmaasävymuotoon. Väriavaruuksien ja suodattamisen osalta esikäsittely oli nopeaa, sillä molemmille tapauksille oli yksittäinen funktio, joka suoritti esikäsittelyn laskennallisesti tehokkaasti.

5.2 Neuroverkkojen rakenteiden vertailu

Konvoluutionaalinen neuroverkko tunnisti syötekuvasta piirteitä konvoluutiokerrosten avulla. Konvoluutiokerroksen piirteentunnistuksessa käytettiin 5x5-kokoista pikselialuetta luokkapohjaisen mallin ensimmäiselle kerrokselle. 256x256-kokoisen kuvan datamäärän suuruutta pienennettiin yhdistämällä maksimiarvot jokaiselle 2x2-kokoiselle alueelle. Luokkapohjaisessa mallissa maksimien yhdistäminen tehtiin RGB-kuvan kaikille kolmelle kanavalle. Regressiivisessä mallissa

maksimien yhdistäminen tehtiin harmaasävykuvalle, jossa värikanavia oli vain yksi. Aktivointifunktiona käytettiin ReLU:a, joka lineaarisesti säilytti positiiviset kertoimet ja nollasi negatiiviset.

Kirjallisuudessa tarkimmat neuroverkot olivat syviä ja ne koostuivat lukuisista konvoluutiokerroksista (Krizhevsky et al. 2012) (He et al. 2016) (Szegedy et al. 2015). Tässä työssä mallin tarkkuus oli avainasemassa, joten neuroverkosta lähdettiin tekemään mahdollisimman syvää rajoitukset huomioon ottaen. Ensimmäisenä toteutuksena koulutettiin kuudesta konvoluutiokerroksesta koostuva luokkapohjainen malli RGB-kuvalle. Toteutuksen tarkoituksena oli saada tarkempi käsitys verkon tarkkuudesta asteen tarkkuudella. Verkon antama tulos oli 2,9 % oikein ja laskemalla viiden asteen maksimivirheelle tulos oli 21,2 %. Mallin rakenne on esitetty kuvassa 18.



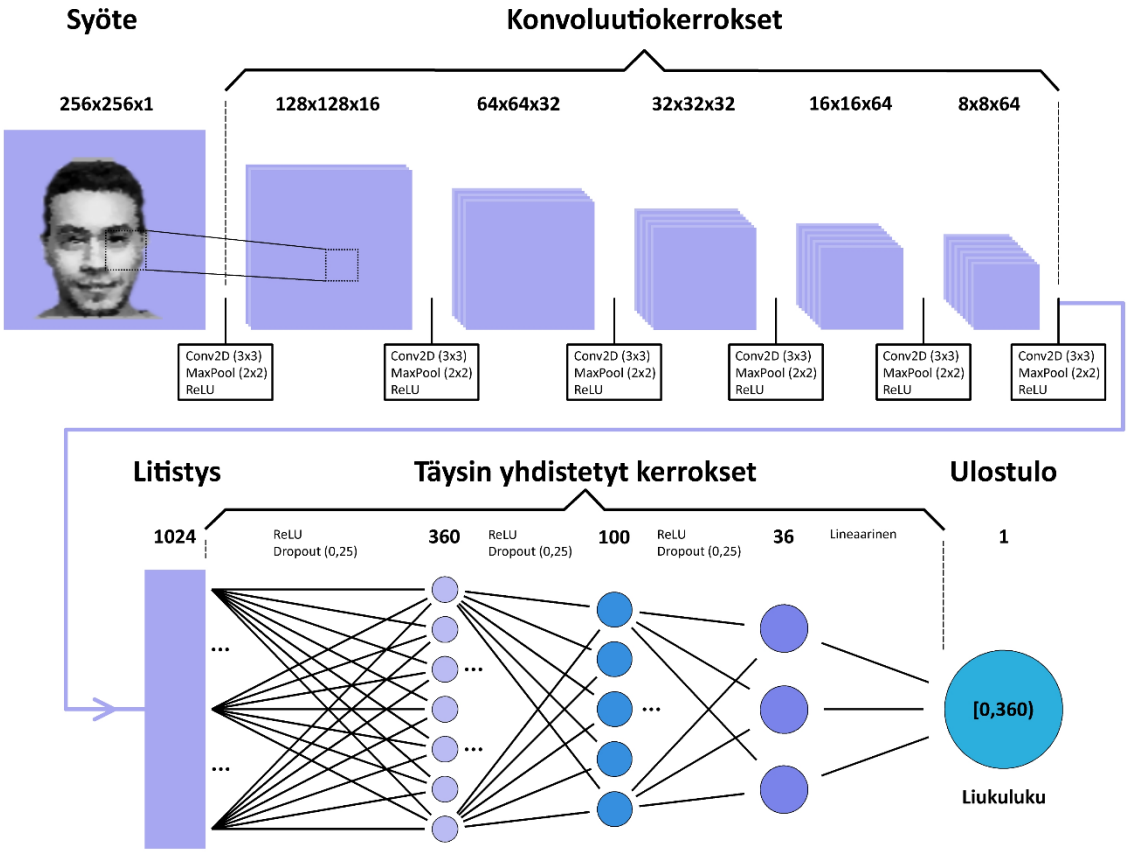
Kuva 18. *Luokkapohjaisen verkon rakenne kaaviona. Syötteenä toimii värikuva. Konvoluutiokerroksissa käytetään 3x3-suodinta, maksimien yhdistämistä ja ReLU-aktivointifunktiota. Täysin yhdistetyissä kerroksissa käytetään satunnaiskarsintaa ja ReLU:a. Ulostuloksi valitaan todennäköisin luokka 360:stä kokonaislukuvaihtoehdosta.*

Kuvassa jokaisen konvoluutiokerroksen laatikon jälkeen tehtiin listatut toimenpiteet eli konvoluutio, maksimien valinta ja ReLU-aktivointifunktio. Konvoluutiokerrosten, maksimien yhdistämisen ja ReLU-aktivointien jälkeen 4x4x80-kokoinen data litistettiin yksiulotteiseksi. Litistyksen jälkeen jokainen neuroni yhdistettiin seuraavan kerroksen jokaiseen neuroniin täysin yhdistetyissä kerroksissa. Viimeinen kerros koostui todennäköisyydestä jokaiselle asteluvulle, jolloin todennäköisin luokka valittiin estimoiduksi valon suuntakulmaksi. Todennäköisimmäksi luokaksi valittiin astelukua vastaavan luokan indeksi eli ulostulo, joka oli kokonaisluku väliltä [0, 359]. Mallissa käytettiin tappiofunktiona kategorista ristientropiaa (engl. categorical cross-entropy). Parannuksia mallin tarkkuuteen suunniteltiin, esimerkiksi kolmen todennäköisimmän luokan valintaa tai datan esikäsittelyssä useamman asteluvun asettamista oikeaksi. Nämä todennäköisesti parantaisivat mallin tarkkuutta, mutta eivät välttämättä parantaisi luotettavuutta mallin toiminnasta.

Mallissa oli lukuisia kehitettäviä asioita. Malli koostui 1,3 miljoonasta parametrista, jolloin mallin kouluttaminen kesti kauan CPU:lla. Lisäksi mallin kouluttamiseen vaikutti opetusnopeuden asettaminen liian pieneksi. Ennenaikaista pysäytystä ei käytetty, vaikka validointidatan kannalta paras malli tallennettiinkin erikseen. Viimeisen kerroksen aktivointifunktio ei ollut välttämättä hyvä. Malli antoi kuitenkin vaikutelman, että 20 asteen tarkkuudella keskimäärin ennustava malli olisi realistinen tavoite. Lisäksi oli tiedossa, että regressiivinen malli sopisi todennäköisemmin estimoimaan valon suuntakulmaa.

Seuraavassa laajemmassa toteutusvaiheessa käytettiin suodatettuja RGB-kuvia. Toteutuksissa mallin kokoa pienennettiin neljään konvoluutionaaliseen kerrokseen. Lisäksi mallista tehtiin regressiivinen. Malli koostui noin 600 000 parametrista. Kouluttaminen oli edelleen hidasta, vaikka satunnaiskarsintaa käytettiin täysin yhdistetyissä kerroksissa suuremmalla arvolla. Mallin virheen keskiarvo oli 38,6 astetta. Vaikka suodatettu RGB-kuva näytti valon suuntakulman selkeämmin kuin RGB-kuva tai harmaasävykuva, ei neuroverkko antanut haluttua tarkkuutta.

Kolmannessa toteutusvaiheessa keskityttiin harmaasävykuvien käyttämiseen syötteenä. Verkon koko pieneni jo värikanavien koon pienennyttyä, joten konvoluutionaalisia kerroksien määrää oli mahdollista nostaa takaisin kuuteen. Lisäksi täysin yhdistettyjen kerrosten määrää pystyttiin nostamaan, sillä neuronien määrä säästi laskentaresursseja regressiivisyyden takia. Kuvassa 19 esitetty malli koostui 690 000 parametrista. Mallin valon suuntakulman keskimääräinen virhekulma oli 7,5 astetta. Vastaavaa rakennetta koulutettiin sekä RGB-kuville että suodatetuille RGB-kuville, mutta niiden keskimääräinen virhe oli yli 20 astetta, jolloin mallia päädyttiin hiomaan käyttämällä harmaasävykuvia.



Kuva 19. *Regressiivisen verkon rakenne kaaviona. Kaavio kuvaa mallia Gray 4. Syötteenä toimii harmaasävykuva. Konvoluutiokerroksissa käytetään 3x3-suodinta, maksimien yhdistämistä ja ReLU-aktivointifunktiota. Täysin yhdistetyissä kerroksissa käytetään satunnaiskarsintaa ja ReLU:a. Ulostuloksi saadaan lineaarisen aktivoitumisfunktion avulla yksittäinen liukuluku väliltä [0, 360), joka kuvaa valon suuntakulmaa asteina.*

Kuvan 19 konvoluutiokerroksille käytetään eräkoon normalisointia. Täysin yhdistettyjen kerrosten viimeisten kerrosten välillä käytetään lisäksi lineaarista aktivointifunktiota, eikä ReLU:a. Regressiivisen mallin vuoksi yhdistettyjen kerrosten viimeinen kerros yhdistyy ulostuloon, joka on yksittäinen liukuluku. Tässä mallissa ulostulo saa arvoja väliltä [0, 360) astetta. Mallissa käytetään tappiofunktiona keskineliövirhettä eli MSE:tä.

Aktivointikerroksissa päädyttiin käyttämään ReLU-aktivointikerroksia, sillä ReLU:n ja satunnaiskarsinnan käyttäminen johtivat nopeampaan opetusprosessiin, jossa konvoluutionaalisten kerrosten ajamisessa ei kulunut tarpeettoman paljoa aikaa CPU:lla. Harmaasävykuvien tapauksessa käytettiin myös muun muassa Sigmoid-aktivointifunktioita sekä hyperbolista tangenttia. ReLU:n tarkkuus oli paras näistä vaihtoehdoista, vaikka ero ei ollut merkittävä tarkkuuden suhteen.

Projektin toteutusvaiheissa keskityttiin parantamaan verkon rakennetta ja hyperparametreja siten, että aidoilla ja generoiduilla kasvokuvilla valonsuunnan estimointi tarkentuisi. Koulutuskausien määrä oli 100 ja eräkoko oli 64. Koulutuskausien määrä todettiin riittäväksi, sillä ennenaikainen pysäyttäminen lopetti kouluttamisen ennen koulutuskausien maksimimäärää. Eräkoko mahdollisti mallin kouluttamisen osissa, jolloin muistia käytettiin vähemmän ja koulutus oli nopeampaa. Eräkoon pienuus vaikutti gradientin tarkkuuteen, jolloin pienempi eräkoko painoarvojen säätämisessä johti epätarkempaan gradienttiin.

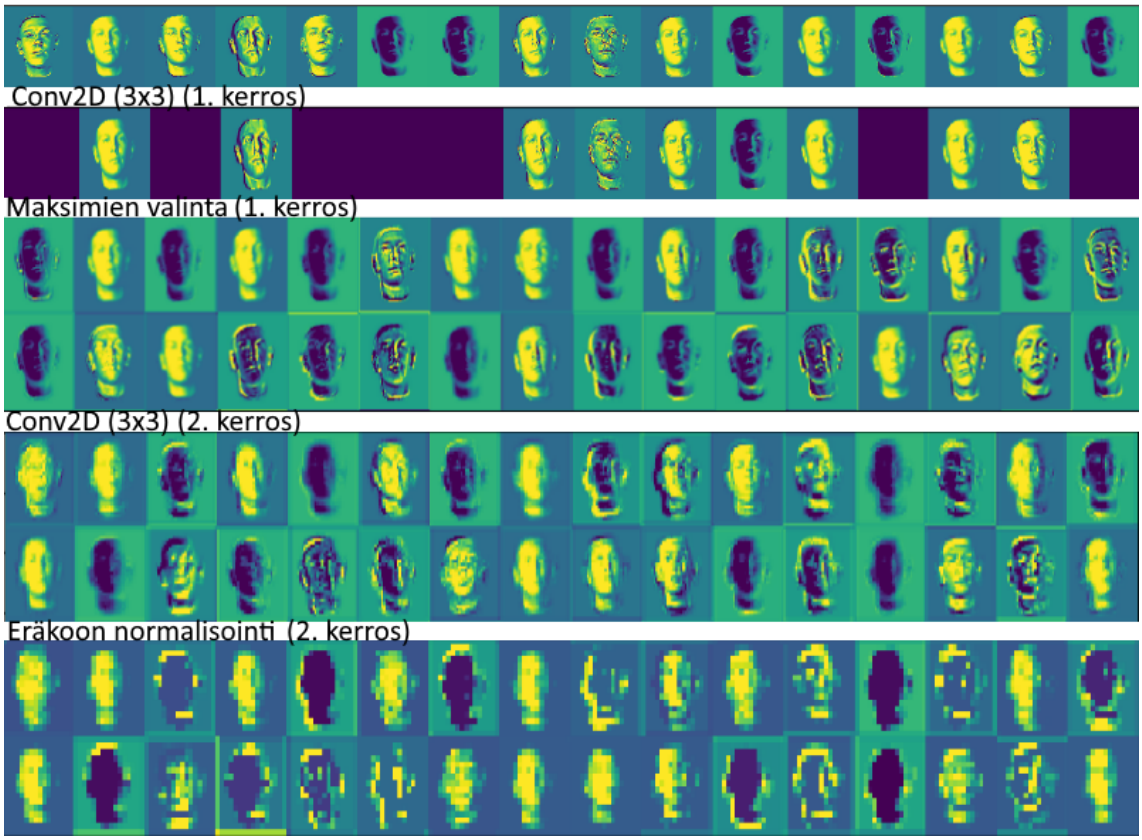
Verkon optimoinnissa käytettiin takaisinkutsufunktioita (engl. callback), joissa verkosta tallennettiin validointidatan perusteella tarkin verkko. Lisäksi verkon tarkkuutta validointidatalla tarkasteltiin koulutushistorian avulla. Adamille asetettu koulutusnopeus muutettiin skaalautuvaksi koulutuskausien ja heikkenemiskertoimen (engl. decay) avulla kaavan 31 mukaan:

$$Lr = Lr * 1/(1 + d * e)$$

Kaava 31. *Kerasin oppimisnopeuden laskeminen jokaiselle koulutuskaudelle.*

Kaavassa 31 Lr kuvaa oppimisnopeutta, d heikkenemiskerrointa ja e koulutuskautta. Oppimisnopeuden skaalaamisella sekä estetään liian suuren oppimisnopeuden aiheuttamaa jäämistä paikalliseen minimiin että nopeutetaan oppimisprosessia liian pienen oppimisnopeuden tapauksessa. Mallissa käytettiin lisäksi ennenaikaista pysäyttämistä, jolloin koulutuksesta tuli sujuvaa ja samalla ylisovittamisen mahdollinen ongelma poistui. Näiden lisäksi ohjelman kaatuminen ei haittaisi niin paljoa, sillä validointidatan perusteella tarkin verkko olisi kuitenkin tallessa.

Jotta neuroverkon toimintaa voidaan ymmärtää, sen aktivointikanavia voidaan visualisoida. Täten saadaan käsitys neuroverkon piirteistä aktivointien osalta. Kuvassa 20 käytetään aktivointien visualisoimisessa avoimen lähdekoodin ohjelmaa ”CNNshapes” (Pierobon, 2018), jota sovellettiin osaksi mallin visualisointia.

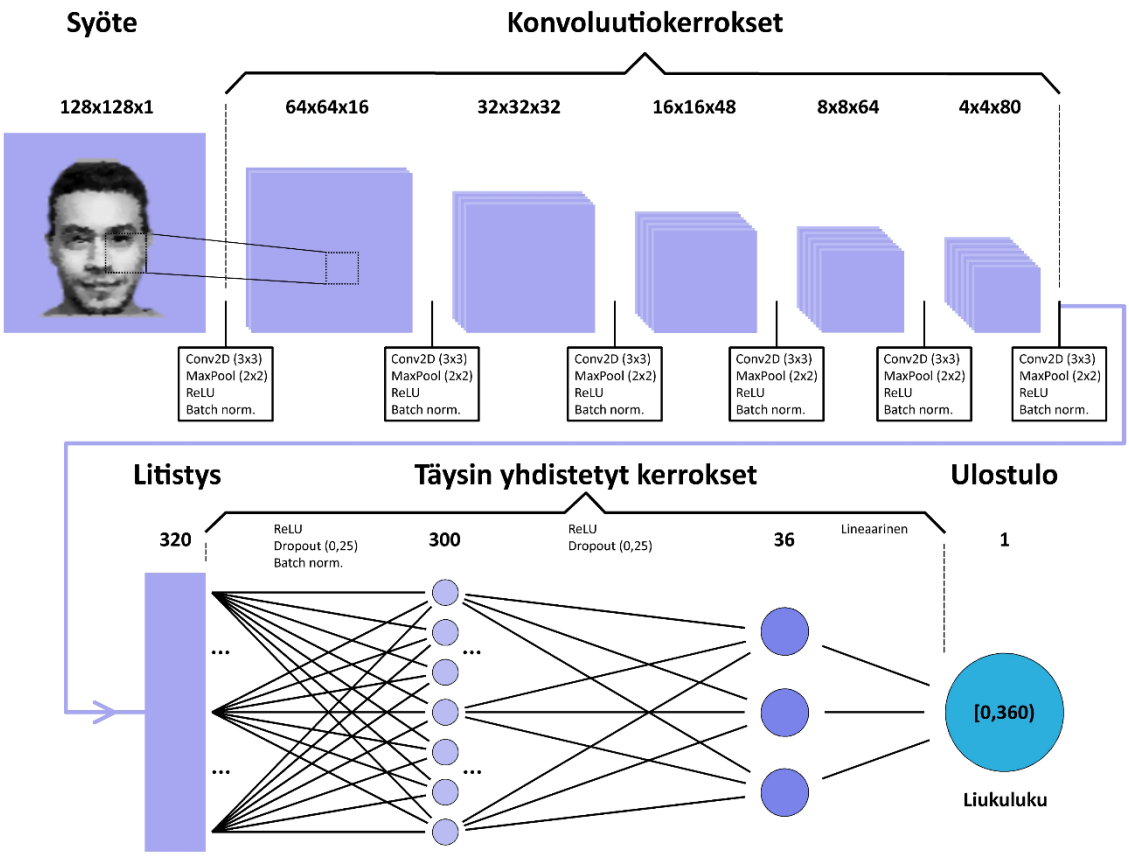


Kuva 20. *Otos ensimmäisten kerrosten aktivointikanavista. Syötteenä olevasta kasvokuvasta visualisoidaan neuronien aktivoitumisia konvoluutiokerroksille, maksimien valinnoille ja eräkoon normalisoinnille.*

Kuvassa 20 kuvataan eri aktivointikanavia harmaasävykuviin pohjautuvalla mallilla. ”Conv2D” kuvastaa konvoluutiokerrosta. Kuvan toiseksi ylimmältä riviltä havaitaan, että kaikki kerroksen neuronit eivät aktivoidu. Aktivointikanavien visualisoimisella päästään konvoluutioneuroverkon mustan laatikon sisään ja nähdään mitä neuroverkon neuroni näkee ihmissilmälle visualisoituna. Kuvasta on myös nähtävissä, että ylimpien rivien kasvokuvaukset ovat tarkkoja, kun alimpien rivien kasvokuvaukset ovat epämääräisempiä ja pikselimäisiä.

5.3 Valitut mallit ja toteutettu rakenne

Lopulta päädyttiin mallien Gray 6 ja Gray 8 rakenteisiin, jotka pohjautuivat kuvan 19 malliin Gray 4. Mallista Gray 4 päädyttiin poistamaan keskimmäinen täysin yhdistetty kerros, jotta mallin Gray 6 koko pieneni 202 000 parametriin ja mallin Gray 8 koko pieneni 490 000 parametriin. Mallien koulutusnopeuden hiominen sekä muut mainitut optimoinnit johtivat mallien keskimääräisen virheen laskemiseen. Mallilla Gray 6 keskimääräinen virhe oli 4,1 astetta ja mallilla Gray 8 keskimääräiseksi virheeksi tuli 4,2 astetta. Malleista Gray 6:lla syötteen koko oli 128x128x1 verrattuna Gray 8:n syötekokoon 256x256x1. Mallin Gray 6 kaavio on esitetty kuvassa 21.



Kuva 21. *Tarkimman regressiivisen verkon rakenne kaaviona mallille Gray 6. Syötteenä toimii 128x128-kokoinen harmaasävykuva. Konvoluutiokerroksissa käytetään 3x3-suodinta, maksimien yhdistämistä, eräkoon normalisointia ja ReLU-aktivointifunktiota. Täysin yhdistetyissä kerroksissa käytetään satunnaiskarsintaa, eräkoon normalisointia ja ReLU-aktivointifunktiota. Ulostuloksi saadaan lineaarisen aktivoitumisfunktion avulla yksittäinen valon suuntakulmaa kuvaava liukuluku väliltä [0, 360).*

Kuvan 21 verkon syöte on 128x128x1, joten se esittää mallia Gray 6. Verkon rakenne on muutoin sama kuin Gray 8:ssa, mutta Gray 8:ssa syötekuvan koko on 256x256x1, jolloin konvoluutiokerrosten koot ovat kaikki kerrottavissa (2x2x1) -elementtikohtaisella arvolla. Täten myös litistyksessä Gray 8:n koko on 1280, kun Gray 6:lla neuroneita on 320. Näiden eroavaisuuksien takia Gray 6 koostuu 202 293 parametrista ja Gray 8 koostuu 490 293 parametrista.

Syötteen koko oli ohjelmoitu parametreihin, jolloin samaa neuroverkon rakennetta voitiin vertailla eri syötteille. Ohjelmassa 6 esitetään konvoluutionaalisten mallien rakenne Gray 6 ja Gray 8.

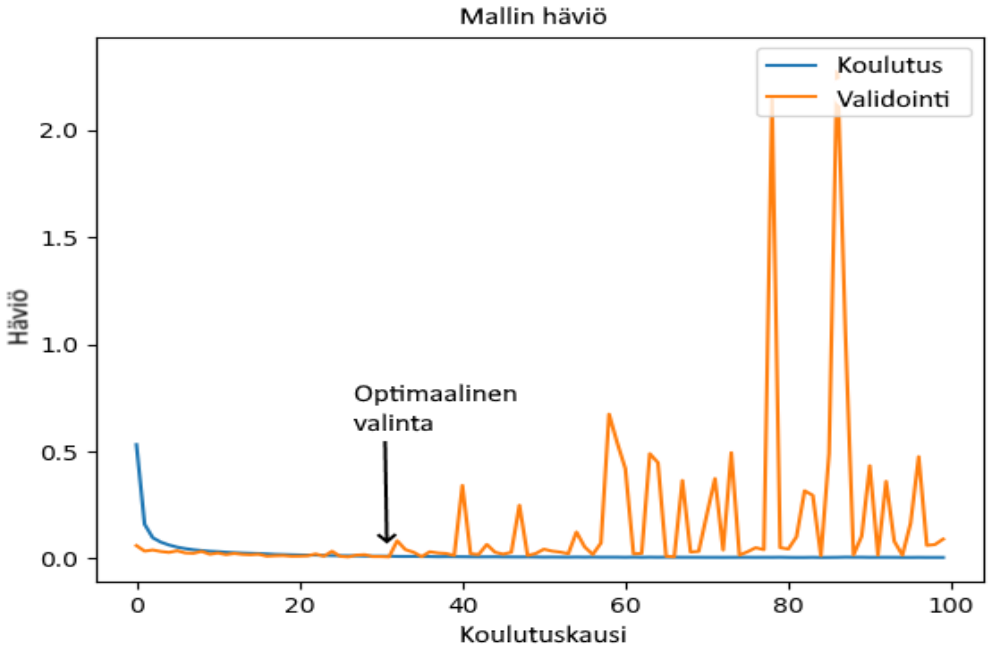
```
def create_cnn(width, height, depth, filters=(16, 32, 48, 64, 80), regress=False):

    input_shape = (height, width, depth, )
    chan_dim = -1
    inputs = Input(shape=input_shape)

    for (i, f) in enumerate(filters):
        # if this is the first CONV layer, then set x as the input
        if i == 0:
            x = inputs

        # CONV => RELU => BN => POOL
        x = Conv2D(f, (3, 3), padding="same")(x)
        x = Activation("relu")(x)
        x = BatchNormalization(axis=chan_dim)(x)
        x = MaxPooling2D(pool_size=(2, 2))(x)

    # Flatten, then FC => RELU => BN => DROPOUT
    x = Flatten()(x)
    x = Dense(300)(x)
    x = Activation("relu")(x)
    x = BatchNormalization(axis=chan_dim)(x)
    x = Dropout(0.25)(x)

    # Apply another FC layer, this one to prepare for regression
    x = Dense(36)(x)
    x = Activation("relu")(x)
    x = Dropout(0.25)(x)

    # if regression is selected, then add output layer for float
    if regress:
        x = Dense(1, activation="linear")(x)

    model = Model(inputs, x)
    return model

model = create_cnn(w, h, d, regress=True)
opt = Adam(lr=1e-4, decay=1e-4 / 400)
model.compile(loss='mse', optimizer=opt, metrics=['mse', 'mae'])
```

Ohjelma 6. Konvoluutionaalisen neuroverkon rakenne.

Ohjelmasta nähdään kaikki oleellinen tieto mallin perustoiminnasta: Malli koostuu viidestä konvoluutiokerroksesta, joissa käytetään 3x3-ikkunointia. Aktivointifunktiona käytetään ReLU:a. Eräkoon normalisointi ja maksimien valitseminen 2x2-alueille ovat myös käytössä. Täysin yhdistetyissä kerroksissa käytetään 300- ja 36-neuronisia kerroksia, joissa on käytössä ReLU-aktivointifunktio ja satunnaiskarsinta. Regressiivisen verkon viimeinen kerros on yksittäinen lukuarvo, jonka muodostamisessa käytetään lineaarista aktivointifunktiota.

5.4 Kouluttamisen pysäyttäminen ja mallin estimaatti

Mallin kouluttaminen lopetettiin, kun viiden koulutuskauden aikana validointisetin tarkkuus ei ollut parantunut. Lopputuloksen kannalta tämä oli tärkeää, sillä neuroverkkojen yksi suurimmista ongelmista oli ylisovittaminen. Ylisovittamisen takia malli ei oppinut käytännön piirteitä kuvista, vaan se yritti sovittaa funktiota opetusaineiston pohjalta, jonka avulla malli olisi voinut oppia datan kokonaan ulkoa. Tällöin mallin käyttäytyminen testisetin suhteen oli ennalta arvaamatonta ja esimerkiksi malli saattoi antaa mahdottomia arvoja testisetin datalle. Kuvassa 22 näkyy erään harmaasävymallin koulutushistoria, jossa ei ole käytetty ennenaikaista pysäyttämistä.



Kuva 22. *Verkon koulutushistoria häviön ja koulutuskausien mukaan. Verkon opetusvaiheen häviötä opetusaineistolle kuvaa koulutus, joka pienenee koulutuskausien mukaan. Validointiaineistolle häviön minimiarvo on optimaalinen valinta koulutuksen lopettamiseen, sillä mallin ylisovittaminen johtaa validointihäviön kasvamiseen.*

Kuva 22 havainnollistaa neuroverkon kykyä saavuttaa minimaalinen häviö koulutusdatan suhteen, mikäli minkäänlaista regularisointia ei ole käytössä. Kuvassa mallin häviön teoreettinen maksimi on 1. Kuitenkin ylisovittamisen tapauksessa validointisetin häviö on yli 1 koulutuskausilla 78 ja 87. Optimaalinen valinta validointisetin perusteella olisi lähellä koulutuskautta 30. Tällöin ennenaikainen pysäytys olisi tapahtunut koulutuskauden 35 jälkeen, jolloin tarkimmaksi malliksi olisi jäänyt koulutuskauden 30 malli.

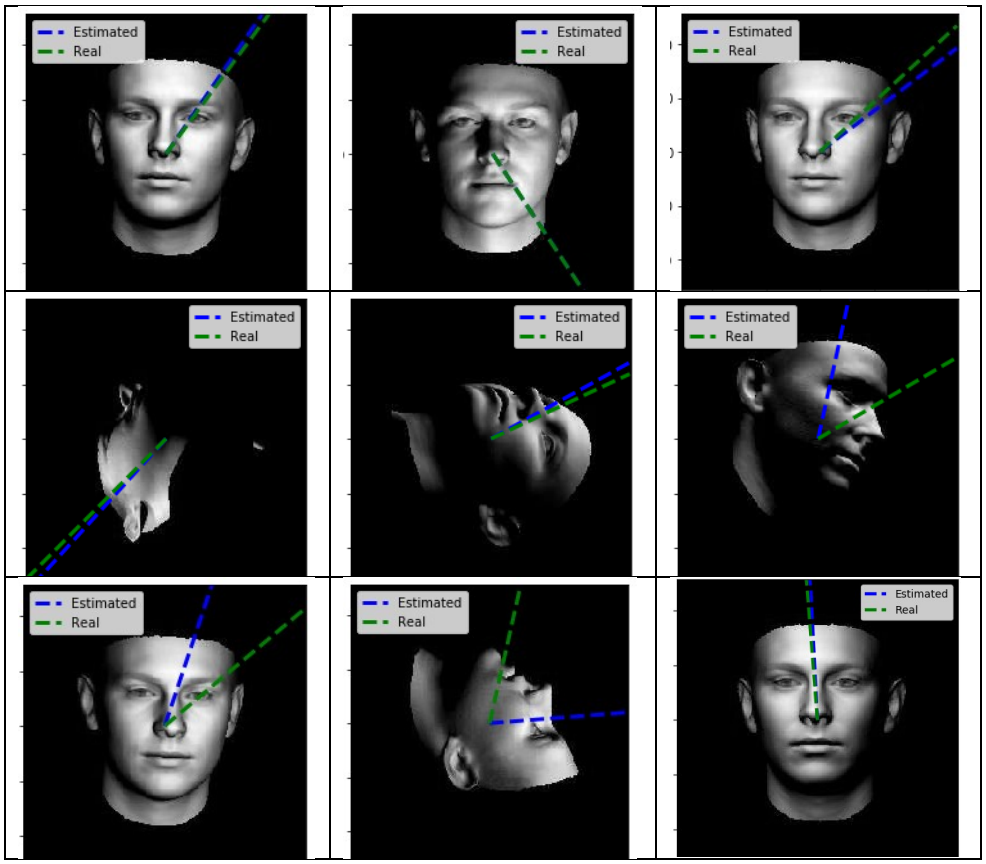
Toimiva malli estimoi testikuvalle valon suuntakulmaa, jolloin mallin ennustuksen pitäisi olla suuntakulma välillä [0, 360) astetta. Sekä regressiivisestä mallista että luokkapohjaisesta mallista data tuli ulos valon suuntakulman astelukuna. Regressiivisessä mallissa luku oli liukuluku, luokkapohjaisessa mallissa luku oli kokonaisluku. Suuntakulma visualisoitiin kasvokuvan päälle siten, että kuvan keskipisteestä (usein myös kasvojen keskipiste) piirrettiin jana suuntakulman osoittamaan suuntaan. Kuva oli mahdollista tallentaa matplotlib-kirjaston tallennustoiminnolla.

6. TULOKSET

Mallit Gray 6 ja Gray 8 olivat generoidun testiaineiston perusteella tarkimmat neuroverkot. Opetukseen käytettiin 40 000 harmaasävykuvaa molemmilla malleilla. Testi- ja validointisetit koostuivat 5 000 generoidusta kasvokuvasta. Koulutuskausien maksimina käytettiin 100 koulutuskautta, mutta verkkojen koulutus lopetettiin 35–40 koulutuskauden jälkeen, koska validointidatan keskineliövirhe ei parantunut koulutuskausien 30–35 jälkeen. Eräkokona käytettiin 64 kuvaa kerrallaan, jonka pohjalta käytettiin myös eräkoon normalisointia useilla konvoluutiokerroksilla.

Malli Gray 6 käytti syötteenä 128x128x1-kokoista kasvokuvaa, joka pienensi mallin parametrien määrää huomattavasti verrattuna malliin Gray 8. Gray 6:lle testiaineisto antoi 4,1 asteen keskimääräisen virheen. Yalen datasetille Gray 6 antoi keskimääräiseksi virheeksi 16,8 astetta. Yalen datasetin tapauksessa tuli ottaa huomioon datan suuntakulman approksimointi kasvoihin nähden, joka ei ollut absoluuttinen totuus vaan approksimaatio perustuen datasetin elevaatio- ja atsimuuttikulmiin.

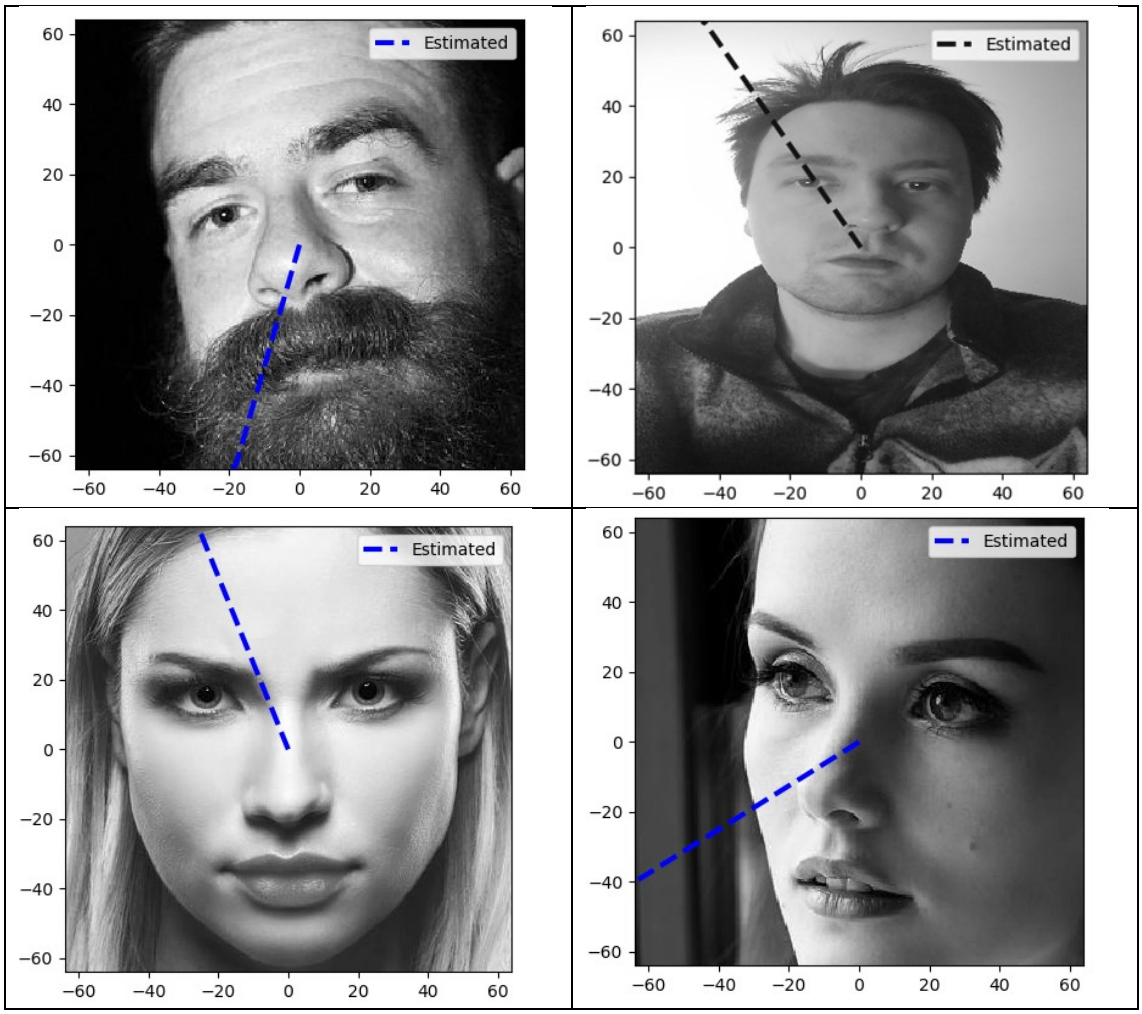
Malli Gray 8 käytti syötteenä 256x256x1-kokoista kasvokuvaa, jonka vuoksi malli koostui 490 000 parametrista. Koulutusaineisto koostui pään asennoista, joissa pää on joko suorassa tai kallistunut x-, y- ja z-akselin suhteen korkeintaan 15 astetta. Generoidulle testiaineistolle keskimääräinen virhe oli 4,2 astetta ja Yalen datasetille virhe oli keskimäärin 15,8 astetta. Tulosten visualisoimisessa käytettiin mallia Gray 8, sillä se suoriutui suurimmalla tarkkuudella Yalen datasetin testeistä. Kuvassa 23 esitetään mallin estimaatteja satunnaisotoksille generoidusta testisetistä.



Kuva 23. *Esimerkkejä generoiduista kasvokuvista. Sininen jana kuvaa estimoitua valon suuntakulmaa ja vihreä jana kuvaa todellista valon suuntakulmaa.*

Kuvan 23 ylärivillä on testikuvia generoidusta testisetistä, jossa kasvot ovat kallistuneet maksimissaan 15 astetta x-, y-, z-akseleiden suhteen. Kuvan alarivillä on testikuvia toisesta generoidusta testisetistä, jossa kasvot ovat kääntyneet maksimissaan 90 astetta jokaisen akselin suhteen. Vihreällä janalla kuvataan todellista valon suuntakulmaa ja sininen jana kuvaa mallista ulos tulevaa estimoitua suuntakulmaa. Kuvissa, joissa pään asento on kääntynyt paljon verrattuna kameraan, suuntakulman estimoiminen on vaikeampaa. Tämä on empiirinen havainto, joka toistuu kaikilla testatuilla malleilla. Oletettavasti konvoluutiokerrokset eivät opi tunnistamaan paljon käännettyjä päitä, jolloin niiden pohjalta valon suunnan estimoiminen on virhealttiimpaa.

Mallia testattiin myös sekä opinnäytetyön tekijän datasetillä että avoimesti Internetistä saatavilla kuvilla, joissa kasvot olivat havaittavissa FaceDetect-ohjelman avulla. Esimerkkikuvista todellista valon tulosuuntaa ei ollut tiedossa. Kuvissa oli lisäksi muita piirteitä, jotka vaikuttivat mahdollisesti valon suuntakulman estimoinnissa: parta, meikki, hiukset ja kasvojen muoto. Valokuvat ja niiden päälle piirretyt estimaatit ovat esitetty kuvassa 24.

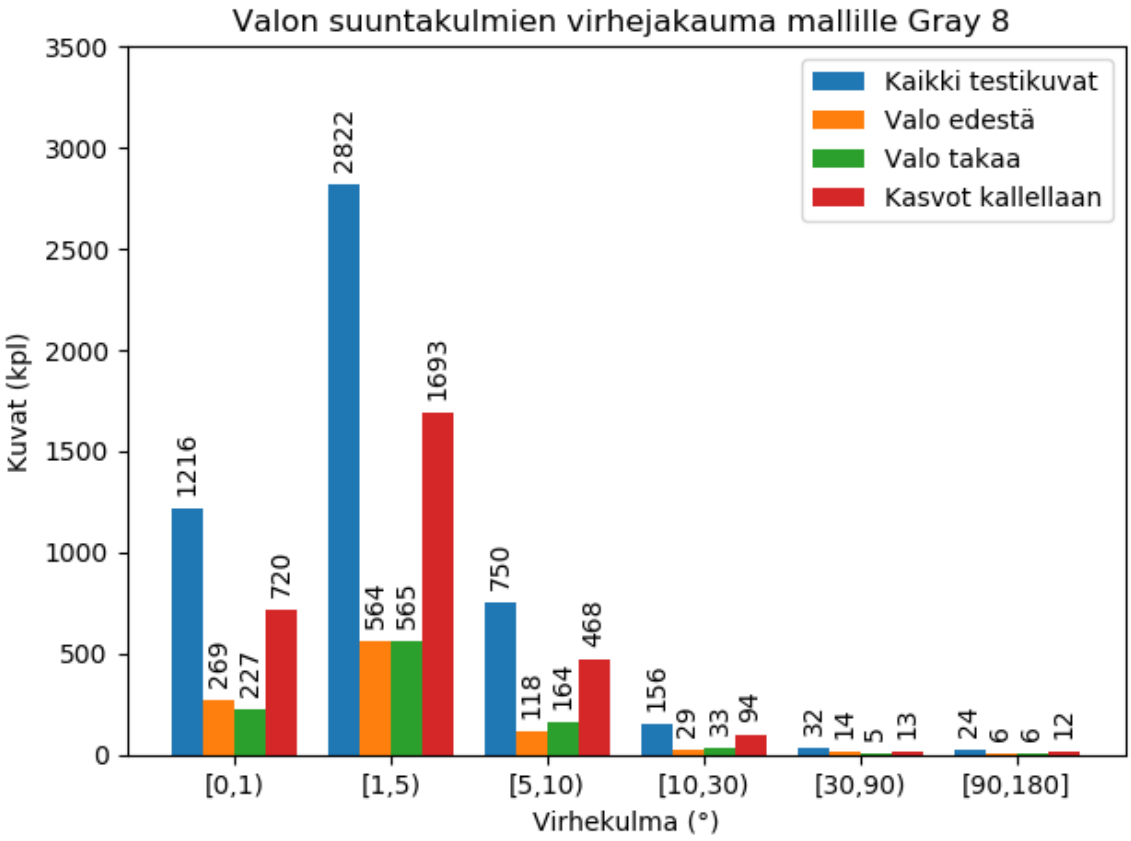


Kuva 24. *Esimerkkikuvia aitojen valokuvien estimoinnista. Katkoviiva kuvaa estimoitua valon suuntakulmaa.*

Kuvan 24 esimerkkikuvista on nähtävissä, että malli tuottaa suuntaa antavia tuloksia saatavissa oleville kasvokuville. Parrakkaan miehen tapauksessa kuvan oikeassa laidassa on havaittavissa varjo, jolloin valo tulee kuvan vasemmalta puoliskolta. Nuoren miehen tapauksessa valo tulee myös kuvan vasemmalta puoliskolta. Kuvassa taustan kirkkaus vaikeuttaa myös ihmissilmälle kasvojen varjoalueiden havainnointia. Alarivin kuvissa estimaatit ovat tarkkoja todelliseen valon suuntakulmaan verrattuna. Kasvojen tasaisuuden vuoksi varjoalueet ovat havaittavissa kohtuullisen helposti ja ne sijaitsevat vastakkaisella puolella estimoituun valon suuntakulmaan nähden.

6.1 Mallien tulosten vertailu

Regressiiviset harmaasävykuviin perustuvat mallit olivat tilastollisesti merkittävästi parempia tarkkuuden suhteen RGB-kuviin tai suodatettuihin RGB-kuviin perustuneisiin malleihin verrattuna. Verrattuna luokkapohjaiseen toteutukseen, jossa mallin tarkkuus oli 2,9 prosenttia ja viiden asteen tarkkuudella mallin tulos oli 21,2 prosenttia, regressiivinen malli Gray 8 suoriutui merkittävästi paremmin. Regressiivisen mallin tarkkuus asteen tarkkuudella oli 24,3 prosenttia ja viiden asteen tarkkuudella 80,7 prosenttia. Kuvassa 25 esitetään mallin Gray 8 virhemäärän jakauma.



Kuva 25. *Tulokset harmaasävykuville mallilla Gray 8. Datasettien virhekulmat on jaoteltu virhealueisiin kasvokuvien kappalemäärien mukaan. Datasetit jakautuvat koko testisettiin ja sen osiin: Suoriin kasvokuviin, joissa valo tulee kasvojen etupuolelta, Suoriin kasvokuviin, joissa valo tulee kasvojen takaa ja Kasvokuviin, joissa kasvot ovat kallellaan valon tullessa mistä tahansa suunnasta.*

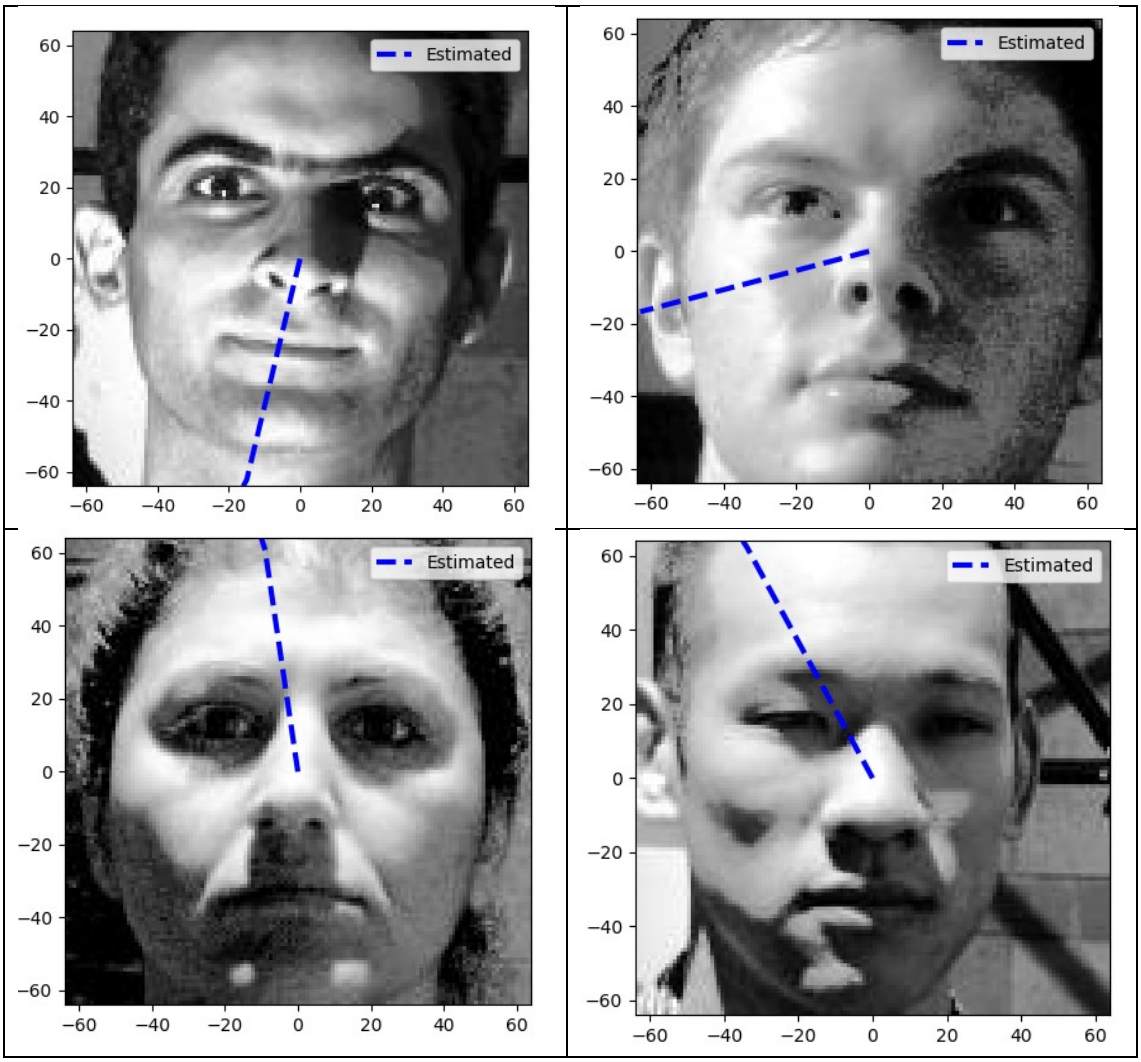
Kuvassa 25 x-akselilla kuvataan absoluuttisen virheen määrää asteina ja y-akselilla kuvataan kuvien määrää. Kuvan pylväsdiagrammissa sininen väri kuvaa koko testiaineistoa. Oranssit pylväät vastaavat suorassa olevia kasvokuvia, joissa valo tulee kasvojen etupuolelta. Vihreät palkit kuvaavat suorassa olevia kasvokuvia, joissa valo tulee kasvojen takaa tai sivuilta. Punaiset pylväät vastaavat kameraan nähden kallistettuja kasvoja. Jakaumasta on nähtävissä, että yhteensä yli 80 prosenttia kasvokuvista kuuluu joko ”[0,1)”-vaihtoehtoon eli virheen määrä on alle yhden asteen tai ”[1,5)”-kohtaan eli virheen määrä on yli yksi, mutta alle viisi astetta. Virheen määrä on yli 90 astetta 24 tapauksessa, joka nostaa keskimääräistä virhettä ja lisää mallin epäluotettavuutta.

6.2 Yalen testidatan tulokset

Yalen datasetille malli Gray 8 antoi keskimääräiseksi virheeksi 15,8 astetta. Gray 6 antoi keskimääräiseksi virheeksi 16,8 astetta. Yalen datasetin suuntakulmat oli annettu eri muodossa ja kaikkia tarvittavia tietoja ei ollut saatavilla, jotta todellinen arvo valon suunnasta olisi ollut mahdollista laskea. Yalen datasetin atsimuuttia ja elevaatiokulmaa hyödyntäen laskettiin estimoitu

valon suuntakulma kuvalle käyttäen kaavoja 28 ja 29. Yalen datasetissä käytettiin 9 eri asentoa ja 64 eri valaistusympäristöä (Georghiades et al. 2001). Pään asennot olivat samankaltaisia kuin opetuksessa käytetyssä datasetissä, jossa pään suurin mahdollinen kulma oli 15 astetta jokaisen akselin suhteen.

Ohjelmassa voitiin visualisoida joko pelkästään estimoitua suuntakulmaa tai sekä estimoitua että todellista suuntakulmaa. Yalen datasetissä ”todellinen” suuntakulma oli myös laskettu estimaatti, jonka virhe oli oletettavasti alle 5 astetta kaikille kuville. Kaavan 28 tangentin laskeminen nollalla oli virhealtista, joten Yalen datasetin tapauksessa atsimuuttiin ja elevaatioon oli lisätty minimaalinen virhe 0,0001, jotta kulmien laskeminen ei aiheuttanut ajonaikaista virhettä. Estimoituja kulmia Yalen datasetin kasvokuville esitetään kuvassa 26.

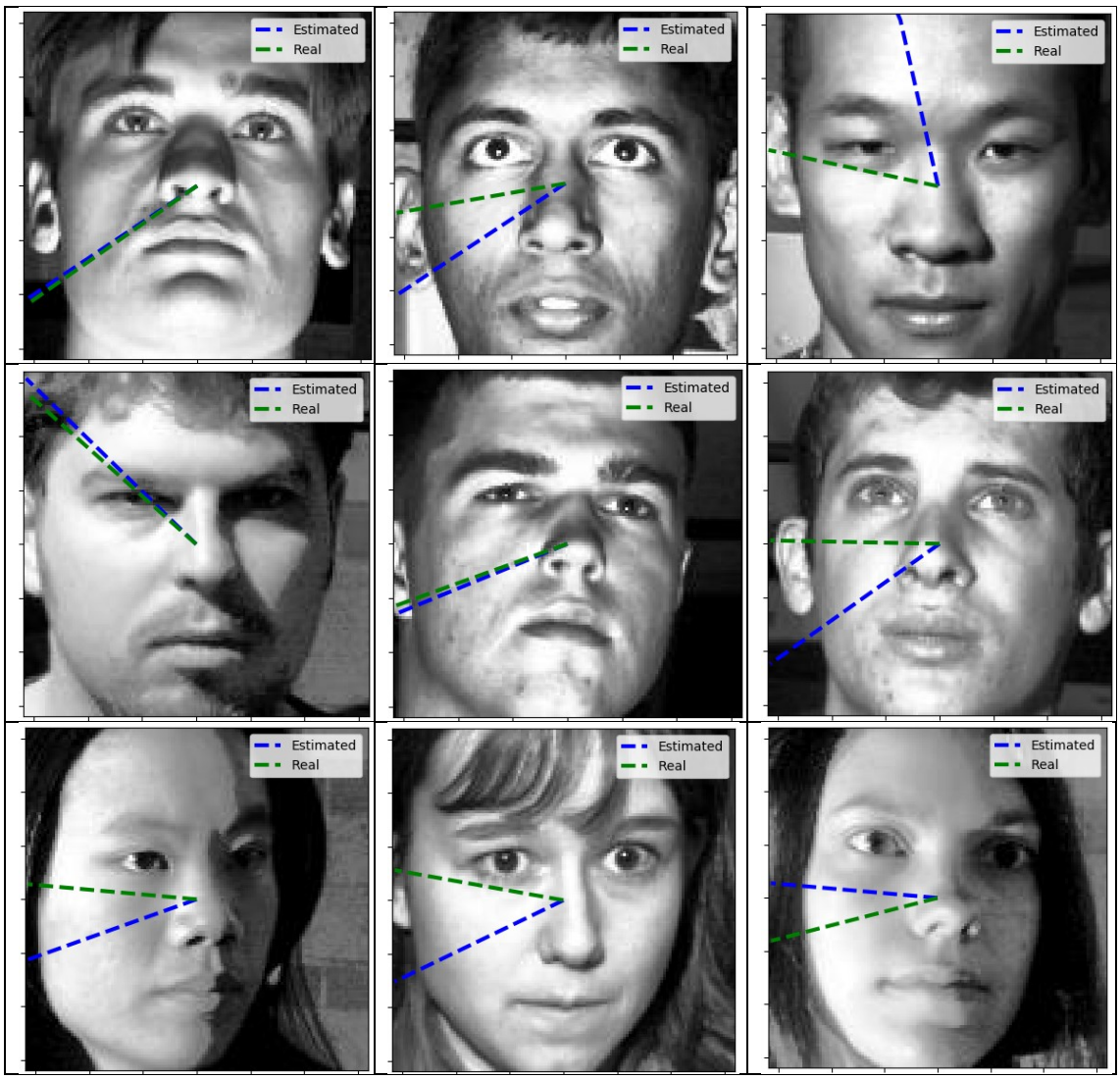


Kuva 26. *Yalen datasetin esimerkkikuvia. Valon suuntakulman estimoitua arvoa kuvataan sinisellä janalla.*

Kuvassa 26 Yalen datasetin kuville on sovellettu kasvojen havainnointialgoritmia, valittu kasvoalue ja syötetty se koulutettuun neuroverkkoon. Tämä kaikki tapahtuu automaattisesti, jolloin neuroverkko palauttaa estimoidun valon suuntakulman ja matplotlib-kirjasto visualisoi sen verkolle syötetyn kuvan päälle. Kuvista on havaittavissa, että malli estimoi valon suuntakulmaa tarkasti.

Satunnaisotoksia Yalen datasetistä visualisoitiin mallin luotettavuuden toteamiseksi. Kuvassa 27 Yalen kasvokuville on laskettu todellista valon suuntakulmaa vastaava arvio sekä mallin estimointialgoritmiin perustuva estimaatti. Todellista valon suuntakulmaa vastaa vihreä jana ja algo-

ritmin tuottamaa estimoitua valon suuntakulmaa vastaa sininen jana. Kuvassa on visualisoitu kasvoja, joiden muoto, pigmentti ja ominaisuudet eroavat toisistaan. Lisäksi kuvissa on huomioitu muun muassa hiusten, kulmakarvojen ja silmien eroavaisuuksia, sillä generoidussa opetusdatassa näitä ominaisuuksia ei ole varioitu.



Kuva 27. *Yalen datasetin kasvokuvia. Sininen jana kuvaa ohjelman estimoimaa valon tulokulmaa ja vihreä jana kuvaa laskettua arvoa todellisesta valon tulokulmasta.*

Kuvasta 27 on havaittavissa mallin Gray 8 tarkkuus Yalen datasetin satunnaisotoksille. Oikean yläkulman kuvassa otsalohko on intensiteetiltään korkea, vaikka valo ei tule suoraan yläpuolelta. Korkean intensiteetin kasvoalueet voivat häiritä mallia tunnistamasta todellista valon suuntaa. Lisäksi alarivillä on havaittavissa, että hiukset voivat aiheuttaa epävarmuutta mallin toiminnassa. Toisaalta tässä tapauksessa kyseessä saattaa olla satunnainen virheiden samankaltaisuus. On oletettavaa, että kasvojen muoto, hiukset ja kasvojen pinta vaikuttavat mallin tarkkuuteen, mutta näitä asioita ei tarkasteltu tässä työssä.

7. ANALYYSI

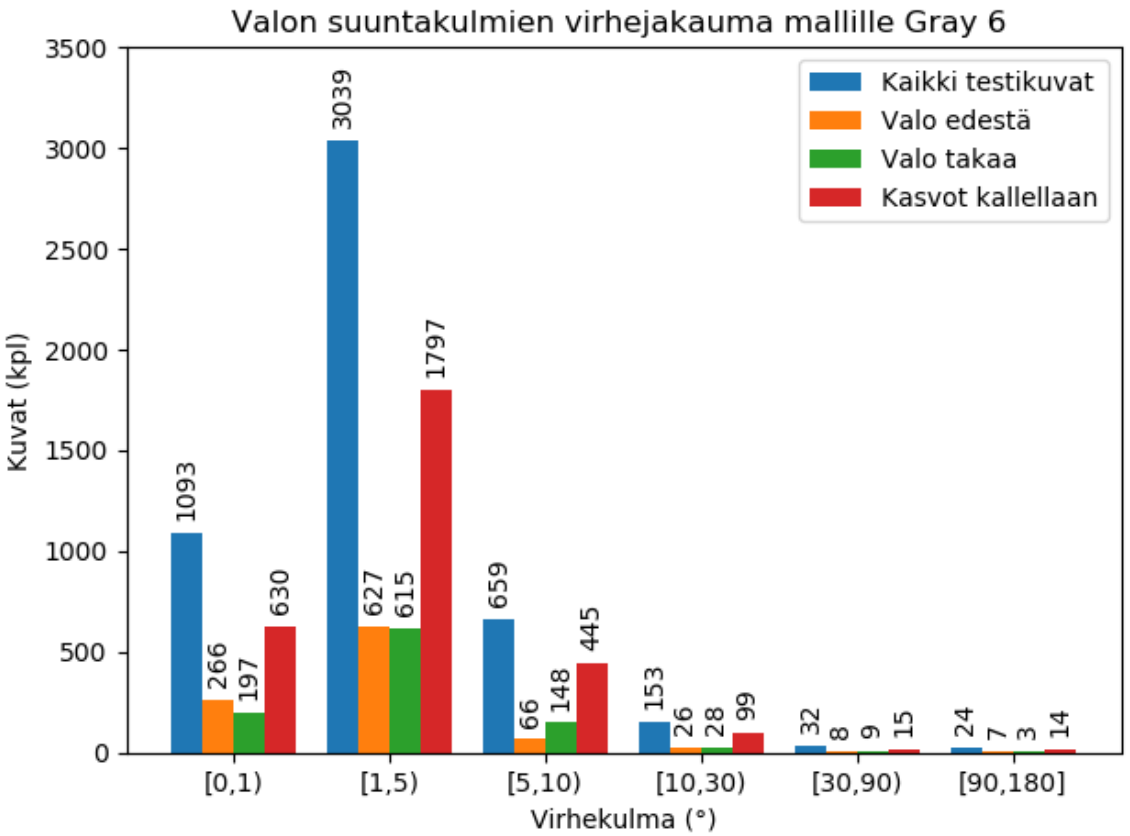
Mallin luotettavuutta voidaan tarkastella luottamusvälin, keskimääräisen virheen ja maksimaalisen virheen mukaan. Lisäksi tilastotieteessä käytettäviä tilastollisen mallinnuksen menetelmiä voidaan hyödyntää. Tilastollinen malli on todennäköisyysjakauma, jonka pohjalta tehdään päätelmiä ja joka muodostuu koulutetun aineiston pohjalta (Davidson, 2003, s. 161–162). Tilastollisia malleja analysoidaan tarkastelemalla muun muassa mallin soveltuvuutta testidatan virheprosentin, virhejakauman sekä virheiden suuruuden mukaan. Tilastollista mallintamista voidaan tehdä esimerkiksi Bayesilaisesta näkökulmasta (Davidson, 2003, s. 565–570).

Tässä työssä malleja tarkasteltiin keskimääräisen virheen keinoin, sillä neuroverkoille oli mahdotonta määrittää luottamusväliä datan ennalta arvaamattomuuden takia. Analyysi pohjautui sekä generoituun että Yalen datasetin testidataan. Testidatan satunnaisotoksia visualisoitiin tuloksia käsittelevässä luvussa. Analyysissä keskityttiin mallin tarkkuuteen laajemmassa mittakaavassa.

Analyysi keskittyi malleihin Gray 6 ja Gray 8, sillä ne erottuivat muista malleista merkittävästi tarkkuuden suhteen. Gray 6 -mallille generoitu testisetti antoi 4,1 asteen keskimääräisen virheen ja Gray 8 -mallille keskimääräinen virhe oli 4,2 astetta. Vaikka generoidulle datalle Gray 8 oli epätarkempi pienellä marginaalilla, suoriutui malli paremmin Yalen datasetin tapauksessa. Eroa näiden harmaasävymallien välillä oli noin 1 aste keskimääräisessä virheessä.

7.1 Virhearviot

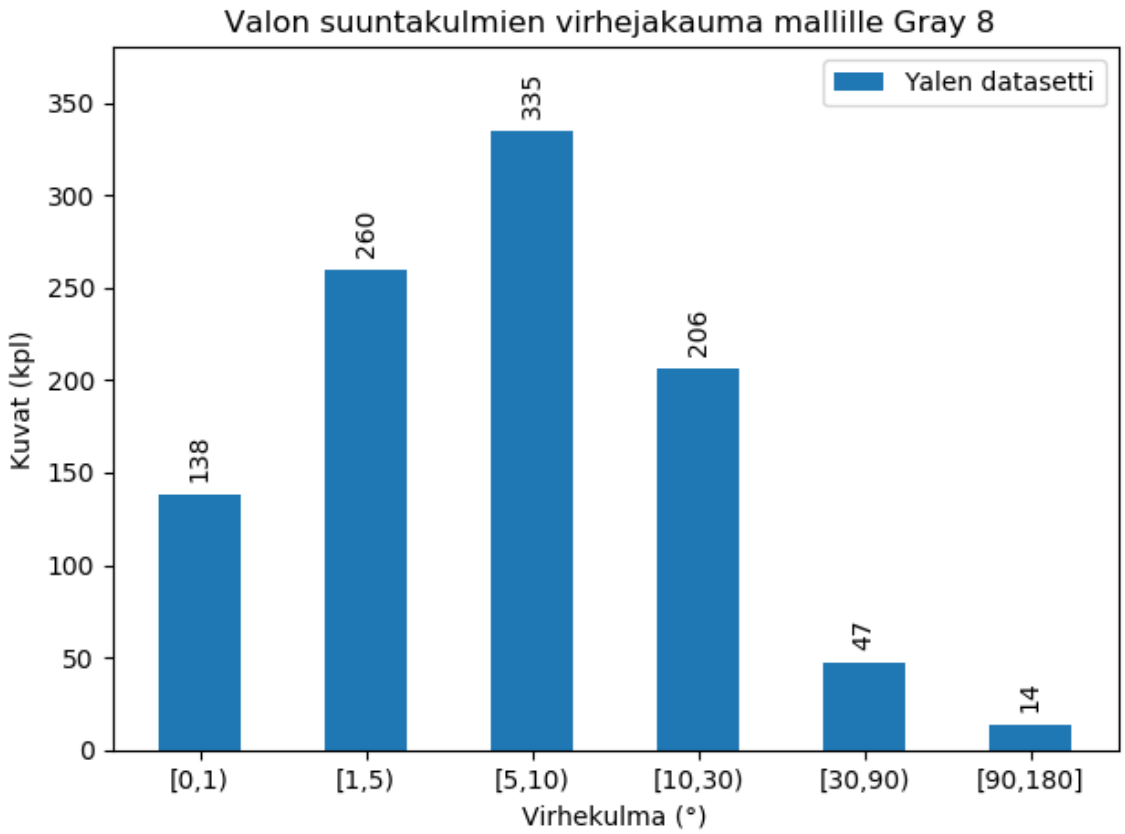
Kuvassa 28 esitetään mallin Gray 6 virhejakauma generoidulle testisetille. Keskimääräinen kulman virhe esitetään asteina pylväsdiagrammissa, jossa siniset pylväät vastaavat koko testiaineistoa. Oranssit, vihreät ja punaiset pylväät kuvaavat datan generoinnissa käytettäviä kombinaatioita kasvojen asennosta sekä valaistuksen syvyyssuunnasta kasvoihin nähden. Y-akseli vastaa kuvamääriä kappaleina ja x-akselilla esitetään absoluuttisen virheen määrä asteväleinä [0,1) astetta, [1,5) astetta, [5,10) astetta, [10,30) astetta, [30,90) astetta ja yli 90 astetta.



Kuva 28. *Tulokset harmaasävykuville mallilla Gray 6. Datasettien virhekulmat on jaoteltu virhealueisiin kasvokuvien kappalemäärien mukaan. Datasetit jakautuvat koko testisettiin ja sen osiin: Suoriin kasvokuviin, joissa valo tulee kasvojen etupuolelta, Suoriin kasvokuviin, joissa valo tulee kasvojen takaa ja Kasvokuviin, joissa kasvot ovat kallellaan valon tullessa mistä tahansa suunnasta.*

Kuvasta 28 on havaittavissa, että suurin osa absoluuttisista virheistä jakautui välille [1, 5) astetta. Lisäksi yli 20 % testidatan kuvien valonlähteiden suuntakulmista osattiin ennustaa asteen tarkkuudella. Huomattavaa oli absoluuttisen virheen vaihtelu jakaumassa, jossa noin 0,5 prosentille tapauksista suuntakulman absoluuttinen virhe oli yli 90 astetta. Tämän vuoksi ei voitu todeta, että mallin estimoiman kulman ja todellisen kulman absoluuttinen erotus oli luottamusväliltään alle 30 astetta. Mallin absoluuttinen virhe generoidulle testidatalle oli kuitenkin erittäin todennäköisesti alle 30 astetta ja todennäköisesti alle 10 astetta. 10 asteen tarkkuudella valon suuntakulman ennustaminen kasvokuvasta ylitti oletettavasti ihmisen keskimääräisen tarkkuuden. Opinnäytetyön tekijä arvioi silmämääräisesti valon suuntakulmaa sadalle generoidulle testikuvalle. Keskimääräinen kulmavirhe oli noin 12 astetta. Testitulos oli vain suuntaa antava otoskoon, testihenkilöiden määrän ja epämääräisen testiympäristön takia.

Mallille Gray 8 suoritettiin testiajo 1 000 Yalen datasetin kasvokuvalle, jossa noudatettiin samoja x- ja y-akseleita kuin kuvassa 28. Jakaumasta muodostui tasaisempi, sillä virheiden suuruus oli keskimäärin suurempi kuin generoidulla datasetillä. Yalen datasetin otoksen jokaisesta kuvasta kasvot havaittiin etukäteen käyttämällä FaceDetect-algoritmia. Kuvassa 29 esitetään mallin Gray 8 absoluuttisen virheen jakauma Yalen datasetin otokselle.



Kuva 29. *Virhekulmien alueellinen jakauma Yalen datasetin kasvokuvien 1000 kappaleen otokselle. Virhekulmien laskemisessa on käytetty mallia Gray 8.*

Kuvan 29 määrältään suurin virheväli oli [5, 10) astetta. Se poikkeaa kuvien 25 ja 28 jakaumasta, sillä generoidulla datalla mallit Gray 6 ja Gray 8 estimoivat valon suuntaa yleisimmin [1, 5) asteen virheellä. Yalen datasetin virhe oli keskimäärin 15,8 astetta, joka sekin poikkeaa generoidun datasetin keskimääräisestä virheestä 4,2 astetta.

7.2 Käyttökohteet

Valittuja malleja voidaan ensisijaisesti käyttää valon suunnan estimoinnissa kuvista, joissa on vähintään yksi kasvokuva. Käyttäminen tapahtuu hyödyntämällä ennustustyökalua, joka automaattisesti havaitsee kasvoalueet, muokkaa kasvoalueiden koon sopivaksi mallille ja tekee esikäsittelyn. Ennustustyökalu käyttää valittua mallia ja palauttaa sekä suuntakulman että visualisoi kulman neuroverkkoon syötetyn kasvokuvan päälle.

Valon suunnan estimoinnille käyttökohteita ovat esimerkiksi auringon suunnan havaitseminen kuvasta, jolloin kuvan suunta on mahdollista laskea koordinaattijärjestelmän ja auringon suunnan avulla. Lisäksi useita kasvoja sisältäviä kuvia voidaan tarkastella. Useita kasvoja sisältävistä kuvista on mahdollista tarkastella kuvan aitoutta, sillä ensisijainen valonlähteen suunta on estimoitavissa jokaiselle havaitulle kasvolle.

Ohjelma on rakennettu komponenteista, jotka ovat hyödynnettävissä osana muita projekteja. Esimerkiksi kasvojen generointiohjelmaa voidaan hyödyntää kasvokuvista koostuvien datasettien luomisessa. Kasvojen havainnointialgoritmia voi käyttää erillisenä komponenttina myös muissa sovelluksissa, kuten ihmiskasvojen datasetin luomisessa, kasvontunnistuksessa tai ilmeiden tunnistuksessa (engl. emotion recognition). Lisäksi datan testi- ja visualisointityökalua on mahdollista käyttää testaamaan ja visualisoimaan tutkimustuloksia myös muissa tutkimustapauksissa. Väriavaruusmuunnoksia tekevät funktiot ovat käytettävissä muissa värimuunnoksia hyödyntävissä ohjelmissa omina funktioinaan.

7.3 Kriittinen analyysi

Kriittisesti tarkasteltuna kuvan 29 perusteella voidaan sanoa, että ohjelma soveltuu vain demonstraatiokäyttöön. Virhealttius käytännölliselle testidatalle on sellaisella tasolla, että ohjelman käyttäminen ihmisen läsnä ollessa tehostaa valon suuntakulman estimointia, mutta mallia ei automaattisesti voida hyödyntää suurille datamäärille ilman ihmisen asiantuntemusta. Vaikka mallin virhe on alle 10 astetta 70 %:lle Yalen datasetin testitapauksista, selkeää luottamusväliä ei voida antaa.

Ohjelmaa ei ole sertifioitu. Ohjelman laatu ei välttämättä vastaa ISO 9000 -standardeja laadunvarmistuksessa (engl. quality assurance), sillä luottamusta ei synny välttämättä organisaatiosta ulospäin. Ohjelma on kuitenkin toteutettu helposti ylläpidettäväksi ja koostuu komponenteista, joita voi yksittäin muuttaa. Täten ohjelma noudattaa pääasiassa ISO 9126 -standardia, joka on ohjelmiston laadun evaluointiin perustuva standardi. Myös ISO 25010 -standardia noudatetaan pääosin – laadunvarmistuksen jäädessä avoinna olevaksi kysymykseksi.

Projektin ohjelmoija on kokenut Python-ohjelmoija, jolla on kokemusta lukuisista ohjelmakokonaisuuksista useiden vuosien ajalta. Ohjelmoijan taso on ammattimainen, mutta hän ei ollut huipputasoa. Projektin vastuuhenkilönä toimii korkeasti koulutettu esimies, jolla on kokemusta sekä projektinhallinnasta että nykyaikaisista teknologioista.

Työ oli kokonaisuudessaan vaativa, mutta se oli täysin toteutettavissa. Kokonaisuuden tiukka aikataulu sekä ohjelmoijan kokemattomuus projektinhallinnassa heikensi ohjelmiston kokonaislaatua. Projektin taso oli silti korkea diplomityön tasoiseen kokonaisuuteen, jossa tuloksia esitettiin ja analysoitiin kattavasti.

Aiempiin tutkimuksiin työtä ei verrattu, sillä ilman muiden sovellusten testaamista puolueetonta näkemystä ei voitu muodostaa. Ongelmaksi vertailussa muodostui ohjelmien saatavuus ja tutkimusten vähyys samankaltaisessa kontekstissa. Jotta tulevaisuudessa vastaavaa asiaa tutkivat asiantuntijat saavat vertailukohdan, on ohjelma saatavissa avoimena lähdekoodina tutkimuskäyttöön osoitteessa [2] (Ijäs, 2019).

Konvoluutionaaliset neuroverkot soveltuivat menetelmänä kuvadatan analysointiin. Huipputason tutkimukset vuodesta 2012 (Krizhevsky et al. 2012) alkaen ovat hyödyntäneet konvoluutiokerroksia sekä lukuisia optimointimenetelmiä, kuten satunnaiskarsintaa ja eräkoon normalisointia. Samoja tutkimuksissa esiteltyjä menetelmiä käytettiin myös tässä työssä. Työn rajoitusten takia konvoluutionaalisten neuroverkkojen täyttä potentiaalia ei saavutettu tämän projektin aikana, mutta jatkokehityksen avulla ohjelma on mahdollista ottaa käyttöön sille soveltuviin käyttökohteisiin.

7.4 Juridiset rajoitteet

Projektin aikana juridiset rajoitteet nousivat esiin lukuisia kertoja. Useimmat potentiaaliset datasetit eivät noudattaneet GDPR:ä tai eivät sallineet datan käyttämistä muuhun kuin tutkimuskäyttöön. Projektin osana tehtiin ohjelmisto, jota voitiin käyttää muussakin kuin tutkimuskontekstissa. Täten datasettejä ei käytetty opetuksessa tai testauksessa. Lisäksi sivustolta Flickr olisi ollut mahdollista hakea kasvokuvia, mutta niitä suojaa Creative Commons -lisenssit. Creative Commons -lisenssit voivat estää kaupallisen käytön tai muutokset kuviin, jolloin niitä ei saa hyvän etiikan mukaan käyttää.

Henkilön tietosuojalaki 1050/2018 täsmentää luonnollisten henkilöiden suojelusta henkilötietojen käsittelyssä. Täten lupien käsittelystä tulisi työlästä. Lisäksi EU:n yleisen tietosuoja-asetuksen GDPR:n sekä EU:n sähköisen viestinnän tietosuojadirektiivin (58/2002/EY) perusteella tietojen keräämiseen, kuten kasvokuvan tallentamiseen, tarvitaan selkeästi ilmaistu suostumus. Yksityisyydensuoja rajoittaa itseään koskevan tiedon kontrollointia, kuten kuvan käyttämistä ja jakamista.

[2] https://github.com/NemoHostem/CNN_Source_of_Light

Kuvien käyttöoikeudet ovat usein suojattu Creative Commons -lisenssillä, joka saattaa rajoittaa niiden käyttöä. Lähtökohtaisesti valokuvat ovat tekijänoikeudellisesti suojattuja. Valokuvan käyttöoikeus on kuvan ottajalla, ei kuvan kohteella. Kuitenkin mikäli valokuvasta voidaan tunnistaa henkilöitä, on kaikilta tunnistettavilta henkilöiltä saatava lupa kuvan käyttöön.

Suurin osa kasvokuvia sisältävistä dataseteistä oli muualta kuin Euroopasta, ja niissä ei ollut mainintaa kuvattavien suostumuksesta, jolloin niiden käyttäminen olisi ollut epäeettistä ja lainvastaista. Kasvokuvien henkilöt olivat tunnistettavissa, joten heidän suostumuksensa olisi pitänyt olla saatavilla myös muille datasetin käyttäjille.

Ohjelma on avoimesti saatavissa (Ijäs, 2019). Ohjelma on avointa lähdekoodia, joten sitä saa käyttää vapaasti tutkimuksessa. Muussa tapauksessa ohjelman käyttö- ja muokkausoikeudet säilyvät Puolustusvoimien tutkimuslaitoksella. Esimerkiksi ohjelman käyttäminen osana maksullista sisältöä tulee sopia Puolustusvoimien tutkimuslaitoksen kanssa erikseen. Ohjelman tai sen osien käyttämisessä tulee noudattaa hyvää etiikkaa ja moraalia. Projektin ohjelmoija ei ota vastuuta avoimen lähdekoodin toimivuudesta, käyttökohteista tai ylläpidosta.

8. YHTEENVETO

Tämän projektin tarkoituksena oli kehittää ohjelmisto, jonka avulla ensisijaisen valonlähteen suuntakulma olisi estimoitavissa. Teknisen osuuden ympärille muodostettiin tieteellinen tutkimus koneoppimisen aihealueelle, jossa perehdyttiin neuroverkkoihin ja vertailtiin teknisen prosessin malleja testidatan osalta. Lopputuloksena muodostui projekti, joka kattoi sekä teknisen että tieteellisen osuuden.

Tieteellisessä osuudessa perehdyttiin valon suunnan estimointiin soveltuviin tekniikoihin. Tutkimuksessa perehdyttiin syväsuuntaavasti neuroverkkoihin ja konvoluutionaalisiin neuroverkkoihin, jotka ovat viimeisen kymmenen vuoden aikana olleet trendinä tietoteknisissä tutkimuksissa. Kirjallisuudessa valon suunnan estimoinnissa oli käytetty muun muassa intensiteettiin ja luminanssiin perustuvia menetelmiä sekä valon suuntaa oli tarkasteltu varjojen ja niitä vastaavien objektien avulla.

Neuroverkkojen opetusta varten generoitiin kasvokuvia Face3D-algoritmilla. Generoituihin kasvokuviin sovellettiin digitaalista kuvankäsittelyä, jonka avulla kuvien väriavaruuksia ja -malleja muunnettiin suodatettuun RGB-muotoon sekä harmaasävykuviksi. Neuroverkon syötteen vaikutusta mallin tarkkuuteen pystyttiin analysoimaan kuvankäsittelyn avulla. Valonlähteen suuntakulmien estimoinnissa käytettiin FaceDetect-algoritmia kasvojen havainnointiin, jotta neuroverkon syöte olisi samantapainen sekä opetus- että testiaineistolle.

Teknisessä osuudessa luotiin lukuisia malleja, joissa syötteenä käytettiin erimuotoista ja -kokoista dataa. Mallit jaottuivat luokkapohjaisiin ja regressiivisiin malleihin. Lisäksi niiden rakenteissa ja optimointiparametreissa oli eroavaisuuksia. Tarkimmaksi malliksi valikoitui harmaasävykuvia syötteenä käyttävä regressiivinen malli. Se kehitettiin Keras-kirjastolla, joka koostui viidestä konvoluutiokerroksesta sekä kolmesta täysin yhdistetystä kerroksesta sisältäen yhteensä 490 000 parametria. Tarkimmalle mallille käytettiin ReLU-aktivointifunktiota ja eräkoon normalisointia useassa konvoluutiokerroksessa sekä satunnaiskarsintaa täysin yhdistetyissä kerroksissa.

Mallin toiminnallisessa tarkastelussa käytettiin testiohjelmaa, joka arvioi testikuvien virhettä ja visualisoi esimerkkikuvia automaattisesti. Tarkin malli antoi generoidulle testisetille keskimääräiseksi virheeksi 4,2 astetta ja Yalen datasetille 15,8 astetta. Malli suoriutui generoidulle testisetille pääasiassa hyvin, sillä yli 93 % testikuvien valon suuntakulmista estimoitiin 10 asteen tarkkuudella. Yalen datasetille vastaava arvo oli noin 72 %. Malli sisälsi kuitenkin epätarkkuuksia, sillä noin 0,5–1,5 % kasvokuvien valon suunnista estimoitiin yli 90 astetta väärin. Malli soveltui tällaisenaan työn demonstrointiin ja jatkokehityksen avulla siitä olisi mahdollista tehdä käyttöympäristöön sopiva analysointityökalu.

Ohjelman käyttökohteita ovat ensisijaisen valonlähteen suunnan estimointi sisä- ja ulkotiloissa sekä muokattujen kuvien tunnistaminen ihmisen avustuksella. Ohjelman kouluttaminen perustuu kasvokuvista koostuvaan aineistoon. Sen testaaminen ja toiminta keskittyy ainoastaan kasvokuvien ympärille. Tulevaisuudessa ohjelmistoa voisi kehittää tunnistamaan pääasiallisen valonlähteen suunnan mistä tahansa valokuvasta. Lisäksi valonlähteitä voisi tunnistaa useampia, eikä ohjelma rajoittuisi pelkästään ensisijaiseen valonlähteeseen.

LÄHTEET

M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, et al., Tensorflow: Large-scale machine learning on heterogeneous distributed systems, arXiv preprint arXiv:1603.04467, 2016. Saatavissa: https://arxiv.org/pdf/1603.04467.pdf, Noudettu: 17.09.2020

J.L. Ba, J.R. Kiros, G.E. Hinton, Layer normalization, arXiv preprint arXiv:1607.06450, 2016. Saatavissa: https://arxiv.org/pdf/1607.06450.pdf, Noudettu: 03.09.2020

D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473, 2014. Saatavissa: https://arxiv.org/pdf/1409.0473.pdf, Noudettu: 07.09.2020

P.L. Bartlett, For valid generalization the size of the weights is more important than the size of the network. In Advances in neural information processing systems, pp. 134-140, 1997. Saatavissa: http://papers.nips.cc/paper/1204-for-valid-generalization-the-size-of-the-weights-is-more-important-than-the-size-of-the-network.pdf, Noudettu: 02.09.2020

P.L. Bartlett, S. Mendelson, Rademacher and Gaussian complexities: Risk bounds and structural results. Journal of Machine Learning Research Vol. 3, No. Nov 2002, pp. 463–482, 2002. Saatavissa: https://www.jmlr.org/papers/volume3/bartlett02a/bartlett02a.pdf, Noudettu: 02.09.2020

BBC News, Artificial Intelligence: Google's AlphaGo beats Go master Lee Se-dol, BBC News, March 12, 2016, Saatavissa: https://www.bbc.com/news/technology-35785875, Noudettu: 28.04.2020

H.D. Beale, H.B. Demuth, M.T. Hagan, Neural network design, Pws, Boston, 1996, Saatavissa: https://www.semanticscholar.org/paper/Neural-Network-Design-Hagan-Demuth/6a8122861b80842ee6f406acdeec35aec913f1b8, Noudettu: 25.05.2020

Y. Bengio, R. Ducharme, P. Vincent, C. Jauvin, A neural probabilistic language model, Journal of machine learning research, Vol. 3, Feb 2003, pp. 1137-1155. Saatavissa: http://papers.nips.cc/paper/1839-a-neural-probabilistic-language-model.pdf, Noudettu: 05.09.2020

J. Bergstra, R. Bardenet, Y. Bengio, B. Kégl, Algorithms for hyper-parameter optimization. In Advances in neural information processing systems, pp. 2546-2554, 2011. Saatavissa: http://papers.nips.cc/paper/4443-algorithms-for-hyper-parameter-optimization.pdf, Noudettu: 02.09.2020

C.M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006, 738 p. Saatavissa: https://cds.cern.ch/record/998831, Noudettu: 14.01.2020

G. Bradski, The OpenCV Library, Dr Dobb's Journal of Software Tools, Vol. 25, pp. 120-125. 2000. Saatavissa: https://github.com/skvark/opencv-python, Noudettu: 17.09.2020

G. Bradski, A. Kaehler, Learning OpenCV: Computer vision with the OpenCV library, “O’Reilly Media, Inc.”, 571 p. 2008, Saatavissa: https://www.bogotobogo.com/cplusplus/files/OReilly%20Learning%20OpenCV.pdf, Noudettu: 18.09.2020

A. Braun, Chatbots in der Kundenkommunikation, Springer-Verlag, 2013, Saatavissa: https://link.springer.com/book/10.1007%2F978-3-642-19021-6, Noudettu: 13.03.2020

R. Brunelli, Estimation of pose and illuminant direction for face processing, Image and Vision Computing, Vol. 15, No. 10, pp. 741-748. 1997, Saatavissa: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.48.5411&rep=rep1&type=pdf, Noudettu: 15.05.2020

M. Bunge, A general Black Box Theory, Philosophy of Science 30, No. 4, 1963, pp. 346-358. Saatavissa: https://doi.org/10.1086/287954, Noudettu: 09.05.2020

C. Cao, Q. Hou, K. Zhou, Displaced dynamic expression regression for real-time facial tracking and animation, ACM Transactions on graphics (TOG), Vol. 33, No. 4, pp. 1-10. 2014. Saatavissa: https://dl.acm.org/doi/pdf/10.1145/2601097.2601204, Noudettu: 14.09.2020

W. Chen, D. Xiang, J. Deng, Surface normals in the wild, In Proceedings of the IEEE International Conference on Computer Vision, pp. 1557-1566. 2017. Saatavissa: https://openaccess.thecvf.com/content_ICCV_2017/papers/Chen_Surface_Normals_in_ICCV_2017_paper.pdf, Noudettu: 12.07.2019

M.M. Cheng, N.J. Mitra, X. Huang, P.H. Torr, S.M. Hu, Global contrast based salient region detection, IEEE transactions on pattern analysis and machine intelligence, Vol. 37, No. 3, pp. 569-582. 2014. Saatavissa: https://ieeexplore.ieee.org/document/6871397, Noudettu: 12.09.2020

K. Cho, B. Van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078, 2014. Saatavissa: https://arxiv.org/pdf/1406.1078.pdf, Noudettu: 07.09.2020

S.I. Choi, C.H. Choi, N. Kwak, Face recognition based on 2D images under illumination and pose variations, Pattern Recognition Letters, Vol. 32, No. 4, pp. 561-571. 2011. Saatavissa: http://mipal.snu.ac.kr/images/c/ce/Pose_illum_PRL.pdf, Noudettu: 15.09.2020

F. Chollet, et al., Keras, Github, 2015, Saatavissa: https://github.com/fchollet/keras, Noudettu: 17.09.2020

A. Choromanska, M. Henaff, M. Mathieu, G.B. Arous, Y. LeCun, The loss surfaces of multilayer networks. In Artificial Intelligence and Statistics, pp. 192-204, 2015. Saatavissa: http://proceedings.mlr.press/v38/choromanska15.pdf, Noudettu: 03.09.2020

D.C Ciresan, U. Meier, J. Masci, L.M. Gambardella, J. Schmidhuber, Flexible, high performance convolutional neural networks for image classification. In Twenty-second international joint conference on artificial intelligence. 2011. Saatavissa: http://people.idsia.ch/~juergen/ijcai2011.pdf, Noudettu: 02.06.2020

J. Clark, Why 2015 Was a Breakthrough Year in Artificial Intelligence, Bloomberg News, Saatavissa: https://www.bloomberg.com/news/articles/2015-12-08/why-2015-was-a-breakthrough-year-in-artificial-intelligence, Noudettu: 13.05.2020

R. Collobert, S. Bengio, Links between perceptrons, MLPs and SVMs. In Proceedings of the twenty-first international conference on Machine learning, p. 23. 2004. Saatavissa: https://dl.acm.org/doi/pdf/10.1145/1015330.1015415?casa_token=hxX30ldN7v4AAAAA:fXvoDSWMt9q2nr0lmfOVZB5sBPRl7PFOw2VEQpp-DywqnDhjyGimmp-uGYEoGmbnOet5ai6zg21NdNg, Noudettu: 01.09.2020

Council Regulation (EU) 2016/679 (General Data Protection Regulation), Official Journal of the European Union, L 119, 2016, Saatavissa: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679, Noudettu: 25.05.2020

D. Crevier, AI: The Tumultuous History of the Search for Artificial Intelligence, Basic Books, Inc. 1996, Noudettu: 20.01.2020

G. Cybenko, Approximation by superposition of sigmoidal functions, Mathematics of Control, Signals and Systems, No. 2, Vol. 4, pp. 303–314, 1989. Saatavissa: https://web.eecs.umich.edu/~cscott/smlrg/approx_by_superposition.pdf, Noudettu: 01.09.2020

B. Davari, W.H. Chang, M.R. Wordeman et al. A high performance 0.25 mu m CMOS technology, Technical Digest, International Electron Devices Meeting, San Francisco, CA, USA, 1988, pp. 56-59, Saatavissa: https://ieeexplore.ieee.org/document/32749, Noudettu: 25.04.2020

A.C. Davidson, Statistical Models, Cambridge University Press, 726 p. 2003. Saatavissa: https://books.google.fi/books?id=gQyIGGAiN4AC, Noudettu: 28.09.2020

M. Denil, B. Shakibi, L. Dinh, N. de Freitas, et al. Predicting parameters in deep learning, Advances in Neural Information Processing Systems, pp. 2148–2156, 2013, Saatavissa: https://papers.nips.cc/paper/5025-predicting-parameters-in-deep-learning.pdf, Noudettu: 26.06.2019

T. DeVries, G.W. Taylor, Improved regularization of convolutional neural networks with cutout. arXiv preprintarXiv:1708.04552, 2017. Saatavissa: https://arxiv.org/pdf/1708.04552.pdf, Noudettu: 12.05.2020

J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, Decaf: A deep convolutional activation feature for generic visual recognition. In International Conference on Machine Learning, Vol. 32, pp. 647–655, 2014. Saatavissa: http://proceedings.mlr.press/v32/donahue14.pdf, Noudettu: 08.09.2020

R. Eldan, O. Shamir, The power of depth for feedforward neural networks, Conference on learning theory, 2016, pp. 907-940. Saatavissa: http://proceedings.mlr.press/v49/eldan16.pdf, Noudettu: 27.05.2020

M. Eskelinen, Värien teoria ja värimallit, Tietokonegrafiikan seminaari kevät 2002, 2002, 24 p. Saatavissa: http://users.jyu.fi/~tro/gtksem02/prujut/matti/varimallit.pdf, Noudettu: 10.09.2020

M.D. Fairchild, Color appearance models, John Wiley & Sons, 2013, 472 p. Saatavissa: https://books.google.fi/books?hl=en&lr=&id=hh3lXMJ4bDIC, Noudettu: 10.05.2020

B.W.A.C. Farley, W. Clark, Simulation of self-organizing systems by digital computer. Transactions of the IRE Professional Group on Information Theory, Vol. 4, No. 4, 1954, pp. 76-84. Saatavissa: https://ieeexplore.ieee.org/document/1057468, Noudettu: 24.05.2020

Y. Feng, Face3d: Python tools for processing 3D face, Github, Saatavissa: https://github.com/YadiraF/face3d, Noudettu: 21.07.2019

W. Gao, B. Cao, S. Shan, X. Chen, D. Zhou, X. Zhang, D. Zhao, The CAS-PEAL large-scale Chinese face database and baseline evaluations, IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, Vol. 38, No. 1, pp. 149-161. 2007. Saatavissa: http://www.jdl.ac.cn/peal/files/IEEE_SMC_A_gao_CAS-PEAL.pdf, Noudettu: 14.09.2020

A.S. Georghiades, P.N. Belhumeur, D.J. Kriegman, From few to many: Illumination cone models for face recognition under variable lighting and pose, IEEE transactions on pattern analysis and machine intelligence, Vol. 23, No. 6, pp. 643-660. 2001. Saatavissa: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.127.3126&rep=rep1&type=pdf, Noudettu: 14.05.2020

L.H. Gilpin, D. Baun, B.Z. Yuan, A. Bajwa, M. Specter, L. Kagal, Explaining Explanations: An Overview of Interpretability of Machine Learning, IEEE 5th International Conference on data science and advanced analytics (DSAA), IEEE, 2018, pp. 80-89. Saatavissa: https://arxiv.org/pdf/1806.00069.pdf, Noudettu: 20.05.2020

R. Girshick, Fast r-cnn. Proceedings of the IEEE international conference on computer vision, pp. 1440-1448. 2015, Saatavissa: https://openaccess.thecvf.com/content_iccv_2015/papers/Girshick_Fast_R-CNN_ICCV_2015_paper.pdf, Noudettu: 25.05.2020

X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the thirteenth international conference on Artificial Intelligence and Statistics, 2010, pp. 249-256. Saatavissa: http://proceedings.mlr.press/v9/glorot10a/glorot10a.pdf, Noudettu: 31.08.2020

B. Goertzel, J. Mossbridge, E. Monroe, D. Hanson, G. Yu, Loving AI: Humanoid Robots as Agents of Human Consciousness Expansion (summary of early research progress). Cornell University Laval, 2017, pp. 1–16. Saatavissa: https://arxiv.org/pdf/1709.07791.pdf, Noudettu: 22.05.2020

S. Goferman, L. Zelnik-Manor, A. Tal, Context-aware saliency detection, IEEE transactions on pattern analysis and machine intelligence, Vol. 34, No. 10, pp. 1915-1926. 2011. Saatavissa: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.309.9134&rep=rep1&type=pdf, Noudettu: 13.09.2020

R.C. Gonzales, R.E. Woods, Digital image processing, 3rd edition, Pearson, 2007, 976 p. Saatavissa: http://sdeuoc.ac.in/sites/default/files/sde_videos/Digital%20Image%20Processing%203rd%20ed.%20-%20R.%20Gonzalez%2C%20R.%20Woods-ilovepdf-compressed.pdf, Noudettu: 09.09.2020

I. Goodfellow, Y. Bengio, A. Courville, Deep learning, MIT press, 2016. Saatavissa: https://www.deeplearningbook.org, Noudettu: 07.09.2020

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, In Advances in neural information

processing systems, pp. 2672-2680. 2014. Saatavissa: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf, Noudettu: 15.09.2020

I. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, Y. Bengio, Maxout Networks, In International conference on Machine Learning, 2013, pp. 1319-1327. PMLR, Saatavissa: https://arxiv.org/pdf/1302.4389v4.pdf, Noudettu: 01.09.2020

M. Greshko, “Meet Sophia, the Robot That Looks Almost Human”, National Geographic, May 18, 2018, Saatavissa: https://www.nationalgeographic.com/photography/proof/2018/05/sophia-robot-artificial-intelligence-science/, Noudettu: 22.05.2020

R. Guo, Q. Dai, D. Hoiem, Single-image shadow detection and removal using paired regions, IEEE, CVPR 2011, 2011, pp. 2033-2040, Saatavissa: http://dhoiem.web.engr.illinois.edu/publications/cvpr11_shadow.pdf, Noudettu: 23.07.2019

K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778, 2016. Saatavissa: https://openaccess.thecvf.com/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf, Noudettu: 02.09.2020

T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, M. Li, Bag of tricks for image classification with convolutional neural networks, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 558-567. 2019. Saatavissa: https://openaccess.thecvf.com/content_CVPR_2019/papers/He_Bag_of_Tricks_for_Image_Classification_with_Convolutional_Neural_Networks_CVPR_2019_paper.pdf, Noudettu: 21.07.2019

D.O. Hebb, The organization of behavior: A neuropsychological theory, Psychology Press, 2005, 365 p. Saatavissa: https://pure.mpg.de/rest/items/item_2346268/component/file_2346267/content, Noudettu: 23.05.2020

W.R. Hendee, E.R. Ritenour: *Medical Imaging Physics*, John Wiley & Sons, 2003. ISBN 9780471461135. Noudettu: 28.06.2019

M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, S. Hochreiter, GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In Advances in Neural Information Processing Systems 30 (NIPS 2017), 2017, Saatavissa: https://papers.nips.cc/paper/7240-gans-trained-by-a-two-time-scale-update-rule-converge-to-a-local-nash-equilibrium.pdf, Noudettu: 31.08.2020

G. Hinton, L. Deng, D. Yu, G.E. Dahl, A.R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, B. Kingsbury, Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal Processing Magazine, Vol. 29, No. 6, pp. 82-97, 2012. Saatavissa: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/HintonDengYuEtAl-SPM2012.pdf, Noudettu: 05.09.2020

S. Hochreiter, J. Schmidhuber. Long short-term memory. Neural Computation, Vol. 9, No. 8, pp. 1735-1780, 1997. Saatavissa: https://www.bioinf.jku.at/publications/older/2604.pdf, Noudettu: 02.09.2020

A. Hodges, Alan Turing: The Enigma: The Enigma, Random House, 2012, 768 p. Saatavissa: https://books.google.fi/books?hl=en&lr=&id=EpAl0piM38cC, Noudettu: 17.01.2020

A. Holzinger, M. Plass, K. Holzinger, G.C. Crisan, C.M. Pintea, V. Palade, A glass-box interactive machine learning approach for solving NP-hard problems with the human-in-the-loop, 2017, Saatavissa: https://arxiv.org/pdf/1708.01104.pdf, Noudettu: 22.05.2020

B.K.P. Horn, Understanding image intensities, Artificial intelligence, Vol. 8, No. 2 pp. 201-231. 1977, Saatavissa: https://www.researchgate.net/profile/Berthold_Horn/publication/222440208_Understanding_Image_Intensities/links/5a7c756c0f7e9b477a02c791/Understanding-Image-Intensities.pdf, Noudettu: 12.04.2020

G. Huang, Z. Liu, K.Q. Weinberger, L. Van Der Maaten, Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700-4708, 2017. Saatavissa: https://openaccess.thecvf.com/content_cvpr_2017/papers/Huang_Densely_Connected_Convolutional_CVPR_2017_paper.pdf, Noudettu: 08.09.2020

G. Huang, Y. Sun, Z. Liu, D. Sedra, K.Q. Weinberger, Deep networks with stochastic depth, In European conference on computer vision, Springer, Cham, pp. 646–661, 2016. Saatavissa: https://arxiv.org/pdf/1603.09382.pdf, Noudettu: 08.09.2020

J.D. Hunter, Matplotlib: A 2D graphics environment, Computing in science & engineering, Vol. 9, No. 3, pp. 90-95. 2007. Saatavissa: https://ieeexplore.ieee.org/document/4160265, Noudettu: 16.09.2020

M. Ijäs, CNN Source of Light, Github, Saatavissa: https://github.com/NemoHostem/CNN_Source_of_Light, Noudettu: 13.09.2020

S. Ioffe, C. Szegedy, Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, arXiv preprint arXiv:1502.03167, 2015. Saatavissa: https://arxiv.org/pdf/1502.03167.pdf, Noudettu: 03.09.2020

A.G. Ivakhnenko, V.G. Lapa, Cybernetics and forecasting techniques, 1967, Saatavissa: https://books.google.fi/books?id=rGFgAAAAMAAJ, Noudettu: 25.05.2020

A.S. Jackson, A. Bulat, V. Argyriou, G. Tzimiropoulos, Large pose 3D face reconstruction from a single image via direct volumetric CNN regression, In Proceedings of the IEEE International Conference on Computer Vision, pp. 1031-1039. 2017. Satavissa: https://openaccess.thecvf.com/content_ICCV_2017/papers/Jackson_Large_Pose_3D_ICCV_2017_paper.pdf, Noudettu: 19.09.2020

L.P. Kaelbling, M.L. Littman, A.W. Moore, Reinforcement Learning: A Survey, Journal of Artificial Intelligence Research, Vol. 4, 1996, pp. 237-285, Saatavissa: https://arxiv.org/pdf/cs/9605103.pdf, Noudettu: 12.07.2019

J. Kahn, Accenture Unveils Tool to Help Companies Insure Their AI Is Fair, Bloomberg & Company, June 13, 2018, Saatavissa: https://www.bloomberg.com/news/articles/2018-06-13/accenture-unveils-tool-to-help-companies-insure-their-ai-is-fair, Noudettu: 13.05.2020

N. Kalchbrenner, E. Grefenstette, P. Blunsom, A convolutional neural network for modelling sentences. arXiv preprint arXiv:1404.2188, 2014. Saatavissa: https://arxiv.org/pdf/1404.2188.pdf, Noudettu: 07.09.2020

J. Kannala, E. Rahtu, Bsif: Binarized statistical image features. In Proceedings of the 21st international conference on pattern recognition (ICPR2012), pp. 1363-1366. IEEE, 2012. Saatavissa: http://www.ee.oulu.fi/~jkannala/publications/icpr2012a.pdf, Noudettu: 11.09.2020

B. Karlik, A.V. Olgac, Performance analysis of various activation functions in generalized MLP architectures of neural networks, International Journal of Artificial Intelligence and Expert Systems, Vol. 1, No. 4, 2011, pp. 111-122. Saatavissa: http://www.cscjournals.org/manuscript/Journals/IJAE/Volume1/Issue4/IJAE-26.pdf, Noudettu: 25.05.2020

D.P. Kingma, J. Ba, Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2015. Saatavissa: https://arxiv.org/pdf/1412.6980.pdf, Noudettu: 12.05.2020

S.C. Kleene, Representation of events in nerve nets and finite automata (No. RAND-RM-704). RAND PROJECT AIR FORCE SANTA MONICA CA. 1951, Saatavissa: https://apps.dtic.mil/dtic/tr/fulltext/u2/a596138.pdf, Noudettu: 22.05.2020

A. Klein, et al., imageio/imageio: V2.5.0 (Version v2.5.0). Zenodo. 2019, Saatavissa: http://doi.org/10.5281/zenodo.2558175, Noudettu: 18.09.2020

A. Krizhevsky, Learning multiple layers of features from tiny images, Technical report, University of Toronto, 60 p. 2009. Saatavissa: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.222.9220&rep=rep1&type=pdf, Noudettu: 07.09.2020

A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Advances in neural information processing systems, 2012, pp. 1097-1105, Saatavissa: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf, Noudettu: 23.07.2019

J.R. Koza, F.H. Bennett III, D. Andre, M.A. Keane, Automated Design of Both the Topology and Sizing of Analog Electrical Circuits Using Genetic Programming, Artificial Intelligence in Design'96, Springer, Dordrecht, 1996, pp. 151-170. Saatavissa: https://link.springer.com/chapter/10.1007/978-94-009-0279-4_9, Noudettu: 23.03.2020

S. Laine, T. Karras, T. Aila, A. Herva, S. Saito, R. Yu, H. Li, J. Lehtinen, Production-level facial performance capture using deep convolutional neural networks, In Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 1-10. 2017. Saatavissa: https://users.aalto.fi/~laines9/publications/laine2017sca_paper.pdf, Noudettu: 14.03.2020

G. Larsson, M. Maire, G. Shakhnarovich, Fractalnet: Ultra-deep neural networks without residuals, arXiv preprint arXiv:1605.07648, 2016. Saatavissa: https://arxiv.org/pdf/1605.07648.pdf, Noudettu: 02.09.2020

M. Laskowski, Detection of light sources in digital photographs, 11th Central European Seminar on Computer Graphics, 2007, Saatavissa: https://old.cescg.org/CESCG-2007/papers/Szczecin-Laskowski-Maciej.pdf, Noudettu: 21.06.2019

S. Lawrence, C.L. Giles, A.C. Tsoi, A.D. Back, Face recognition: A convolutional neural-network approach. IEEE transactions on neural networks, Vol. 8, No. 1, pp. 98-

113. 1997, Saatavissa: http://www.cs.cmu.edu/afs/cs/user/bhiksha/WWW/courses/deeplearning/Fall.2016/pdfs/Lawrence_et_al.pdf, Noudettu: 27.05.2020

Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition. Proceedings of the IEEE, Vol. 86, No. 11, 1998. Saatavissa: https://ieeexplore.ieee.org/document/726791, Noudettu: 03.09.2020

K. Li and J. Malik, Learning to optimize neural nets, arXiv preprint arXiv:1703.00441, 2017. Saatavissa: https://arxiv.org/pdf/1703.00441.pdf, Noudettu: 03.09.2020

J. Lighthill, Artificial Intelligence: A General Survey, Science Research Council, 1973, Saatavissa: http://www.chilton-computing.org.uk/inf/literature/reports/lighthill_report/p001.htm, Noudettu: 20.05.2020

D. Liu, A Practical Guide to ReLU, Medium, November 30, 2017, Saatavissa: https://medium.com/@danqing/a-practical-guide-to-relu-b83ca804f1f7, Noudettu: 26.05.2020

D. Liu, J. Yu, Otsu method and K-means. In 2009 Ninth International Conference on Hybrid Intelligent Systems, Vol. 1, pp. 344-349. IEEE, 2009. Saatavissa: https://ieeexplore.ieee.org/document/5254345, Noudettu: 12.09.2020

D.G. Lowe, Object recognition from local scale-invariant features. Proceedings of the seventh IEEE international conference on computer vision, Vol. 2, pp. 1150.1157, 1999. Saatavissa: https://ieeexplore.ieee.org/document/790410, Noudettu: 03.09.2020

G. Marcus, Deep Learning: A Critical Appraisal, arXiv preprint arXiv:1801.00631, 2018, Saatavissa: https://arxiv.org/ftp/arxiv/papers/1801/1801.00631.pdf, Noudettu: 26.06.2019

J. Markoff, Computer wins on 'Jeopardy!': Trivial, it's not. New York Times, 16, 2011, Saatavissa: https://www.nytimes.com/2011/02/17/science/17jeopardy-watson.html, Noudettu: 27.03.2020

J.R. Mashey, Big data… And the next wave of infrastress, Computer Science Division Seminar, University of California, Berkeley, 1997, Saatavissa: https://static.usenix.org/event/usenix99/invited_talks/mashey.pdf, Noudettu: 14.05.2020

M. Matsugu, K. Mori, Y. Mitari, Y. Kaneda, Subject independent facial expression recognition with robust face detection using a convolutional neural network. Neural Networks, Vol. 16, No. 5–6, pp. 555–559, 2003, Saatavissa: http://www.iro.umontreal.ca/~pift6080/H09/documents/papers/sparse/matsugo_etal_face_expression_conv_nnet.pdf, Noudettu: 07.09.2020

J. McAuley, J. Leskovec, Image labeling on a network: using social-network metadata for image classification, In European conference on computer vision, Springer, Berlin, Heidelberg, pp. 828-841. 2012. Saatavissa: https://cs.stanford.edu/~jure/pubs/image-eccv12.pdf, Noudettu: 04.04.2020

P. McCorduck, C. Cli, Machines who think: A personal inquiry into the history and prospects of Artificial Intelligence, CRC Press, 2004, 576 p. Saatavissa: https://books.google.fi/books?hl=en&lr=&id=r2C1DwAAQBAJ, Noudettu: 24.03.2020

W.S. McCulloch, W. Pitts, A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, Vol. 5, No. 4, 1943, pp.115-133. Saatavissa: http://aiplaybook.a16z.com/reference-material/mcculloch-pitts-1943-neural-networks.pdf, Noudettu: 22.05.2020

McKinsey & Company, Ask the AI experts: What's driving today's progress in AI?, McKinsey Analytics, 2017, Saatavissa: https://www.mckinsey.com/business-functions/mckinsey-analytics/our-insights/ask-the-ai-experts-whats-driving-todays-progress-in-ai, Noudettu: 22.05.2020

C. Mead, M. Ismail, Analog VLSI Implementation of Neural Systems (Vol. 80), Springer Science & Business Media, 1989, 250 p. Saatavissa: https://books.google.fi/books?hl=en&lr=&id=9e29dOiXeiMC, Noudettu: 12.05.2020

H. N. Mhaskar, Approximation properties of a multilayered feedforward artificial neural network. Advances in Computational Mathematics, Vol. 1, No. 1, pp. 61–80, 1993. Saatavissa: https://doi.org/10.1007/BF02070821, Noudettu: 01.09.2020

M. Minsky, S.A. Papert, Perceptrons: An introduction to computational geometry, MIT Press, 2017, 316 p. Saatavissa: https://books.google.fi/books?hl=en&lr=&id=PLQ5DwAAQBAJ, Noudettu: 24.05.2020

T. Mitchell, M. Hill, Machine Learning, 1997, 414 p. Saatavissa: http://www.cs.cmu.edu/~tom/mlbook.html, Noudettu: 13.01.2020

G.E. Moore, Cramming more components onto integrated circuits, 1965, Saatavissa: https://newsroom.intel.com/wp-content/uploads/sites/11/2018/05/moores-law-electronics.pdf, Noudettu: 12.05.2020

M. Najibi, P. Samangouei, R. Chellappa, L.S. Davis. Ssh: Single stage headless face detector, In Proceedings of the IEEE international conference on computer vision, pp. 4875-4884. 2017. Saatavissa: https://openaccess.thecvf.com/content_ICCV_2017/papers/Najibi_SSH_Single_Stage_ICCV_2017_paper.pdf, Noudettu: 28.08.2020

National Research Council, Funding a Revolution: Government Support for Computing Research, National Academic Press, 1999, 260 p. Saatavissa: https://books.google.fi/books?hl=en&lr=&id=XuhuAgAAQBAJ, Noudettu: 20.05.2020

S.K. Nayar, R.M. Bolle, Reflectance based object recognition, International journal of computer vision, Vol. 17, No. 3 pp. 219-240. 1996. Saatavissa: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.17.6253&rep=rep1&type=pdf, Noudettu: 13.09.2020

H. Newquist, Brain Makers, Editors & Engineers, Limited. 1994, 488 p. Saatavissa: https://dl.acm.org/doi/book/10.5555/528367, Noudettu: 15.05.2020

P. Nillius, J.O. Eklundh, Automatic estimation of the projected light source direction", Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2001, Vol. 1, 2001, Saatavissa: http://www.csc.kth.se/~nillius/publications/nillius_lightest.pdf, Noudettu: 13.05.2020

T.E. Oliphant, A guide to NumPy, Trelgol Publishing USA, Vol. 1, 378 p. 2006, Saatavissa: https://ecs.wgtn.ac.nz/foswiki/pub/Support/ManualPagesAndDocumentation/numpybook.pdf, Noudettu: 17.09.2020

F. Patin, An introduction to digital image processing, 2003, 49 p. Saatavissa: http://teachme.free.fr/ImageProc.pdf, Noudettu: 09.09.2020

J. Patterson, A. Gibson, Deep Learning: A Practitioner's Approach, O'Reilly Media, Inc., 2017, 532 p. Chapter 1, Saatavissa: https://books.google.fi/books?id=rLcuD-wAAQBAJ, Noudettu: 26.06.2019

R. Perrault, Y. Shoham, E. Brynjolfsson, J. Clark, J. Etchemendy, B. Grosz, T. Lyons, J. Manyika, S. Mishra, J.C. Niebles, The AI Index 2019 Annual Report, AI Index Steering Committee, Human-Centered AI Institute, Stanford University, Stanford, CA, 291 p. 2019. Saatavissa: https://hai.stanford.edu/sites/default/files/ai_index_2019_report.pdf, Noudettu: 07.05.2020

D.L. Pham, C. Xu, J.L. Prince, Current methods in medical image segmentation, Annual review of biomedical engineering, Vol. 2, No. 1, pp. 315-337. 2000. Saatavissa: https://www.ece.lsu.edu/gunturk/Topics/Segmentation-1.pdf, Noudettu: 12.09.2020

G. Pierobon, Visualizing intermediate activation in Convolutional Neural Networks with Keras, Github, 2018. Saatavissa: https://github.com/gabrielpierobon/cnnshapes, Noudettu: 19.09.2020

I. Pitas, Digital Image Processing Algorithms and Applications, John Wiley & Sons, 2000, 432 p. Saatavissa: https://books.google.fi/books?hl=en&lr=&id=VQs_Ly4DYDMC, Noudettu: 12.05.2020

D. Poole, A. Mackworth, R. Goebel, Computational Intelligence: A Logical Approach. New York: Oxford University Press, 1998, ISBN 978-0-19-510270-3. Noudettu: 14.01.2020

Python Software Foundation, CSV file reading and writing, 2009. Saatavissa: https://docs.python.org/3/library/csv.html, Noudettu: 18.09.2020.

Python Software Foundation, Mathematical functions, 2001. Saatavissa: https://docs.python.org/3/library/math.html, Noudettu: 18.09.2020.

E. Rahtu, J. Kannala, M. Salo, J. Heikkilä, Segmenting salient objects from images and videos. In European conference on computer vision, pp. 366-379. Springer, Berlin, Heidelberg, 2010. Saatavissa: https://link.springer.com/content/pdf/10.1007/978-3-642-15555-0_27.pdf, Noudettu: 11.09.2020

S. Ren, K. He, R. Girshick, J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In Advances in Neural Information Processing Systems, pp. 91–99, 2015. Saatavissa: http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf, Noudettu: 25.05.2020

N. Rochester, J. Holland, L. Haibt, W. Duda, "Tests on a cell assembly theory of the action of the brain, using a large digital computer", IRE Transactions on information Theory, Vol. 2, No. 3, 1956, pp. 80-93. Saatavissa: https://ieeexplore.ieee.org/document/1056810, Noudettu: 23.05.2020

F. Rosenblatt, The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, Vol. 65, No. 6, 1958, pp. 386–408. Saatavissa: http://web.engr.oregonstate.edu/~huanlian/teaching/machine-learning/2017fall/extra/rosenblatt-1958.pdf, Noudettu: 24.05.2020

H.A. Rowley, S. Baluja, T. Kanade, Neural network-based face detection. IEEE Transactions on pattern analysis and machine intelligence, Vol. 20, No. 1, pp.23–38, 1998. Saatavissa: https://www.ri.cmu.edu/pub_files/pub1/rowley_henry_1996_3/rowley_henry_1996_3.pdf, Noudettu: 03.09.2020

S. Ruder, An overview of gradient descent optimization algorithms, 19 Jan 2016, Saatavissa: https://ruder.io/optimizing-gradient-descent/, Noudettu: 30.08.2020

D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning internal representations by error propagation, No. ICS-8506, California Univ San Diego La Jolla Inst for Cognitive Science, 1985, Saatavissa: https://archive.org/details/paralleldistribu00rume/mode/2up, Noudettu: 25.08.2020

O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, L. Fei-Fei, Imagenet large scale visual recognition challenge. International Journal of Computer Vision, Vol. 115, No. 3, pp. 211–252, 2015. Saatavissa: https://arxiv.org/pdf/1409.0575.pdf, Noudettu: 03.09.2020

I. Russell, The delta rule, University of Hartford, West Hartford, 2012. Saatavissa: https://web.archive.org/web/20160304032228/http://uhavax.hartford.edu/compsci/neural-networks-delta-rule.html, Noudettu: 17.08.2020

S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach (2nd Edition), 2003, 1080 p. Saatavissa: https://books.google.fi/books?id=KI2WQgAACAAJ, Noudettu: 15.05.2020

I. Sample, Computer says no: why making Ais fair, accountable and transparent is crucial, the Guardian, November 5, 2017, Saatavissa: https://www.theguardian.com/science/2017/nov/05/computer-says-no-why-making-ais-fair-accountable-and-transparent-is-crucial, Noudettu: 13.05.2020

A.L. Samuel, Some Studies in Machine Learning Using the Game of Checkers, IBM Journal of Research and Development, Vol 3, No 3, 1959, pp. 210-229. Saatavissa: https://ieeexplore.ieee.org/document/5392560/, Noudettu: 12.03.2020

A. Sanin, C. Sanderson, B.C. Lovell, Shadow detection: A survey and comparative evaluation of recent methods, Pattern recognition, Vol. 45, No. 4, pp. 1684-1695. 2012. Saatavissa: https://www.sciencedirect.com/science/article/abs/pii/S0031320311004043 Noudettu: 13.09.2020

J. Schaeffer, Didn't Samuel Solve That Game? One Jump Ahead, Springer, Boston, MA, 2009, pp. 1-11. Saatavissa: https://link.springer.com/chapter/10.1007%2F978-0-387-76576-1_7, Noudettu: 12.03.2020

D. Scherer, A. Müller, S. Behnke, Evaluation of pooling operations in convolutional architectures for object recognition. In International conference on artificial neural networks, Springer, Berlin, Heidelberg, 2010, pp. 92-101. Saatavissa: https://www.ais.uni-bonn.de/papers/icann2010_maxpool.pdf, Noudettu: 01.06.2020

J. Schmidhuber, Deep learning in neural networks: An overview, Neural networks, Vol. 61, 2015, pp. 85-117. Saatavissa: https://arxiv.org/pdf/1404.7828.pdf, Noudettu: 24.05.2020

S. Shalev-Shwartz, S. Ben-David, Understanding machine learning: From theory to algorithms, Cambridge University Press, 2014, Saatavissa:

https://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/understanding-machine-learning-theory-algorithms.pdf, Noudettu: 26.06.2019

A. V. Sharma, Understanding Activation Functions in Neural Networks, The Theory of Everything, Medium, Saatavissa: https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0, Noudettu: 25.08.2020

K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556, 2014, Saatavissa: https://arxiv.org/pdf/1409.1556.pdf, Noudettu: 01.09.2020

G. Selvendy, editor. *Handbook of Industrial Engineering: technology and operations management*, John Wiley & Sons, 2001. ISBN 9780471330578. Noudettu: 27.06.2019

L.G. Shapiro, G. Stockman, Computer Vision, Pearson, 608 p. 2001. Saatavissa: http://nana.lecturer.pens.ac.id/index_files/referensi/computer_vision/Computer%20Vision.pdf, Noudettu: 12.09.2020

M. Sipser, Introduction to the Theory of Computation (3rd Edition), Cengage Learning, 2012, 458 p. Saatavissa: https://dl.acm.org/doi/pdf/10.1145/230514.571645, Noudettu: 14.01.2020

N. Srivastava, G.E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, Vol. 15, No. 1, pp. 1929–1958, 2014. Saatavissa: https://dl.acm.org/doi/pdf/10.5555/2627435.2670313, Noudettu: 01.09.2020

Suomen standardisoimisliitto, SI-opas, SFS-oppaat, 2002, 32 p. Saatavissa: http://web.archive.org/web/20120831234747/http://www.sfs.fi/files/70/si-opas.pdf, Noudettu: 10.09.2020

I. Sutskever, J. Martens, G. Dahl, G. Hinton, On the importance of initialization and momentum in deep learning. In International conference on machine learning, pp. 1139-1147, 2013. Saatavissa: http://proceedings.mlr.press/v28/sutskever13.pdf, Noudettu: 02.09.2020

I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks. In Advances in Neural Information Processing Systems, pp. 3104–3112, 2014. Saatavissa: http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf, Noudettu: 07.09.2020

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1-9, 2015. Saatavissa: https://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Szegedy_Going_Deeper_With_2015_CVPR_paper.pdf, Noudettu: 08.09.2020

A.B. Tickle, R. Andrews, M. Golea, J. Diederich, The truth will come to light: directions and challenges in extracting the knowledge embedded within trained artificial neural networks, IEEE Transactions on Neural Networks, Vol. 9, No. 6, 1998, pp. 1057-1068, Saatavissa: https://ieeexplore.ieee.org/document/728352, Noudettu: 12.05.2020

S. Tiwari, FaceDetect, Github, 2017, Saatavissa: https://github.com/shantnu/FaceDetect, Noudettu: 19.09.2020

G. Trigeorgis, P. Snape, I. Kokkinos, S. Zafeiriou, Face Normals "In-the-Wild" Using Fully Convolutional Networks, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 340-349. 2017. Saatavissa: https://ieeexplore.ieee.org/document/8099527, Noudettu: 12.04.2020

A. Turing, Computing Machinery and Intelligence, Parsing the Turing Test, 2009, pp. 23-65, Saatavissa: https://link.springer.com/chapter/10.1007/978-1-4020-6710-5_3, Noudettu: 23.01.2020

S. Van der Walt, J.L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J.D. Warner, N. Yager, E. Gouillart, T. Yu, scikit-image: image processing in Python, PeerJ 2, e453. 2014. Saatavissa: https://peerj.com/articles/453/, Noudettu: 18.09.2020

G. Van Rossum, F.L. Drake Jr, Python Tutorial: Release 3.8.1 (2020), Vol. 620, Amsterdam: Centrum voor Wiskunde en Informatice, 1995, Saatavissa: http://msekce.karlin.mff.cuni.cz/~halas/IT/tutorial.pdf, Noudettu: 24.4.2020

V. Vapnik, The nature of statistical learning theory. Springer science & business media, 314 p. 2013. Saatavissa: https://books.google.fi/books?hl=en&lr=&id=EqgACAAAQBAJ, Noudettu: 28.06.2020

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems, 2017, pp. 5998-6008, Saatavissa: https://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf, Noudettu: 21.05.2020

J. Wang, Formal Methods in Computer Science, CRC Press, 2019, 350 p. Saatavissa: https://books.google.fi/books?hl=en&lr=&id=KUqfDwAAQBAJ, Noudettu: 23.05.2020

P.D. Wasserman, T. Schwartz, Neural networks. II. What are they and why is everybody so interested in them now? IEEE Expert Vol. 3, No. 1, 1988, pp. 10–15. Saatavissa: https://ieeexplore.ieee.org/document/2091, Noudettu: 01.09.2020

J. Watkinson, The art of digital video, 4th edition, Taylor & Francis, 2008, 672 p. Saatavissa: https://books.google.fi/books?id=8uLEXlN9ouAC, Noudettu: 10.09.2020

R.J. Williams, Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning, Vol. 8, No. 3-4, pp. 229-256, 1992. Saatavissa: https://link.springer.com/content/pdf/10.1007/BF00992696.pdf, Noudettu: 02.09.2020

L. Xie, A. Yuille, Genetic CNN, In Proceedings of the IEEE international conference on computer vision, pp. 1379-1388, 2017. Saatavissa: https://openaccess.thecvf.com/content_ICCV_2017/papers/Xie_Genetic_CNN_ICCV_2017_paper.pdf, Noudettu: 08.09.2020

Y. Yao, L. Rosasco, A. Caponnetto, On early stopping in gradient descent learning. Constructive Approximation, Vol. 26, No. 2, pp. 289–315, 2007. Saatavissa: http://web.mit.edu/lrosasco/www/publications/earlystop.pdf, Noudettu: 02.09.2020

X. Yin, X. Yu, K. Sohn, X. Liu, M. Chandraker, Towards large-pose face frontalization in the wild, In Proceedings of the IEEE international conference on computer vision, pp. 3990-3999. 2017. Saatavissa: https://openaccess.thecvf.com/content_ICCV_2017/papers/Yin_Towards_Large-Pose_Face_ICCV_2017_paper.pdf, Noudettu: 14.09.2020

S. Zafeiriou, M. Hansen, G. Atkinson, V. Argyriou, M. Petrou, M. Smith, L. Smith, The photoface database, In CVPR 2011 WORKSHOPS, IEEE, pp. 132-139. 2011. Saatavissa: https://ieeexplore.ieee.org/document/5981840/, Noudettu: 13.03.2020

W. Zaremba, I. Sutskever, O. Vinyals, Recurrent neural network regularization. arXiv preprint arXiv:1409.2329, 2014. Saatavissa: https://arxiv.org/pdf/1409.2329.pdf, Noudettu: 03.09.2020

A. Zell, Simulation neuronaler netze, Vol. 1, No. 5.3, Bonn: Addison-Wesley, 1994. Noudettu: 23.08.2020

C. Zhang, S. Bengio, M. Hardt, B. Recht, O. Vinyals, Understanding deep learning requires rethinking generalization, arXiv preprint arXiv:1611.03530, 2016, Saatavissa: https://arxiv.org/pdf/1611.03530.pdf, Noudettu: 21.05.2020

B. Zoph, Q.V. Le, Neural architecture search with reinforcement learning, arXiv preprint arXiv:1611.01578, 2016, Saatavissa: https://arxiv.org/pdf/1611.01578.pdf, Noudettu: 21.05.2020

B. Zoph, V. Vasudevan, J. Shlens, Q.V. Le, Learning transferable architectures for scalable image recognition, Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8697-8710. Saatavissa: http://openaccess.thecvf.com/content_cvpr_2018/papers/Zoph_Learning_Transferable_Architectures_CVPR_2018_paper.pdf, Noudettu: 21.05.2020