OULUN YLIOPISTO

TIETO- JA SÄHKÖTEKNIIKAN TIEDEKUNTA

Oula Hourula
Taru Järvelä
Joonas Soudunsaari

Kangasmerkkien tunnistus neuroverkkojen avulla

Kandidaatintyö
Tietotekniikan tutkinto-ohjelma
Kesäkuu 2019

Hourula O, Järvelä T. & Soudunsaari J. (2019) Kangasmerkkien tunnistus neuroverkkojen avulla. Oulun yliopisto, tietotekniikan tutkinto-ohjelma. Kandidaatintyö, 27 s.

TIIVISTELMÄ

Lisätty todellisuus on noussut 2010-luvulla suureksi keskustelun aiheeksi. Ihmiset näkevät tulevaisuuden, jossa teknologia ja todellinen elämä sulautuvat toisiinsa. Tämän suorittamiseen kuvantunnistus ja konenäkö ovat olennaisia osia.

Projektin pohjana oli luoda sovellus, jolla käyttäjä voi kuvata kangasmerkkiä ja sovellus antaa palautteena merkin tarjoajan ja paikan. Sovellus tarvitsi siis sisäänsä jonkinlaisen kuvantunnistusosion. Kuvan tunnistamiseen on kuitenkin tarjolla jo monia erilaisia algoritmeja.

Työssä päädyttiin käyttämään neuroverkkoa, joka käyttää Alexnetin johdannaista mallia. Ideana oli arvioida kuvien määrän vaikutusta neuroverkon oppimiseen sekä neuroverkon sopivuutta ylipäänsä tämän tyyppiseen kuvantunnistusongelmaan. Neuroverkko pyrkii lajittelemaan kuvat haluttuihin luokkiin. Tutkimusmateriaalina käytettiin kangasmerkkejä. Materiaalin määrää moninkertaistettiin pyörittämällä merkkejä eri kulmiin verrattuna alkuperäiseen. Verkon tehokkuutta tutkittiin tekemällä viisi eri versiota, joihin jokaiseen tehtiin muutoksia verrattuna edellisiin. Muutoksia olivat koulutusiteraatioiden määrä sekä koulutusmateriaalin määrä. Hypoteesina oli, että mitä enemmän materiaalia neuroverkolla on käytettävänä koulutuksessa, sitä paremmin sen tulisi myös kyetä arvaamaan kuvan luokka oikein.

Tuloksia käsiteltiin rakentamalla taulukko eri neuroverkon versioista sekä tutkimalla neuroverkon kehitystä samojen kuvien suhteen. Neuroverkon kehittyminen koulutuskuvien määrän suhteen ei muuttunut merkittävästi 1000 koulutuskuvan jälkeen. Tunnistusprosentti säilyi noin 95% kohdalla, vaikka koulutuskuvia käytettiin 3000. Selkeinä eroina nähtiin kuvien värin tai värimaailman vaikutus. Kangasmerkkien muodoilla nähtiin suhteellisen vähän vaikutusta neuroverkon tunnistamiskykyyn.

Tutkimuksen perusteella voidaan todeta, että edellä kuvattu kuvantunnistusmenetelmä kykenee täyttämään sovelluksen vaatimukset kangasmerkkien tunnistamiseksi kohtuullisella määrällä kuvia.

Avainsanat: kuvantunnistus, koneoppiminen, konenäkö

Hourula O, Järvelä T. & Soudunsaari J. (2019) Recognizing patches with neural networks. University of Oulu, Degree Programme in Computer Science and Engineering. Bachelor's Thesis, 27 p.

ABSTRACT

Augmented reality has risen to be a familiar topic in the 2010s. People see the possibilities in merging technology with reality. This is where image recognition and machine vision become essential parts of.

The basic idea of the project was to create an application that an user can take a picture of a patch with and in return get information about the patch provider and their location. The application would, in this case, need some kind of image recognition section. To recognize an image, there have already been developed many different algorithms.

The work ended up using a neural network that uses Alexnet's derivative model. The idea was to estimate how the amount of images affects the neural network learning and, in general, how well neural networks serve to solving this kind of image recognition problem. The neural network aims at sorting images to desired classes. Cloth patches were used as a research material. The amount of research material was multiplied by rolling each patch to different angles compared to the original. The efficiency of the neural network was studied by creating five different versions, making changes to each of them in comparison to previous version. The changes included the number of training iterations and the amount of training material. The hypothesis was that the more training material there was for the neural network, the better it was guessing the class right.

The results were handled by creating a table of different neural network versions and studying the neural network evolution between the same images. The evolution involving the amount of training images didn't change much after 1000 training images. The percent of successful recognition stayed at about 95 % although 3000 training images were used. Clear differences were noticed in the effect of image color and color scheme. The shape of the patches had relatively small impact on recognition. Based on the study, it can be concluded that the above-mentioned image recognition method can fulfill the application's requirements for identifying patches with a reasonable number of images.

Keywords: Image recognition, Machine learning, Machine vision

SISÄLLYSLUETTELO

ALKULAUSE

Haluamme kiittää Teemu Tokolaa kärsivällisestä työn ohjauksesta sekä professori Juha Röningiä.

Oulu, kesäkuu 2019

Taru Järvelä
Joonas Soudunsaari
Oula Hourula

LYHENTEIDEN JA MERKKIEN SELITYKSET

AR	Augmented reality
MNIST	Modified National Institute of Standards and Technology
RGB	Red Green Blue
Kaggle	datatieteeseen ja koneoppimiseen erikoistunut nettiyhteisö
Pillow/PIL-kirjasto	Python Imaging Library
CPU	Central Processing Unit
GPU	Graphics Processing Unit
LMDB	Lightning Memory-Mapped Database
CUDA	Compute Unified Device Architecture
BLAS	Basic Linear Algebra Subprograms
OpenCV	Open Source Computer Vision
BVLC	Berkeley Vision and Learning Center

1. JOHDANTO

Konenäkö on vanha tutkimusalue, mutta vasta viime vuosikymmenen tuoma laskentatehon suuri kasvu ja mahdollisuus turvautua pilvipalveluihin, on sallinut käytännön sovellusten kehittämisen. Tästä huolimatta kuvantunnistus on edelleen laskennallisesti raskasta. Konenäön iän vuoksi aiheesta on tehty paljon tutkimuksia ja tutkimusmenetelmiä on tuhansia.

Tutkimuksen tavoitteena oli tutkia eri menetelmiä havaita valokuvasta kangasmerkkejä ja tunnistaa mistä merkistä on kyse. Sovellus, jolle tätä tutkimusta tehtiin, hakisi tunnistuksen jälkeen tietokannasta merkin nimen ja myyjän, minkä jälkeen tiedot näytettäisiin käyttäjälle. Sovelluksen on tarkoitus olla lisätyn todellisuuden sovellus, joka reaaliajassa tunnistaa kuvia ja antaa palautetta käyttäjälle.

1.1. Taustatietoa

Azuma ja Ronald T. määrittelevät vuonna 1997 kirjoitetussa artikkelissaan “A survey of augmented reality” lisätyn todellisuuden seuraavalla tavalla: yhdistää todellista ja virtuaalista maailmaa, on vuorovaikutteinen reaaliaikaisesti ja asettaa virtuaaliset ja reaaliset objektit geometrisesti kolmeen ulottuvuuteen todellisessa maailmassa. [1] Käytännössä tämä tarkoittaa sitä, että esimerkiksi huoneeseen, jota tarkastellaan jonkin AR-laitteen kautta, voidaan lisätä sovelluksen avulla virtuaalisia huonekaluja, jotka näyttävät sulautuvan todelliseen maailmaan noudattaen sen rajoituksia.

Lisättyä todellisuutta käytetään monilla aloilla, kuten turismissa, viihteessä, markkinoinnissa, kirurgiassa, logistiikassa, teollisuudessa ja huoltamisessa [2]. Lisätyn todellisuuden virtuaalisia objekteja voidaan käyttää auttamaan käyttäjää suorittamaan tehtäviä oikeassa maailmassa [1]. Esimerkiksi aivokirurgi, kerättyään ensin dataa tutkimuksilla, voisi käyttää lisättyä todellisuutta näyttämään potilaan kallossa tarkan kohdan, johon porata reikä [1].

Iso alue lisätyn todellisuuden alueella on kuvantunnistus, sillä modifioidakseen ympäristöä, koneen täytyy kyetä ymmärtämään sitä. Kuvantunnistussovelluksia on olemassa kohtalaisesti. Varsinkin Google on alan edelläkävijä, kuten havainnollistaa jatkuvassa kehityksessä oleva ja hyvin suoriutuva tensorflow-kehitysympäristö [3]. Useimmat Googlen, ja muiden, sovellukset eivät kuitenkaan erikoistu tarpeeksi syvälle. Ne tunnistavat lähinnä korkeampia luokkia, kuten eläimiä, esineitä ja ihmisiä. Kangasmerkeistä ne saattaisivat tunnistaa tekstiä tai materiaaleja, mutta eivät kykene kovin tarkasti kuvailemaan merkkiä, eivätkä nimeämään niitä, ellei niitä ole erikseen koulutettu siihen. Kuvantunnistusta käytetään kuitenkin paljon myös tunnistamaan ihmisiä ja kasvoja. Mikroilmeiden tunnistaminen on hyödyllistä esimerkiksi tunnistamaan ovatko videomateriaalin kasvot aidot [4][5]. Myös ihmisen ikää voidaan estimoida käyttäen kuvantunnistusta [6].

Nykyajan älypuhelimissa ja tableteissa on tehokkaat prosessorit, isot näytöt, sisäänrakennetut paikannusanturit ja kamerat. Näiden ansiosta ne tarjoavat hyvän alustan lisätyn todellisuuden sovelluksille [7]. Kangasmerkkien tapauksessa lisättyä todellisuutta voisi hyödyntää merkin skannauksen yhteydessä. Kangasmerkit voivat olla esimerkiksi käytössä kuluneita, likaisia tai vinossa. Näistä syistä sovelluksen tunnistus ei ole aina 100% tarkka. Käyttäjä voisi vahvistaa merkin vastaavaksi, kun alkuperäinen käyttämätön merkki näytetään lisätyn todellisuuden avulla skannatun

merkin tilalla tai vierellä. Tällöin, jos merkit vastaavat toisiaan, käyttäjälle näytetään mikä merkki on kyseessä ja kuka sen tarjoaja on.

1.2. Tavoitteet ja haasteet

Tavoitteena on luoda sovellus, joka kykenee tunnistamaan eri kangasmerkit toisistaan. Laitteistona käytössä on Raspberry Pi 2 -minitietokone, johon on liitettävissä näyttö ja kamera. Laitteen on tarkoitus saada kuva- tai videosyöttö, josta sovellus eristää halutun merkin. Sovellus vertaa syötettyä merkkiä tietokannassa oleviin merkkeihin ja ilmoittaa ainejärjestön, joka merkkiä mahdollisesti tarjoaa. Mahdollisena lisäominaisuutena voidaan sovellukseen lisätä käyttäjälle mahdollisuus lisätä uusia merkkejä järjestelmään. Sovellus voi myös toimia jonkinlaisena pelinä, missä voi kerätä mahdollisimman paljon erilaisia merkkejä. Palkintona pisteytyksiä ja saavutuksia kerättyjen merkkien määrien mukaan.

Suurimpana haasteena projektissa oli tarve isolle määrälle kangasmerkkejä. Vaikka rajaamme tutkimuksessamme tunnistettavien kuvien määrän matalaksi, pitkällä tähtäimellä ollakseen hyödyllinen, sovelluksen tulisi pystyä tunnistamaan suuri määrä eri merkkejä. Käytännön tasolla suurin haaste on materiaalin kerääminen. Vastaavaa samanlaista tutkimusta kangasmerkkien tunnistamisesta ei tiettävästi ole tehty.

1.3. Tutkimusmenetelmät ja työn rajaus

Kokonaisen sovelluksen kehittäminen vaatii todella paljon taustatyötä, sillä kangasmerkkejä löytyy lukemattomia ja jokainen vaatii työn alussa tuntemattoman määrän kuvia. Näistä syistä työ rajattiin kuvantunnistusalgoritmin kehittämiseen vain muutaman eri kangasmerkin tunnistamiseen. Työn on tarkoitus kartoittaa sovelluksen todellista mittakaavaa arvioimalla tarvittavaa kuva määrää sekä aikaa, jonka koulutus vie. Työryhmä keräsi koulutusmateriaalin itse ja tulokset arvioitiin suoraan koulutusajasta sekä erillisten testien määrittelemän tarkkuusprosenttien mukaan. Materiaalin kerääminen todellista sovellusta varten tehtäisiin käyttämällä käyttäjäkantaa, joka syöttäisi kuvia sovellukseen. Neuroverkko pyörisi koulutettavana pilvispalvelimella kouluttaen itseään tasa-ajoin uudella syötetyllä datalla. Näin myös neuroverkon koulutus pitkällä aikavälillä tehostuu.

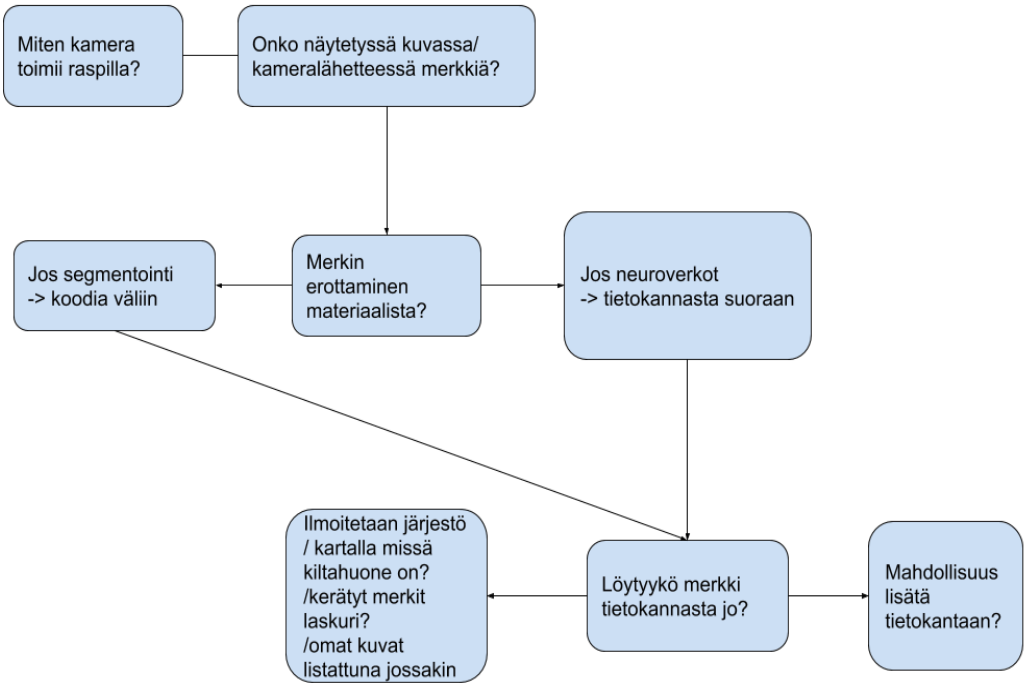
1.3.1. Tutkimuskysymykset

Tutkimuksen pääasiallinen tehtävä oli selvittää sekä tunnistusprosentti jollekin kuvantunnistusmenetelmälle että koulutustietokannan laajuuden vaikutus tähän prosenttiin. Kuvakannan kokoa rajaamalla pystytään selvittämään kannan koon vaikutusta tunnistuskykyyn.

Toinen tutkimusalue oli koulutusaika. Koulutuksen edistymistä mitattiin koko ajan ja pyrittiin arvioimaan valmistumisprosenttia mittaamalla, kuinka paljon verkko tekee virheitä arvioissaan. Lisäksi pyrittiin tutkimaan eri koulutusmenetelmiä ja vertaamaan niitä toisiinsa.

Tutkimustavoitteet olivat kohtalaisen yksinkertaisia, joten varsin yksinkertaisten tutkimusmenetelmien uskottiin riittävän vastausten löytämiseen. Koulutusajan

mittaaminen oli tosin omalla tavallaan haastavaa, sillä vaikka ajan ottaminen koulutuksesta oli helppoa, pitäisi koulutus suorittaa monta kertaa, jotta saataisiin hyödyllistä informaatiota. Tämä pyrittiin tekemään ajan salliessa.



Kuva 1. Suunnitellun sovelluksen perustoimintakaavio.

1.3.2. Työvaiheet

Ensimmäinen työvaihe oli tehdä taustatutkimusta eri konenäkö- ja kuvantunnistusmenetelmistä. Vaikka neuroverkot olivat lupaavia alustavilla tiedoilla, ei haluttu lukkiutua yhteen menetelmään, mikäli parempi tapa löytyisi. Pyrittiin etsimään samantyylisiä aiemmin tehtyjä tutkimuksia sekä muita samankaltaisia projekteja ja selvittämään minkälaisia menetelmiä niissä on käytetty. Kuvia erilaisista kangasmerkeistä kerättiin projektin alkupuoliskon aikana.

Tämän jälkeen päästiin itse sovelluksen suunnitteluun, jota nähdään kuvassa 1. Funktionaalista ohjelmistoa varten tarvittiin ainakin tapa luoda koulutus- ja testikuvatietokannat, varsinaisen näkevän algoritmin sekä tavan testata algoritmia. Tutkimuksen tavoitteiden täytön jälkeen aioimme sivututkimuksena selvittää tavan ottaa kuvia kameralla, sekä näyttää tulokset käytössä olevalla näytölle. Käytettyyn laitteistoon kuului kosketusnäyttö, langaton verkkoyhteys ja kamera. Suunnitelmissa oli, että testausvaiheessa kuvia otettaisiin laitteiston kameralla.

Varsinaisen algoritmin testaus ja koulutus tehtiin tiimin sisäisesti ottamalla paljon kuvia kangasmerkeistä erilaisia pintoja vasten. Kuvista luotiin lisämateriaalia kuvanmuokkausohjelmalla, jotka sittemmin jaettiin koulutus- ja testauskuviin. Viimeisenä työvaiheena oli raportin tekeminen, johon kerättiin kaikki tutkimusvaiheet ja käsiteltiin saatu data.

2. YLEISIÄ KUVANTUNNISTUSMENETELMIÄ

Ihmisille kuvantunnistus on luontevaa ja ihminen kykenee helposti erottamaan kaksiulotteisesta kuvasta objekteja, mutta koneet ovat historiallisesti olleet varsin huonoja tässä. Koska aivomme ovat kehittyneet analysoimaan ympäröivää maailmaamme näköaistin avulla, emme usein hahmota kuvantunnistuksen laskennallista haastavuutta. Julkaisussaan "Why is Real-World Visual Object Recognition Hard" [8], James J DiCarlo analysoi syitä näkemisen hankaluudelle. DiCarlo nostaa ensisijaisesti esille sen, että jokainen objekti kykenee luomaan rajattoman määrän kuvia katsojalle, riippuen katsojan näkökulmasta ja objektin valaistuksesta.

Kuvantunnistukselle on kuitenkin suuri määrä hyödyllisiä käyttökohteita, aina kappaleiden lajittelusta itseohjautuviin autoihin, mikä on johtanut useisiin eri menetelmiin hahmottaa maailmaa koneellisesti. Iso osa digitaalisen kuvankäsittelyn menetelmistä kehitettiin 1960-luvulla [9], ja nämä ovat vuosien aikana kehittyneet ja johtaneet uusiin laskennallisiin menetelmiin nähdä maailmaa. Näistä tutkittiin läpi muutamia, joiden arveltiin soveltuvan tähän työhön.

2.1. Segmentointimenetelmät

Kuvan segmentoinnilla tarkoitetaan kuvan jakamista eri alueisiin, jolloin vierekkäiset alueet eivät muodosta yhdessä homogeenistä kokonaisuutta. Mustavalkokuvan segmentoinnissa kuva jaotellaan tarkastelemalla eristettyjä pisteitä, viivoja ja reunoja, etsien jyrkkiä muutoksia harmaasävyn arvossa. Segmentointi voidaan suorittaa myös perustuen alueiden homogeenisyyteen. Tällöin käytettäviin menetelmiin kuuluu kynnystys, klusterointi, alueen kasvatus, alueen lohkotus ja yhteen sulauttaminen.

2.1.1. Värikuvan segmentointi

Värillisestä kuvasta saadaan intensiteetin lisäksi enemmän tietoa kuin harmaasävyisestä kuvasta ja väri usein helpottaa kuvan alueiden erittelyä. Väri on hyödyllinen kuviontunnistukselle ja konenäölle. Suurinta osaa harmaakuvan segmentointimenetelmistä voidaan hyödyntää myös värikuvan tapauksessa. Näitä ovat muun muassa histogrammin kynnystys, klusterointi, reunan tunnistus ja neuroverkot. Harmaakuvan segmentointimenetelmiä voidaan käyttää värikuvan yksittäisille komponenteille ja yhdistämällä niistä saadaan lopullinen tulos segmentoinnille. Tästä seuraavia ongelmia ovat esimerkiksi se, miten saatua väritietoa käytetään kokonaisuutena jokaiselle pikselille. Kun väri jaetaan kolmeen komponenttiin, kuvasta tulee käytännössä monispektrinen ja ihmissilmän havaitsema tieto väreistä kadotetaan. Ongelmana on myös valita segmentoinnissa oikea värien kuvaustapa.[10]

RGB-värimalli ei sovellu hyvin kuvien segmentointiin johtuen R-, G- ja B-komponenttien välisestä suhteesta. Jos kuvan intensiteetti muuttuu, kaikkien kolmen komponentin arvot muuttuvat yhtä paljon. Myöskään kahden värin välinen etäisyys RGB-avaruudessa ei kerro värien samankaltaisuudesta.[10]

Yleensä värikuvien segmentointimenetelmissä alueen määrittely perustuu värin samankaltaisuuteen. Ongelmia algoritmeille aiheuttavat varjot, valotukset ja pinnan rakenne.[10]

2.1.2. Muotovertailu

S.Belongie, J.Malik, ja J.Puzicha kuvasivat muodon vertailua (shape matching) julkaisussaan [11] yksinkertaiseksi mutta monipuoliseksi tavaksi tunnistaa objekteja kuvasta. Menetelmä perustuu kolmivaiheiseen lähestymistapaan jossa:

1. Ratkaistaan vastaavuusongelma kahden muodon välillä
2. Käytetään vastaavuutta toisen muodon muuttamiseen
3. Lasketaan muotojen välinen etäisyys summaamalla vastapisteiden ero yhdessä muodonmuutosta kuvaavan termin kanssa

Käytännössä tämä tarkoittaa sitä, että aluksi muotoon määritellään n määrä pisteitä, esimerkiksi reunavertailua apuna käyttäen. Jotta edellä mainittu onnistuisi, täytyy kuva segmentoida etukäteen. Seuraavaksi selvitetään jokaiselle pisteelle vastinkappale toisesta kuvasta (vaihe 1). Kun vastakappaleet ovat tiedossa, toista kuvaa voidaan muokata siten, että muodot täsmäävät (vaihe 2). Viimeiseksi lasketaan kuinka lähellä kuvat ovat toisiaan, jotta voidaan sanoa, onko kuvissa sama objekti.

Edellä mainitussa tutkimuksessa saavutettiin vakuuttavat tulokset. Käsin kirjoitettujen numeroiden tunnistuksessa virhe oli vain 0,63 prosenttia ja kolmiulotteisien objektien tunnistuksessa 2,4 prosenttia. Koska tutkimuksemme kuvantunnistusongelmassa käsitellään kaksiulotteisia kuvia, olisi muodon vertailulla suurta potentiaalia ratkaisuna

2.1.3. Reunavertailu

Reunavertailussa kuva analysoidaan ensin siten, että käydään kuvan jokainen pikseli läpi, ja todetaan, onko se osa reunaa vai ei. P. H. Eichelin ja D. J. Delphin julkaisussa [12] todetaan, että algoritmit jakautuvat kolmeen: liuku (gradient-based), malliparitus (template matching) ja parametrimallit (parametric models). Näiden välillä merkittävät erot ovat kuvan pikseleiden intensiteettien käsittelyn arviointi. Yleensä kaikki nämä tavat kuitenkin käyttävät jotain yleistä tai joustavaa raja-arvoa, joka päättää onko annettu pikseli osa reunaa vai ei.

2.1.4. Aluetunnistus

Aluetunnistuksessa tavoite on paikallistaa alueita kuvasta pääasiallisesti vertailemalla pikseleitä toisiinsa esimerkiksi RGB-väriarvon perusteella. Pikselit, jotka ovat lähellä toisiaan ja jakavat samankaltaisen väriarvon, ovat luultavasti osa samaa kappaletta. Työssään aluetunnistamisen ja yhdistämisen parissa Alain Tremeau ja Nathalie Borel suorittivat aluetunnistuksen kolmen kriteerin avulla [13]. Ensin he käyttivät paikallista homogeenisyyden kriteeriä (LHC) joka keskittyy vain naapuripikseleiden vertailuun. Seuraavaksi käytetään keskiarvoista homogeenisyyden kriteeriä (AHC1), joka käyttää lokaalia ja alueellista vertailua keskittyen vain yhteen tutkittavaan alueeseen. Viimeisenä käytetään toista keskiarvoista homogeenisyyden kriteeriä

(AHC2) joka eroaa edellisestä siten että se käyttää vertailussaan myös globaalisia arvoja, saaden vaikutusta muista alueista.

Tremeaun ja Borelin esimerkki on vain yksi havainnollistava tapa erotella alueita toisistaan, mutta se havainnollistaa hyvin aluetunnistuksessa avainasemassa olevan kriteerin valinnan tärkeyden. Eri menetelmät valitsevat erottelu kriteerinsä eri perustein saadakseen joko laajempia taikka yksityiskohtaisempia alueita. Aluetunnistus soveltuu tunnistamaan nimensä mukaisesti alueita, joten se sopisi hyvin merkkien löytämiseen kuvasta, mutta se ei yksikseen kykene luokittelemaan merkkejä erilleen, joten aluetunnistus jätettiin syrjään tutkimuksestamme.

2.2. Neuroverkot

Viime aikoina yhdeksi suosituimmista menetelmistä kuvantunnistuksessa ovat nousseet neuroverkot. Neuroverkot ovat olleet tunnettuja 80-luvulta asti, mutta niiden ei pitkään uskottu kykenevän kompleksisiin analyyseihin. Tämä johtui paljolti siitä, että alkeelliset neuroverkot eivät sisältäneet piilotasoja ja koska syvien verkkojen kouluttamiseen ei ollut hyviä menetelmiä. Tähän tuli muutos vastavirta-algoritmin (backpropagation) muodossa, jonka näytettiin kykenevän ratkaisemaan xor-ongelman [14], mikä oli aiemmin ollut hankala, sillä yksinkertaiset neuroverkot eivät olleet hyviä ratkaisemaan epälineaarisia ongelmia. Verkkojen koulutusmenetelmien kehittyminen ja laskentatehon kasvu ovat nostaneet neuroverkkojen asemaa ja neuroverkot ovatkin yksi käytetyimpiä ja tutkituimpia konenäön osa-alueita. Vaikka neuroverkon kouluttaminen on resursseille kallista, ei valmis verkko ole kovinkaan raskas koulutukseen verrattuna.

Neuroverkot on todettu universaaleiksi approksimaattoreiksi [15]. Tämä tarkoittaa sitä, että oikein rakennettu ja koulutettu verkko pystyy approksimoimaan minkä tahansa epälineaarisen funktion. Toisin sanoen kuva-analyysissä voitaisiin approksimoida funktio, joka laskee sisääntulosta (kuvasta) tuloksen, ja kangasmerkkien tapauksessa kertoisi mikä merkki on kyseessä.

Jotta kuvamateriaalia saataisiin tarpeeksi, on tutkimuksen suorittajien itse ottamien kuvien lisäksi tarpeen keinotekoisesti lisätä kuvamäärää. Tämä suoritetaan kopioimalla kuvia ja manipuloimalla näitä. Esimerkkejä manipulaatiosta voisi olla kiertää kuvaa eri kulmaan tai vaihtaa väritasapainoa. Koska kuitenkin riittävän kuvamateriaalin kerääminen tulee olemaan haastavaa, algoritmia voitaisiin testata myös vapaassa levityksessä olevilla, tutkimuskäyttöön tarkoitetuilla kuvatietokannoilla. Esimerkiksi käsialan tunnistukseen tarkoitettu MNIST-tietokanta olisi sopiva testikohde [16] varmistaa verkon toimivuus.

Neuroverkot perinteisesti koostuvat kolmesta kerrostyypistä. Sisääntulo-, piilo- ja ulostulokerroksista. Sisään- ja ulostulokerrokset on rajattu yhteen kappaleeseen molempia. Piilokerrokset ovat varsinaisia laskennallisia kerroksia ja niitä voi olla niin paljon kuin tarvitaan tai resurssit antavat myöten. Jokainen kerros koostuu neuroneista, joita myöskin voi olla tarvittava määrä. Mitä suurempi verkko on, sitä enemmän laskentavoimaa sillä on mutta sitä raskaampi siitä tulee kouluttaa ja ajaa. Neuroneita yhdistää toisiinsa painotetut tiet, jotka tekevät suurimman työn laskennassa ja joita muuttamalla verkko koulutetaan. Vuonna 2012 tehdyssä tutkimuksessa [17] monisarakkeinen syvä verkko onnistui tunnistamaan käsialaa melkein ihmisen tasolla ja liikennemerkkejä kaksinkertaisesti ihmistä varmemmin.

Vaikka neuroverkot ovatkin suoriutuneet varsin hyvin tunnistustehtävissä, eivät ne kuitenkaan ole erehtymättömiä. Erityisesti ne ovat varsin heikkoja antagonistisille

kuville, joissa pienillä mutta tarkoituksenmukaisilla pikseli muutoksilla pystytään muuttamaan neuroverkon luokittelua [18]. Tällainen tahallinen häirintä on vakava ongelma turvallisuutta vaativissa applikaatioissa kuten itse ajavilla autoilla, joilla ei ole varaa tunnistaa kohteita väärin. Tutkimuksen käyttökohteessa antagonistiset kuvat eivät kuitenkaan ole ongelma sillä tahallisesti aiheutettu virhe kuvantunnistuksessa ei aiheuta vahinkoa, jota tulisi välttää.

Kuvia ei kuitenkaan pysty sellaisenaan syöttämään neuroverkolle, koska sisääntulojen lukumäärä olisi käytännössä pikselimäärä, mikä olisi liikaa jopa supertietokoneelle laskettavaksi. Siksi on kehitetty konvoluutiopohjainen neuroverkko, joka kiinteyttää kuvan pieneen määrään tunnistettavia ominaisuuksia. Koska neuroverkot ovat osoittautuneet varsin tehokkaiksi ja ovat kohtalaisen suosittuja alalla, voisivat ne soveltua tutkimuksen käyttöön hyvin.

3. TYÖN TOTEUTUS

Työ toteutettiin kolmessa vaiheessa. Ensimmäinen vaihe koostui lähdemateriaalin keräämisestä. Materiaalia tarvittiin varsin mittavasti, joten sen keräämiseen käytettiin aikaa kohtuullisesti. Toinen vaihe sisälsi algoritmin opettamisen tunnistamaan eri kangasmerkkejä. Koulutusmenetelmän pienehkön satunnaisuuden, ja lisäkuvien keräämisen vuoksi, tämä vaihe jouduttiin toistamaan useasti. Viimeiseksi vaiheeksi jäi luonnollisesti algoritmin testaus ja tulosten käsittely.

3.1. Materiaali ja sen kerääminen

Ennen varsinaisen materiaalin keräämistä, oli tarpeen testata minkä tyyppiset kuvat sopivat algoritmille. Testauksessa käytettiin kagglea: datatieteeseen ja koneoppimiseen erikoistunutta nettisivua. Tarkemmin, käytettiin kagglessa vuonna 2013 järjestettyä kilpailua, jossa tavoitteena oli kirjoittaa algoritmi, joka tunnistaa kissoja ja koiria. Kilpailun datasetti sisältää harjoitus- ja koulutusdataa [19], joissa yhteensä 37 500, joista 25 000 koulutuskuvia ja loput harjoitukseen. Koska kissat ja koirat sisältävät paljon samoja piirteitä, pitäisi algoritmin selvitä verrattain yksinkertaisesta kangasmerkkitunnistuksesta helposti, mikäli se saavuttaa kohtalaisen tunnistusprosentin eläimillä.

Kangasmerkkejä käytetään paljon opiskelijoiden käyttämissä haalareissa, joita käytetään usein opiskelijatapahtumissa. Nämä tapahtumat kohdistuivat aluksi materiaalin, eli kangasmerkkikuvien, keräyspaikaksi. Neuroverkon ensimmäiseen versioon kangasmerkeistä kerättiin noin 100 kappaletta kuvia parista eri opiskelijatapahtumasta. Kuvien ottamiseen käytettiin Motorolan Moto G 3rd generation -puhelinta. Materiaali oli hajautunutta usean erilaisen kangasmerkin välillä. Niistä valikoitiin 10 eniten kuvattua merkkiä, jotka ovat näkyvät kuvassa 2. Eniten kuvia yhdestä merkistä oli 11 kappaletta.

Kuva 2. Ensimmäiseen iteraatioon käytetyt kymmenen kangasmerkkiä.

Kuvien keräämisen ja valikoimisen jälkeen kuvat käytettiin kuvanmuokkausalgoritmin läpi. Algoritmi on ohjelmoitu Python-kielellä ja siinä käytettiin lähinnä Pillow/PIL-kirjastoa, joka oli ladattavissa verkosta. Aluksi algoritmi rajasi kuvan, sillä ohjelmoitu neuroverkko vaati neliönmuotoisia kuvia. Tällä myös vähennettiin turhan datan käsittelyä. Tämän jälkeen algoritmi pyöritti kuvaa 45 asteen verran ja tallensi kuvan kangasmerkin nimen mukaan numerojärjestyksessä. Toiminto suoritettiin samalle kuvalle siten, että yhdestä kuvasta saatiin tasaisesti kahdeksaan eri asentoon pyöritettyä kuvaa. Lisäksi kuvaa pyöritettiin vielä 345 asteeseen lisätutkimusmateriaalin saamiseksi.

Työn edettyä huomattiin lisämateriaalin tarve. Jotta saisimme mahdollisimman paljon kuvia yhtä merkkiä kohden, kuvaluokkien määrä rajattiin viiteen. Kuvia otettiin Honor 8 ja Samsung S6 matkapuhelimilla kummallakin lisää 50 kuvaa jokaisesta viidestä valikoidusta merkistä. Kuvat otettiin kuvan 2 merkeistä a, b, c, d

ja e, koska nämä olivat tarpeeksi erilaisia toisistaan. Kuvat käytettiin sen jälkeen edellä mainitun algoritmin läpi, jonka jälkeen kuvia oli yhdestä merkistä 900 kappaletta.

Tunnistusprosenttia varten jaettiin kuvatietokannasta kansiot eri algoritmin versioille, jossa opetukseen käytettyjä kuvia oli 1000, 2000 jne. Jos versiossa oli esimerkiksi 1000 kuvaa opetukseen varattuna, käytettiin loput kuvat kyseisessä versiossa neuroverkon tunnistusprosentin määrittämiseen.

3.2. Neuroverkkojen toteutus

Neuroverkon toteutuksessa suurin varsinainen työ on saada Caffe-kehitysympäristö toimintaan. Caffe on tehokas ympäristö, mutta sillä on paljon riippuvuuksia, joista osa on sangen vanhoja ja vähän tuettuja. Neuroverkkojen koulutus vaati paljon rinnakkaista laskentaa, joten oli tärkeää, että laskenta saatiin siirrettyä näytönohjaimelle sillä ne ovat luotuja suorittamaan useita rinnakkaisia laskuja. Työssä käytettiin kannettavaa konetta osittain resurssien puutteen takia mutta myös siksi että neuroverkon opetus syö paljon aikaa. Oli siis myös tärkeää, että käytössä oli tietokone, joka voitiin omistaa työlle keskeytyksettä.

3.2.1. Vaatimukset

Konenäkösovellus on rakennettu Berkeleyn Caffe kehitysympäristön ympärille, joka vaatii toimiakseen monia ohjelmistoja. Osa näistä on kohtalaisen vanhoja ja ympäristön rakentamiseen suositellaankin Linux-pohjaista järjestelmää, joka tukee näitä paremmin. Edellä käydään läpi vaatimukset yksityiskohtaisemmin.

Caffe [20] on syväoppimisteknologia, joka on luotu Berkleyn yliopiston AI tutkimusryhmän ja vapaaehtoisten voimin. Teknologian hyviin puoliin lasketaan muun muassa se, että malleja ei tarvitse koodata vaan ne luodaan asetuksien kautta, ja prosessointi voidaan siirtää CPU:n ja GPU:n välillä yhden lipun kautta. Lisäksi Caffen suosio tutkijoiden ja harrastajien joukossa takaa sen, että malleja on paljon tarjolla, vähentäen suunnitteluun käytettävää aikaa. Caffe käyttää LMDB:tä (Lightning Memory-Mapped Database) datan syöttämiseen verkkoon. LMDB:llä on todettu Bahrampourin julkaisun mukaan olevan muiden kehitysympäristöjen tietokantoja parempi suorituskyky [21], mikä on suotavaa isoja tietokantoja analysoitaessa.

CUDA (Compute Unified Device Architecture) on Nvidian luoma rinnakkaislaskenta-alusta, joka mahdollistaa grafiikkaprosessorin käytön yleislaskentaan. CUDA-alusta on pakollinen Caffen toiminnan mahdollistamiseksi, mikäli halutaan ulkoistaa laskenta GPU:lle. Koska CUDA on Nvidian alusta, asettaa tämä laite rajoituksia tutkimukselle, sillä vain Nvidian näytönohjaimet toimivat CUDA:lla, ja niistäkin vain osa.

BLAS (Basic Linear Algebra Subprograms) sisältää matalan tason lineaarialgebran rutiineja, kuten pistetulo sekä matriisikertolaskuja. Neuroverkot yleisesti vaativat tämänkaltaista algebraa, joten Caffe vaatii BLAS-kirjastoa toimiakseen.

Open Source Computer Vision eli OpenCV on kirjasto ohjelmointifunktioita, jotka keskittyvät konenäköön. Ei pakollinen Caffen toimintaa varten, mutta jotkut sen ominaisuudet vaativat sitä. Lisäksi kuvia muokattiin OpenCV:tä käyttäen, ennen

niiden syöttöä neuroverkkoon. Muokkaus rajoittuu kuvien koon säätämiseen sekä histogrammin tasoitukseen.

Muita pienempiä mutta tärkeitä lohkoja ovat seuraavat. Protobuffit eli protocol buffers ovat mekanismeja datan sarjallistamiseksi. Glog on googlen lokitusmoduuli ja gflags on komentorivin lippujen prosessointiin. Hdf5 on monipuolinen datamalli kompleksisen datan esittämiseen ja pääohjelmisto kirjoitettiin python 3.5 käyttäen.

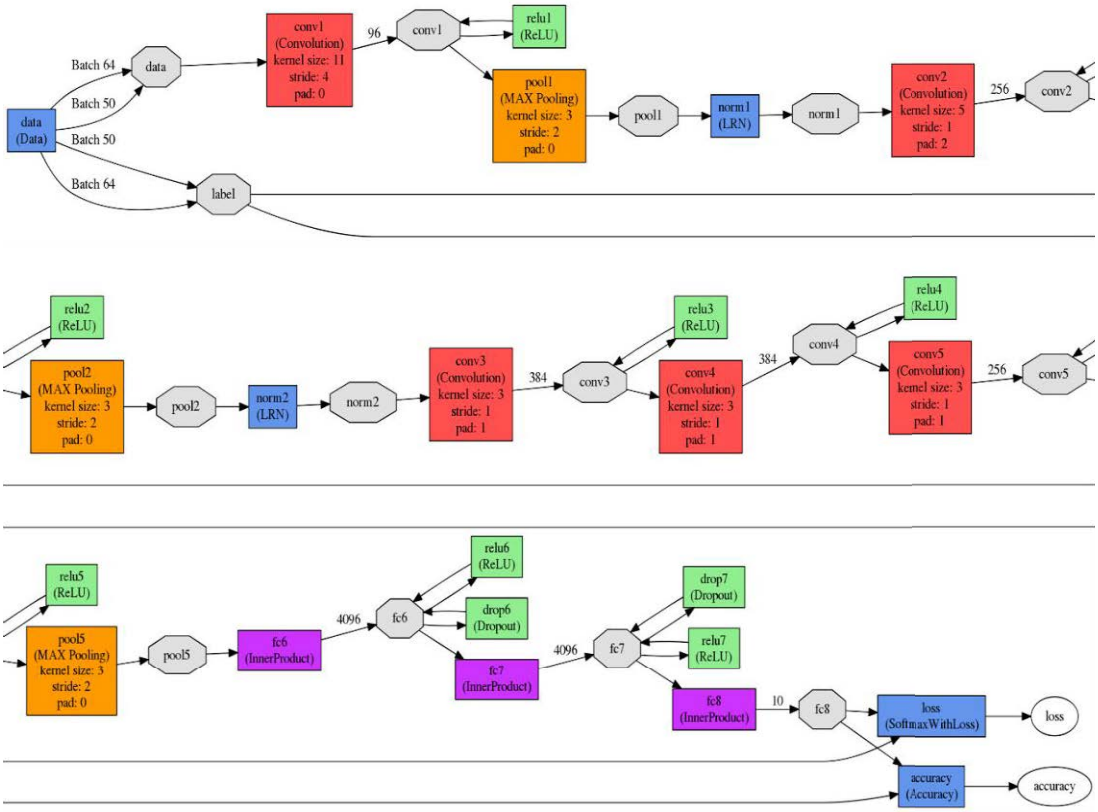
3.2.2. Neuroverkon koulutus

Koko algoritmin pohjana käytettiin Adil Moujahidin esittelemää neuroverkkoa kissojen ja koirien tunnistamiseen [22]. Ensimmäiseksi data valmisteltiin verkkoa varten. Tässä tapauksessa tämä tarkoittaa kuvien yhtenäistämistä, eli muokkaamista samaan kokoon ja histogrammin tasoitusta. Lisäksi Caffe käyttää LMDB:tä tietokanta tyyppiä, joten kuvat täytyi myös siirtää kyseiseen tietokantaan. Tietokantoja tarvittiin kaksi, toinen koulutusta varten ja toinen koulutetun verkon testaamista varten. Kuvat jaettiin kantoihin 1:5 suhteella, testikannan ollessa pienempi. Tietokantoihin täytyi luonnollisesti tallentaa myös jokaisen kuvan nimike, jotta verkko pystyi oppimaan minkälainen kuva kuuluisi millekin nimikkeelle. Kuvien nimeäminen oli työmäärältään raskas vaihe, mutta ensimmäisen verkon kouluttamisen jälkeen tätä voitiin käyttää lisäkuvien nimeämiseen. Näin iteroimalla voitiin kouluttaa useita verkkoja, jotka saivat jokaisella kierroksella enemmän materiaalia kuin aiemmat versiot ja samalla kehittyivät kuvan tunnistamisessa. Tällä tavalla saatiin myös dataa siitä, kuinka kuvamäärä vaikutti verkon tunnistuskykyyn ja oppimisnopeuteen.

Tietokantojen luomisen jälkeen täytyi määritellä neuroverkon malli ja ratkaisija (solver). Malli tarkoittaa konvoluutioverkon arkkitehtuuria, eli montako kerrosta ja miten ne linkitetään toisiinsa. Ratkaisija taas on vastuussa verkon optimoinnista. Ratkaisija määritteli muun muassa testi-iteraatioiden määrän, testi-intervallin ja oppimisnopeuden. Konvoluutiomallit ovat yleensä varsin suuria, sillä kuvien tunnistaminen on kohtalaisen monimutkikasta. Lisäksi mallien luonti on osaltaan summittaista. Yleisesti hyvä malli löytyy kokeilemalla ja tähän ovat monet tutkijat käyttäneet paljon aikaa useiden yleisessä käytössä olevien mallien luomiseksi. Mallin luonnin hankaluuden vuoksi käytettiin valmista mallia. Työssä mallina oli käytössä Moujahidin käyttämä BVLC (Berkeley Vision and Learning Center) caffenet-malli, joka pohjautui Alexnet-malliin. Alexnet on kirjoitettu CUDA-yhteensopivaksi ja sisältää grafiikkaprosessorituen. Lisäksi Alexnet saavutti Imagenet Large Scale Visual Recognition Challenge -kilpailussa top-5 virheen 15,3 %. Tämä tekee Alexnetin johdannaisen kelvolliseksi käyttöön. Ratkaisija pohjautui samaten Moujahidin versioon. Neuroverkon visualisoitu malli näkyy kuvassa 3.

Itse koulutus suoritettiin Asuksen X550L kannettavalla tietokoneella. Tutkimuksen suuntaa-antavan luonteen vuoksi todettiin, että paikallinen kone on tehokkaampi nopeaan testaukseen eikä ole tarvetta suuritehoiseen palvelimeen. Kone sisälsi Intelin core i5 suorittimen ja Nvidian Geforce 740M grafiikkasuorittimen, joka oli yhteensopiva CUDA:n kanssa. Valitettavasti GPU rajoitusten vuoksi, koulutussarjan koko jouduttiin rajaamaan 64:ään kuvaan alkuperäisen 256 sijaan. Sarjalla tarkoitetaan yhdessä iteraatiossa käytettyjen kuvien määrää. Tämä ei vaikuta algoritmin tarkkuuteen mutta hidastaa iteraation validointia kuvien hitaamman lataamisen vuoksi. Geforce 740M sisälsi 2 Gt muistia, joka riitti hädin tuskin 64 kuvan yhtäaikaiseen käsittelyyn. Neuroverkko tallensi version itsestään 5000

iteraation välein. Alustavasti verkkoa koulutettiin 40 000 iteraatiota. Ensimmäisen neuroverkon version tulosten perusteella iteraatio määrä muutettiin sopivammaksi 10 000 kierrokseen.



Kuva 3. Neuroverkon malli visualisoituna.

3.2.3. Neuroverkon versiot

Jotta saataisiin kattavaa tietoa neuroverkon oppimisesta, verkosta koulutettiin useampi versio. Versioita tehtiin viisi kappaletta ja luontijärjestyksessä ne nimettiin: V1, V1-1, V2, V3 ja V4. Versioista kerätty data on luettavissa taulukosta 1 sivulla 19. Ero versioiden välillä on ainoastaan koulutuskuvien määrä sekä V1:n ja V1-1:n kohdalla luokkien määrä. Tämä johtui rajallisesta kuvamäärästä. V1 koulutettiin alkuperäisellä 1153 kuvalla ja testattiin koko kuvakokoelmalla sisältäen 4569 kuvaa. Versio 1-1 koulutettiin samoilla kuvilla mutta koulutuskuvat poistettiin testikuvista. Versiot 2, 3 ja 4 sisälsivät kuvia vain viidestä eri luokasta, eli kangasmerkistä. Kuvat näihin versioihin olivat satunnaisotanta kaikista 4572 kuvasta kasvavassa määrin niin, että versio 2 sisälsi 1000 kuvaa koulutusvaiheessa, versio 3 sisälsi 2000 ja versio 4 sisälsi 3000. Ylijääneet kuvat käytettiin neuroverkkojen testaukseen.

3.2.4. Neuroverkon testaus

Jokaisen koulutusiteraation välissä ajetut testauskierrokset käyttivät koulutuksesta ylijääneitä kuvia. Neuroverkko teki arvauksensa jokaisesta testimateriaalissa olevasta

kuvasta ja tarkasti oliko arvaus oikein. Näin algoritmi laskee virheen ja käyttää tätä itsensä kehitykseen. Tuhannen iteraation välein algoritmi laski myös osumatarkkuutensa sekä raportoi tämän sekä virheen että oppimisnopeuden lokitiedostoon.

Koulutusiteraatioiden välillä tallennetuista lokeista tehtiin myös kaaviot algoritmin oppimiskäyrästä. Kaavio näyttää oppimiskäyrän koulutuskierrosten perusteella. Kuvat 4, 5, 6, 7 ja 8 näyttävät oppimiskäyrän lisäksi myös testi- sekä harjoitushäviön. Häviöt ovat iteraation aikana laskettujen virheiden summa joko harjoitus- tai testivaiheessa nimensä mukaisesti ja tämä halutaan minimoida koulutuksen aikana.

Koulutetun neuroverkon testaamiseen tultiin myöskin tarvitsemaan kuvia. Kuvien tuli olla samoista kangasmerkeistä, joita käytämme koulutuksessa, sillä neuroverkolla ei ollut luokitusta “joku muu”, vaan se riippumatta annetun kuvan sisällöstä luokittelee sen johonkin koulutuksessa määriteltyyn luokkaan. Kuvia ei tarvittu yhtä paljon kuin koulutusvaiheessa. Sadan kuvan arvioitiin riittävän tarpeeksi tarkan osumaprosentin määrittämiseen. Tässä vaiheessa ei voitu käyttää koulutusvaiheeseen käytettyjä kuvia, joten testaukseen käytettiin siihen varattuja ylijäämäkuvia.

Testi kirjasi ylös myöhempää analyysiä varten yleisen tunnistusprosentin, yksittäisen merkin osumat, sekä virhemarginaalin. Väärin arvatut kuvat arvauksineen myös kirjattiin erilliseen tiedostoon. Testi kirjasi myös kolme suurimmalla todennäköisyydellä olevaa arvausta jokaisen arvatun kuvan kohdalla. Näitä voitiin käyttää virheiden syiden arviointiin. Testiprosessissa neuroverkolle kerrottiin kuvan nimen perusteella, oliko sen arvaus oikein.

3.2.5. Tulosten käsittely

Kangasmerkeistä etsittiin piirteitä jotka neuroverkko löysi helposti. Yritettiin myös selvittää mitkä piirteet olivat hankalampia tunnistaa. Esimerkiksi useat kangasmerkit ovat pyöreitä ja sisältävät tekstiä, joten hypoteesi oli, että nämä menisivät neuroverkolla keskenään helposti sekaisin. Toisaalta oli myös hyvin uniikkeja merkkejä kuten kulmikkaat “pikselireunaiset” merkit kuten kuvan 2 b ja d. Ajateltiin, että näiden tunnistaminen olisi neuroverkolle helpompaa. Tuloksissa vertailtiin myös koulutuskuvien määrän ja neuroverkon koulutusiteraatioiden määrän vaikutusta neuroverkon oppimiskäyrään.

4. POHDINTA

Työn aikana saatiin käsiteltävää dataa hyvin paljon. Tämän käsittelyyn käytettiin ajallisesti eniten aikaa. Kun tuloksia alettiin käsitellä ensimmäistä kertaa, huomattiin tarve useammalle neuroverkon versiolle. Tulosten käsittelyn jälkeen todettiin paljon kehitysmahdollisuuksia ja ideoita tulevaisuudelle.

4.1. Lopputulokset

Tuloksia saatiin uusien neuroverkon versioiden myötä enemmän. Dataa kerättiin oikein ja väärin menneistä neuroverkon arvioista sekä kokonaisarvausprosentista. Taulukosta 1 nähdään oleelliset kerätyt tiedot neuroverkkojen versioista.

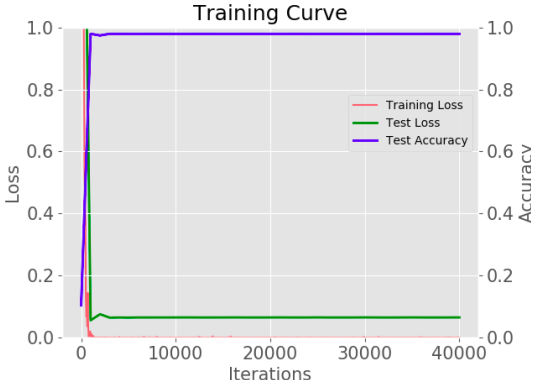
Taulukko 1. Neuroverkkojen koulutusten ja testausten jälkeen koottu data

Neuroverkon version nimi	V1	V1-1	V2	V3	V4
Koulutuskuva-hakemisto	Kuvat	Kuvat	kuvat1	kuvat2	kuvat3
Testikuvahakemisto	Kaikki	Testi1	testi2	testi2	testi3
Koulutuskuvien määrä	1153	1153	1000	2000	3000
Testauskuvien määrä	4569	3572	2572	2572	1572
Arvaustarkkuus koulutusvaiheessa	0,9792	0,9688	0,9701	0,9580	0,9800
Arvaustarkkuus todellisuudessa testausvaiheessa	0,3848	0,4490	0,9635	0,9526	0,9676
Koulutuskierrosten määrä	40 000	10 000	10 000	10 000	10 000
Koulutusluokkien määrä	10	10	5	5	5
Koulutuskuvia luokkaa kohden	100	100	200	400	600
Koulutusaika	27h 36min	7h 2min	7h 0min	7h 7min	7h 12min

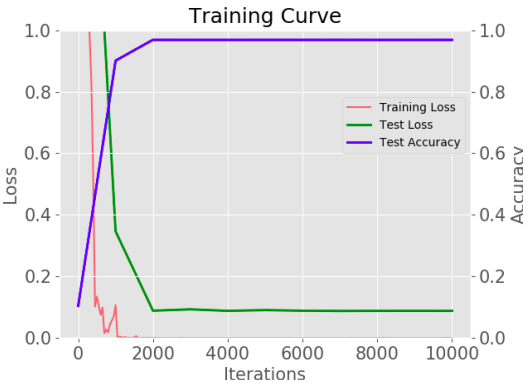
4.1.1. Oppimiskäyrä

Neuroverkon oppimiskäyrä oli hyvin lyhyt kaikissa testeissä (alle 10 000 kierrosta). Tämä todettiin jo 1. versiossa, jossa algoritmi käytettiin 40 000 kierroksen läpi. Lokeista kerätyn datan ja niistä piirretyn käyrän, kuvan 4, perusteella voidaan todeta, että 10 000 kierrosta riitti hyvin. 40 000 iteraation käyttäminen oli myös turhan hidasta suhteutettuna oppimiskäyrään (noin 27 tuntia). Kun myöhempiin versioihin käytettiin 10 000 kierrosta, jotka kestivät noin 7 tuntia. Ne olivat siis ajankäytöltään paljon tehokkaampia sillä oppimista ei enää juuri tapahtunut 4000 iteraation jälkeen. Iteraatioiden määrä olisi siis voitu lyhentää jo 5000 kierrokseen. V1 ja V1-1 huono arvaustarkkuus on osittain selitettävissä 10 eri luokan perusteella. Versioissa 1 ja 1-1

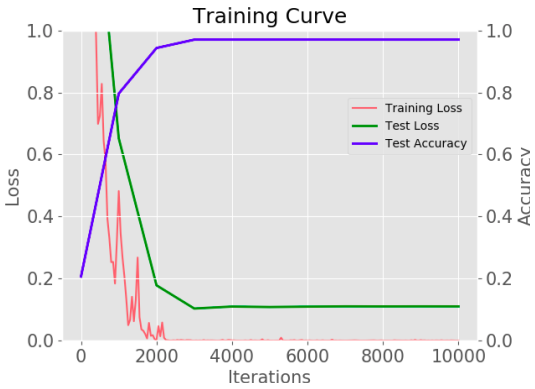
neuroverkolle oltiin koulutusvaiheeseen annettu koulutusmateriaalia kymmenestä eri merkistä. Määrä laskettiin tämän jälkeen viiteen eri merkkiin. Jälkimmäisissä neuroverkon versioissa oli siis 20% arvaustodennäköisyys ja ensimmäisessä kahdessa versiossa se taas oli 10%.



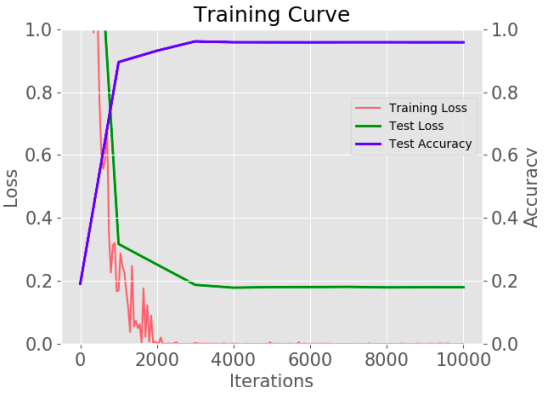
Kuva 4. V1.



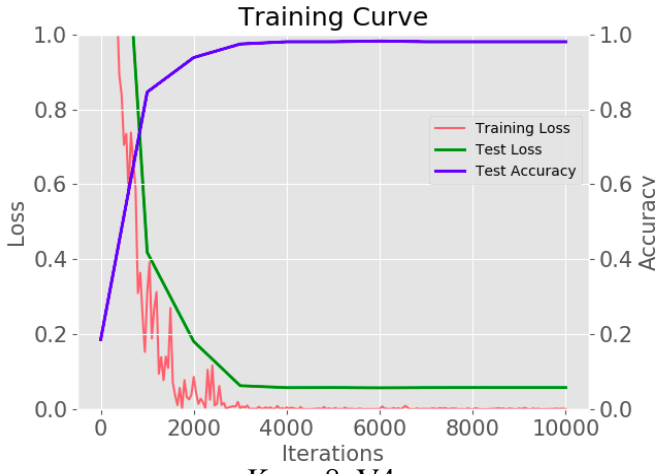
Kuva 5. V1-1.



Kuva 6. V2.



Kuva 7. V3.

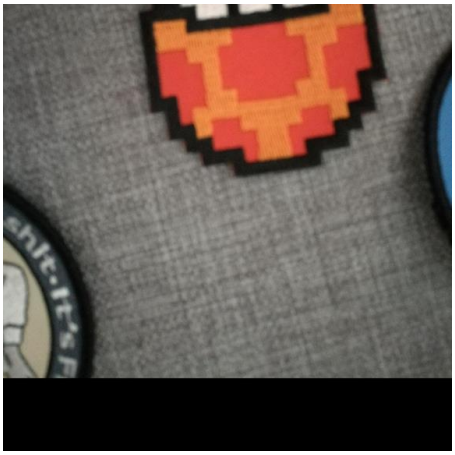


Kuva 8. V4.

4.2. Merkkien tunnistaminen

Dataa merkkien tunnistamisesta kerättiin paljon. Niiden tunnistus jouduttiin tekemään manuaalisesti katsomalla ja vertailemalla neuroverkon tunnistusprosentteja. Kyseessä oli kuitenkin kuvantunnistus, tutkittiin koneen kykyä tunnistaa kuvia verrattuna ihmisen kykyyn.

Datasta pystyttiin erottamaan selkeitä piirteitä esimerkiksi värin vaikutuksesta neuroverkon tunnistuskykyyn. Kuvan 2 merkit a ja e esimerkiksi tunnistuivat keskenään usein väärin. Samanlaisuuksia näistä nähdään esimerkiksi pyöreä muoto sekä tekstiä merkissä. Myös merkkien värimaailma sinisen sävyissä on ihmissilmälle samankaltainen. Värimaailman vaikutus näkyi myös, kun algoritmi arvasi merkkiä b merkiksi c ensimmäisissä versioissa enemmän muihin merkkeihin verrattuna. Määrät vääristä neuroverkon arvauksista eivät olleet yksinään riittävä periaate tehdä päätelmiä. Jotkin kuvista olivat hyvinkin hämääviä jopa ihmissilmälle, kuten kuva 9.



Kuva 9. Yksi käytetyn materiaalin kuvista. Kuva on tarkoitettu kuvaamaan yläreunassa olevaa kangasmerkkiä.

Kuvassa 9 oli alun perin tarkoitettu kuvamaan merkkiä b mutta kuvassa näkyvät myös toisten koulutusluokkien merkit c ja e reunat. On siis ymmärrettävää, mikäli neuroverkko olisi tunnistanut kuvasta minkä tahansa kolmesta. Algoritmia ei kuitenkaan opetettu käsittelemään vastaavia tapauksia, joten mikäli neuroverkon arvaus kuvalle ei ollut “sieni” niin sen vastaus laskettiin vääräksi.

5. JATKOKEHITYS

Tutkimus antoi hyvän pohjan jatkokehitykselle vahvistamalla algoritmin toiminnan ja antamalla suoraan dataa kuva vaatimuksista ja koulutus ajasta mikä voidaan suoraan johtaa tulevaisuuden resurssivaatimuksiin. Jatkokehityksen suurimpia haasteita tulevat olemaan kuvien määrän kasvattaminen sekä käyttöliittymän kehitys.

5.1. Jatkokehityksen haasteet

Työn suorituksen aikana huomattiin kohtia, joita voidaan tulevaisuudessa ottaa huomioon ja jatkokehittää. Näitä ovat esimerkiksi otettujen kuvien laatu ja määrä sekä muokattujen kuvien laatu. Myös tekniikan laskentatehoa tulee nostaa kuvamäärän kasvaessa.

Kuvienmuokkausalgoritmin parempi optimointi olisi voitu ottaa käyttöön. Kuvien määrää olisi voitu lisätä myös muokkaamalla niiden kirkkautta tai sävyjä. Valaistus vaikuttaa aina kameroiden kuvien laatuun sekä värimaailmaan normaaleissakin olosuhteissa. Kuvien pyörityksetkin ovat jälkeenpäin katsottuna epämääräisiä sillä moniin kuviin syntyi paksuja mustia reunoja. Näitä reunoja löytyi tosin kaikkien merkkien koulutusdatasta, joten teknisesti ei sillä täytyisi olla suurta merkitystä. Vastaavia reunoja tosin harvoin löytyy normaaleista kuvista, joten ne ovat voineet vääristää neuroverkon oppimaa kuvaa kaikista merkeistä. Selvä kehityskohde tässä olisi siis ensin leikata kuvat neliön muotoisiksi ja sitten vasta suorittaa kuvien pyöritys. Kuvien mustat reunat ovat kuitenkin tässäkin tapauksessa luultavasti vaikuttava tekijä. Mustia alueita voitaisiin muokata esimerkiksi lisäämällä esimerkiksi kohinaa, mutta ongelman vaikutusta pitäisi ylipäätään tutkia erikseen.

Aiemmin mainittu kuvien määrän puute oli myös havaittavissa. Kuvien generoiminen ei ole realistista mutta helpotti huomattavasti lisädatan saamista. Kuvien manuaalinen kerääminen on aikaa vievää ja vaikeaa jos ne olisi kerätty oikeista tai yleisimmistä ympäristöistä, eli opiskelijahaalareista. Materiaalin tehokkaampi kerääminen olisi siis myös jatkotutkimusta varten hyödyllistä.

5.2. Tulevaisuus

Koska kangasmerkkien tunnistus esitetyllä mallilla onnistuu kohtalaisen hyvin, olisi tulevaisuuden ehkä lupaavin kehityssuunta valjastaa sosiaalinen media kuvakirjaston laajentamiseen. Algoritmi voisi olla jatkuvassa koulutuksessa serverillä, johon käyttäjät pystyisivät lataamaan kuvia esimerkiksi chat-palvelulla, kuten Telegram-applikaatiolla. Tällöin serveri vastaisi oman arvauksensa kuvan merkistä ja käyttäjä vastaisi takaisin oikean merkin. Tällä tavalla kuvakantaa pystyttäisiin kasvattamaan automaattisesti ajan kanssa.

Toinen kehityssuunta voisi olla käyttää olemassa olevaa materiaalia ja tutkia mahdollisuutta kuvantunnistukseen ohjaamattomassa oppimisessa. Tällaisessa syväoppimisen menetelmässä kuvia ei luokitella etukäteen vaan algoritmi oppii itse löytämään yhtäläisyydet kuvista. Lähtökohtaisesti tämä on vaikeampaa, mutta mielenkiintoinen oppimistapa.

Joka tapauksessa tutkimus osoittaa, että käytetty algoritmi kykenee riittävällä tarkkuudella tunnistamaan kangasmerkkejä, jotta sen pohjalta voitaisiin tehdä käyttäjille suunnattu sovellus merkkien tunnistamiseen. Ainoa rajoittava tekijä on

tarvittava kuvamäärä, mutta kuvien hankintaa voidaan helpottaa ylempänä mainitulla tavalla. Luultavimmin järkevin tapa toteuttaa sovellus olisi nettipohjaisena siten, että itse tunnistus algoritmi on serverillä. Tämän päivän puhelimissa lienee tarpeeksi tehoja algoritmin ajamiseen, mutta mikäli on tarkoitus jatkuvasti kehittää verkkoa, täytyisi uusi versio siitä ladata erikseen käyttäjille. Tämä taas olisi työläämpää kuin koko algoritmin pyörittäminen serverillä.

6. PROJEKTIN KUVAUS

Työ tehtiin osana kurssia 521275A Sulautettujen ohjelmistojen projekti kolmen hengen ryhmänä. Projekti aloitettiin marraskuussa 2016 ja työn aihe saatiin päätettyä. Työ jatkui alkututkimusta tehdessä ja tutustuessa neuroverkkoihin sekä muihin mahdollisiin kuvantunnistusmenetelmiin. Materiaalin etsinnässä käytettiin useampia työtunteja. Varhainen versio tutkimuksessa käytetystä algoritmista saatiin kasaan syksyllä 2017. Tutkimusryhmän pienen koon vuoksi tutkimus meni pitkälle tauolle tutkimusryhmän jäsenten muiden kiireiden vuoksi. Syksyllä 2018 suoritettiin työn katsaus ja tutkimusraportti edistyi tuolloin nopeasti. Suurin osa tutkimustuloksista saatiin kokoon myös tällöin. Keväällä 2019 työ saatiin lopulta päätökseen, kun viimeiset tulokset oli saatu käsiteltyä ja raportti kirjoitettiin loppuun. Projektin hajanaisen työrytmin vuoksi työtuntien kertymistä ei onnistuttu kirjaamaan ylös.

7. YHTEENVETO

Tutkimuksessa haettiin sopivaa menetelmää tunnistaa kangasmerkkejä pienen kuvamateriaalin perusteella. Tulokseksi saatiin neuroverkko, joka pystyi tunnistamaan testien perusteella viittä eri kangasmerkkiä noin 95% todennäköisyydellä. Materiaalista suuri osa oli itse tuotettu alkuperäistä materiaalia hyväksi käyttäen. Tällaista menetelmää todennäköisesti tarvittaisiin myös lopullisen sovelluksen toteutuksessa, koska sovelluksen käyttäjäkunnalta on vaikea saada kerralla yhdestä merkistä 100 erilaista kuvaa. Tutkimus täytti haetut tavoitteet ja projektiin suunniteltu sovellus jää odottamaan tulevaisuuden jatkokehitystä.

LÄHTEET

[1] Azuma, R.T (1997) A survey of augmented reality. Teleoperators and Virtual Environments 6, 4 (August 1997), 355-385.

[2] Krevelen D.W.F, Poelman R (2010) A Survey of Augmented Reality Technologies, Applications and Limitations. International Journal of Virtual Reality, 2010, 9(2):1-20.

[3] Parvat A, Chavan J, Kadam S, Dev S, Pathak V. (2017) A survey of deep-learning frameworks. 2017 International Conference on Inventive Systems and Control (ICISC), Coimbatore, 2017, pp. 1-7.

[4] Pfister T, Li X, Zhao G, Pietikäinen M. (2011) Recognising spontaneous facial micro-expressions.

[5] Zhaoa G, Huanga X, Taini M, Li S.Z, Pietikäinen M. (2011) Facial expression recognition from near-infrared videos.

[6] Ylioinas J, Hadid A, Hong X, Pietikäinen M. (2013) Age Estimation Using Local Binary Pattern Kernel Density Estimate. A. Petrosino (Ed.): ICIAP 2013, Part I, LNCS 8156, pp. 141–150, 2013. c Springer-Verlag Berlin Heidelberg 2013.

[7] Nee A.Y.C,Ong S.K, Chryssolouris G, Mourtzis D (2012) Augmented reality applications in design and manufacturing. CIRP Annals - Manufacturing Technology 61 (2012) 657–679.

[8] Pinto N, Cox D.D, DiCarlo J.J (2008) Why is Real-World Visual Object Recognition Hard? Plos Computational biology.

[9] Rosenfeld A (1969) Picture Processing by Computer. Journal ACM Computing Surveys (CSUR) Surveys Homepage archive. Volume 1 Issue 3, Sept. 1969.

[10] Cheng H.D, Jiang X.H, Sun Y, Jingli Wang (2001) Color image segmentation: advances and prospects

[11] Belongie S, Malik J, Puzicha J (2002) Shape matching and object recognition using shape contexts. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 4, pp. 509-522, April 2002.

[12] Eichel P.H, Delp D.J. (1990) Quantitative Analysis of a Moment-Based Edge Operator. IEEE Transactions on Systems, Man, and Cybernetics, vol. 20, no. 1, pp. 58-66, Jan.-Feb. 1990.

[13] Tremeau A, Borel N. (1997) A region growing and merging algorithm to color segmentation (1997) Patern Recognition volume 30 issue 7 July 1997

[14] Rumelhart D.E, Hinton G.E, Williams R.J (1985) Learning Internal Representations by Error Propagation.

[15] Leung F.H.F, Lam H.K, Ling S.H, Tam P.K.S (2003) Tuning of the structure and parameters of a neural network using an improved genetic algorithm. *IEEE Transactions on Neural Networks*, vol. 14, no. 1, pp. 79-88, Jan. 2003.

[16] LeCun Y, Cortes C, Burges C.J.C. MNIST (Modified National Institute of Standards and Technology database) dataset for handwriting. http://yann.lecun.com/exdb/mnist/ (Luettu 1.9.2017).

[17] Ciregan D, Meier U, Schmidhuber J (2012) Multi-column deep neural networks for image classification. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, Providence, RI, 2012, pp. 3642-3649.

[18] Tabacof P, Valle E. (2016) Exploring the space of adversarial images. 2016 International Joint Conference on Neural Networks (IJCNN).

[19] https://www.kaggle.com/c/dogs-vs-cats Datakokoelma koirista ja kissoista vuoden 2013 kagglen kilpailusta. (Luettu 3.9.2017)

[20] Yangqing J, Shelhamer E, Donahue J, Karayev S, Long J, Girshick R, Guadarrama S, Darrell T. (2014) arXiv preprint arXiv:1408.5093 Caffe: Convolutional Architecture for Fast Feature Embedding.

[21] Bahrampour S, Ramakrishnan N, Schott L, Shah M. (2016) Comparative study of caffe, neon, Theano and Torch for deep learning.

[22] Moujahid A. (2017) A Practical Introduction to Deep Learning with Caffe and Python.