Tampereen yliopisto

Matti Jaakkola

AVOIMEN LÄHDEKOODIN OPPIVA KONENÄKÖJÄRJESTELMÄ TEOLLISUUSKÄYTÖSSÄ

Diplomityö
Tekniikan ja luonnontieteiden tiedekunta
Tarkastaja: Prof. Minna Lantz
Tarkastaja: TkT Niko Siltala
Marraskuu 2022

TIIVISTELMÄ

Matti Jaakkola: Avoimen lähdekoodin oppiva konenäköjärjestelmä teollisuuskäytössä
Diplomityö
Tampereen yliopisto
Konetekniikan diplomi-insinöörin tutkinto-ohjelma
Marraskuu 2022

Tämä tutkimustyö on suoritettu ABB Oy:lle, sen Smart Power -yksikölle. ABB Smart Power valmistaa erilaisia sähkön tuotannossa, jakelussa ja käyttämisessä tarvittavia tuotteita. Työn tavoitteena oli tutkia avoimen lähdekoodin oppivan konenäköjärjestelmän soveltuvuutta teollisuuden käyttöön. Neuroverkot ja niiden konenäkösovellukset ovat kehittyneet viime vuosina merkittävästi. Kaupallisia neuroverkkoihin perustuvia konenäköratkaisuja on saatavilla, mutta myös avoimen lähdekoodin neuroverkkosovelluksissa on omat mahdollisuutensa. Tätä mahdollisuutta haluttiin tutkia tässä työssä.

Työn teoriaosuudessa tutustuttiin konenäön, neuroverkkojen ja avoimen lähdekoodin teorioihin, tutkimuksiin sekä esimerkkisovelluksiin. Neuroverkkoja on tutkimusten mukaan hyödynnetty erilaisiin kohteiden tunnistamisiin ja luokitteluihin sekä poikkeamien etsimiseen. Internetistä on saatavilla esimerkkisovelluksia kaikista näistä sovelluksista.

Työn käytännön osuudessa tuotannosta kartoitettiin kaksi käyttökohdetta haastattelujen ja kyselyiden avulla. Käyttökohteiksi valikoituivat tuotepakkauksen sisällön tarkistus -sovellus ja laaduntarkastussovellus. Näille käyttökohteille suoritettiin Proof-of-Concept-testit. Ensimmäisessä PoC-testissä, tuotepakkauksen sisällön tarkistuksessa, tutkittiin kahden eri neuroverkon soveltuvuutta. Molemmat käytetyt neuroverkot kykenevät tunnistamaan pakkaukseen kuuluvat komponentit luotettavasti, tunnistusvarmuuksien ollessa erinomaisia.

Toisessa PoC-testissä tutkittiin neuroverkkojen soveltuvuutta laaduntarkastuskäyttöön. Tavoitteena oli löytää neuroverkko, joka kykenee tunnistamaan minkä tahansa tuotteessa esiintyvän kosmeettisen virheen tai puutteen. Tässäkin testissä käytettiin kahta eri neuroverkkoa. Toinen osoittautui tavoiteltuun käyttöön soveltumattomaksi, mutta toinen toimi juuri halutulla tavalla. Käytetyn neuroverkon tunnistusvarmuus oli hyvä, ja se kykeni löytämään yhtä lukuun ottamatta kaikki virheet.

Proof-of-Concept-testeillä osoitettiin avoimen lähdekoodin konenäköjärjestelmän toimivuus ja jatkokehitysmahdollisuudet. Tutkimuksen perusteella voidaan todeta, että avoimen lähdekoodin oppiville konenäköjärjestelmille voidaan löytää sovelluksia teollisesta tuotannosta. Jatkokehitystä kuitenkin tarvitaan ennen kuin tässä tutkimuksessa käytetyt neuroverkot soveltuvat tuotantokäyttöön.

Avainsanat: Neuroverkot, konenäkö, koneoppiminen, tunnistaminen, laaduntarkastus

ABSTRACT

Matti Jaakkola: An open-source machine learning vision system in industrial use
Master of Science Thesis
Tampere University
Master's Degree Programme in Mechanical Engineering
November 2022

This research has been done for ABB Oy, the Smart Power unit. ABB Smart Power manufactures various products needed in the production, distribution, and use of electricity. The aim of this research is to examine how open-source machine learning vision system can be used in the industrial manufacturing. The neural networks and their machine vision applications have developed over the past few years. Commercial machine vision solutions based on neural network are available, but open-source neural network applications also have possibilities. This possibility is investigated in this work.

In the theoretical part of the work, the theories, studies and example applications of machine vision, neural networks and open-source code are examined. According to studies, neural networks have been used for various object recognition and classification applications, also for looking deviations from objects. Sample applications for all these applications are available on the Internet.

In the practical part of the work, two use cases were mapped from production using interviews and surveys. The selected cases were the application for checking contents of product packaging and the application of quality control. The Proof-of-Concept tests were performed for these cases. In the first PoC test, checking the contents of the product package, the applicability of two different neural networks were investigated. Both used neural networks can reliably identify the components belonging to the package, with excellent identification certainty.

The second PoC test investigated the suitability of neural networks for quality control use. The goal was to find a neural network capable of identifying any cosmetic error that appears in the product. In this test two different neural networks were used. One turned out to be unsuitable for the intended use, but the other worked exactly as desired. The recognition reliability of the neural network used was good, and it was able to find all but one of the errors.

Proof-of-concept tests demonstrated the functionality of the open-source machine vision system and the possibilities for further development. Based on the research, it can be concluded that applications for open-source machine vision systems can be found in industrial production. However, further development is needed before the neural networks used in this research are suitable for production use.

Keywords: Neural networks, machine vision, machine learning, recognition, quality control

ALKUSANAT

Tämän mielenkiintoisen tutkimustyön myötä sain hyvän katsauksen avoimen lähdekoodin neuroverkkoihin ja niiden konenäkösovelluksiin. Aihe on hyvin ajankohtainen. Neuroverkkojen käyttö tekee tuloaan myös teollisuuden automatisointiin.

Kiitos ABB Oy:n Smart Power -yksikön tuotannonkehitystiimille diplomityön aiheesta, ohjauksesta ja kannustuksesta. Kiitos myös tuotannon työntekijöille keskusteluista ja kommenteista aiheeseen liittyen.

Kiitos työn ohjaajille ja tarkastajille palautteesta ja kannustuksesta.

Kiitos perheelleni tuesta, ymmärryksestä, kannustuksesta ja ennen kaikkea kärsivällisyydestä tehdessäni tätä työtä.

Seinäjoella, 14.11.2022

Matti Jaakkola

SISÄLLYSLUETTELO

KUVALUETTELO

LYHENTEET JA MERKINNÄT

ABB SP	ABB Smart Power
PoC	Proof of Concept, konseptin toimuvuuden todennus
YuMi	ABB IRB14000 yhteistyörobotin tavaramerkki, myös puhekielessä käytettävä nimitys tälle robottimallille
YOLO	You Only Look Once, neuroverkkomalli
VGG	Visual Geometry Group, neuroverkkomalli
SSD	Single Shot Detector, neuroverkkomalli

1. JOHDANTO

Valmistavassa teollisuudessa tuotteiden laaduntarkastus on hyvin merkittävässä osassa. Konenäköä käytetään aina vain enemmän teollisuudessa erilaisiin tuotannollisiin ja laadunvalvonnallisiin tehtäviin. Tuotannollisissa käyttökohteissa konenäköä voidaan käyttää esimerkiksi kappaleen löytämiseen ja poimintapisteen määrittämiseen robotilla tehtävässä kokoonpanossa. Laadunvalvonnallisissa käyttökohteissa voidaan valmistettavasta tuotteesta etsiä virheitä, jotka johtaisivat tuotteen hylkäämiseen. [1]

Tekniikan kehittyminen ja sopivan hintaisten kameroiden parempi saatavuus on osaltaan johtanut optisen laaduntarkastuksen yleistymiseen. Ilman automatisoitua laaduntarkastusta laaduntarkastus on työntekijöiden vastuulla heidän normaalien työtehtävien lisäksi. Joissakin tapauksissa laadunvalvontaa voi suorittaa erikseen tehtävään nimetty henkilö. Ihmiselle laaduntarkastuksen suorittaminen on kuitenkin yksitoikkoista ja väsyttävää työtä, joten virheiden mahdollisuus kasvaa ihmisen väsyessä. Ihmisen näkökyvyssä on lisäksi rajoitteita, joita kameroissa ei välttämättä ole. Kuten esimerkiksi ihminen ei kykene näkemään infrapunavaloa, mikä taas kameroilla on mahdollista. Siksi automaattisella konenäöllä voidaan myös tehdä tarkistuksia mihin ihminen ei kykene. [1]

Tekoälyn tutkimus- ja käyttöönottoprojekteja on Suomessa useita. Työ- ja elinkeinoministeriön vetämä tekoälyn käytön edistämisohjelma ”Tekoäly 4.0” on ollut käynnissä vuodesta 2020. Sen tavoitteena on:

- edistää tuottavuutta ja kestävyyttä lisääviä digi-investointeja
- monipuolistaa teollisuuden ja palvelujen ekosysteemejä ja
- vahvistaa Suomen asemaa Euroopan strategisen autonomian lisäämiseen tähtäävissä toimissa [2]

”Tekoäly 4.0” ohjelman raportin mukaan [2] Suomessa tekoälyn käyttö on EU:n kärkipäätä. Suomalaisista yli 10 hengen yrityksistä 12 % käyttää tekoälysovelluksia, kun EU-maiden keskiarvo on 7 %. Tekoälyn käyttö on yleisempää suurissa yrityksissä ja yhtiöissä kuin pienissä ja keskisuurissa yrityksissä. Suomella on mahdollisuus toimia tiennäyttäjänä tekoälyn ja datan käytössä Euroopan unionissa.

Avoimen lähdekoodin ratkaisut tekoälyn ja tarkemmin sen konenäön saralla ovat mielenkiintoinen mahdollisuus teollisuuden konenäkösovellusten toteuttamiseen. Avoin lähdekoodi mahdollistaa käytettävän ohjelman muokkaamisen tarkoituksiin paremmin sopivaksi. Ohjelmien vapaa levitys ja käyttö mahdollistaa niiden käytön hyvin monipuolisesti erilaisiin kohteisiin ilman suuria hankintakustannuksia lisenssien muodossa.

Toisaalta jos käytettävän ohjelmiston kehitys ja ylläpito on henkilöstöllä itsellään, voi niistä muodostua merkittäviä kuluja tehtävään käytetyn työajan kautta. Avoin lähdekoodi vaatii henkilöstöltä osaamista ohjelmistokehityksestä ja eri ohjelmointikielistä. Toisaalta ongelmien ratkontaa varten tarvittava tuki on helposti saatavilla.

Avoimen lähdekoodin ohjelmistojen tietoturva on kuitenkin suuri kysymysmerkki. Riippuen koodin levitysalustasta se on kyllä tarkastettavissa ennen sen lataamista ja asentamista, jolloin voidaan varmistua, että se ei sisällä haittakoodia. Tämä vaatii kuitenkin ymmärrystä ja osaamista käytetystä ohjelmointikielestä sekä aikaa perehtyä koodiin.

1.1 Työn tausta

ABB Smart Power (jäljempänä ABB SP) valmistaa erilaisia tuotteita sähkön tuottamiseen, siirtämiseen ja käyttämiseen liittyviin sovelluksiin. Näitä ovat esimerkiksi kuormakytkimet ja vaihtokytkimet. Vaasassa sijaitseva Suomen tehdas työllistää n. 300 henkilöä ja vastaa kytkintuotteiden valmistuksesta, tuotekehityksestä, myynnistä ja markkinoinnista maailmanlaajuisesti.[3] Kuvassa 1 on esitettynä joitakin kytkinmalleja mitä Vaasan tehtaalla valmistetaan.

Kuva 1. *Joitakin ABB SP:llä valmistettavia kytkimiä [3]*

ABB SP:llä on tehty määrätietoista työtä tuotannon kehittämiseksi. Tuotanto on hyvin pitkälle automatisoitua ja esimerkiksi teollisuusrobottien määrä on moninkertaistunut viimeisen 10 vuoden aikana. Tuotannon automatisoinnilla saadaan tuotanto pidettyä Vaasassa.

Konenäön kehittymisen myötä hyvin moneen automaattiseen kokoonpanosoluun on asennettu konenäkökamera. Myös oppivaa tekoälyä käyttävä kamera on asennettu yhteen robottisoluun, ja sen käytöstä on saatu positiivisia kokemuksia. Kameralaitteistojen ja varsinkin tekoälykameran suuri hankintahinta on herättänyt ajatuksen, että voidaanko niitä korvata jollain muulla, halvemmalla laitteistolla. Edullisella laitteistolla toteutettu toimiva sovellus voidaan monistaa useampaan samankaltaiseen käyttökohteeseen. Edullinen hinta mahdollistaa konenäön käyttämisen myös tuotteen valmistusprosessin ulkopuolella, esimerkiksi valvomaan, että oikeat työkalut ovat käytettävissä.

Kysymys tarkentui aloituskeskustelujen myötä avoimen lähdekoodin konenäkösovelluksiin ja siihen mitä niillä voitaisiin toteuttaa teollisessa ympäristössä. Tästä muotoutuukin diplomityölle asetettu ongelma: kuinka avoimen lähdekoodin konenäköä voisi käyttää teollisuuden sovelluksissa.

1.2 Työn tavoitteet ja tutkimuskysymykset

Diplomityön tavoitteena oli tutkia erilaisia avoimen lähdekoodin oppivia konenäköohjelmistoja ja toteuttaa konseptin testaus muutamassa tuotannon kohteessa.

Tutkimukselle asetettiin tutkimuskysymykseksi:

1. Kuinka avoimen lähdekoodin oppivaa konenäköjärjestelmää voidaan hyödyntää teollisuudessa?

Apututkimuskysymyksiksi vuorostaan:

2. Miten oppivalla konenäköjärjestelmällä voidaan tunnistaa teollisuuden tuotteita?
3. Mitä on oppivan konenäön avoin lähdekoodi?

Tutkimuskysymys 1 on suoraan ongelman asetteluun yhteydessä. Haluamme selvittää kuinka avoimen lähdekoodin oppivia konenäköjärjestelmiä voidaan hyödyntää teollisuudessa. Tutkimuskysymykseen haetaan pohjatietoa kirjallisuustutkimuksella, nykytila-analyysillä, kyselyillä ja haastatteluilla. Tämän jälkeen tutkimuskysymyksen ratkaisuun rakennetaan testisovellukset, joilla saadaan käytännön kokemusta tutkimuskysymyksen aiheesta.

Apututkimuskysymys 2 tarkentaa tutkimuskysymyksen 1 ongelman asettelua tuotteiden tunnistamiseen. Tutkimustyö rajoittuu konenäön eri sovellusten analysointiin ja toimivien ratkaisujen löytämiseen.

Apututkimuskysymys 3 on myös selventämässä tutkimuskysymystä 1. Mitä oppivan konenäön avoimella lähdekoodilla ymmärretään. Miten se eroaa ei-avoimesta lähdekoodista?

1.3 Työn rajaus

Tutkimustyö rajattiin koskettamaan vapaasti saatavilla olevia tekoälyn avoimia lähdekoodeja. Tätä vielä tarkennettiin rajaamalla työ oppivaan konenäköön tarkoitettuihin lähdekoodeihin. Laitteiston osalta työ rajattiin siten että käytetään kuluttajille saatavilla olevia komponentteja. Laitteiston hankintahinta haluttiin pitää edullisena, jotta voidaan tarkastella mahdollisia käyttökohteita ilman ennakkoasetelmaa kalliista konenäköjärjestelmästä.

Koska tutkimustyön yhtenä lähtöajatuksena oli selvittää mihin kaikkeen edullista konenäköjärjestelmää voidaan hyödyntää, oli odotettavissa, että käyttökohteita voi löytyä paljon. Tarkempaan tarkasteluun valitaan näistä kaksi kappaletta. Valintaa varten eri käyttökohteita arvotetaan pääasiassa niiden tarpeellisuuden mukaan.

1.4 Tutkimuksen toteutus ja menetelmät

Diplomityössä käytettiin tutkimusmenetelmänä konstruktiivista tutkimusotetta. Konstruktiivisessa tutkimusotteessa ratkaistaan jokin reaalimaailman ongelma, käyttäen apuna teoreettista ja käytännöllistä tiedon hankkimista. Konstruktiivinen tutkimusote tuottaa ratkaisun, konstruktion, ongelmaan. Konstruktio on abstrakti käsite, jolla voi olla rajattomasti erilaisia toteumia. Lähes mikä tahansa voi siis olla konstruktio. Konstruktiot eivät ole löydettyjä vaan ne keksitään ja kehitetään.

Konstruktiivisen tutkimusotteen keskeiset piirteet ovat Lukan [4] mukaan:

- Ongelmien ratkaisu keskittyy tosielämän ongelmiin
- Tuotos on innovatiivinen konstruktio, jolla ratkaistaan alkuperäinen ongelma
- Tutkimukseen osallistuvien tutkijoiden ja käytännön työn tekijöiden läheinen yhteistyö, ja kokemusperäinen oppiminen
- Kytkeytyy jo olemassa olevaan teorioihin aiheesta
- Saatujen löydösten peilaus takaisin teorioihin

Konstruktiivisessa tutkimusotteessa tutkimusprosessi etenee Lukan mukaan [4] seuraavasti:

1. Etsitään ongelma eli tutkimusaihe.

   Ensimmäisessä vaiheessa etsitään ongelma, joka halutaan ratkaista. Paras ongelma on sellainen, jonka ratkaisulla on käytännön merkitystä.

2. Selvitetään tutkimusyhteistyön mahdollisuudet.

   Toisen vaiheen tutkimusyhteistyön selvityksessä tutkitaan mahdollisuudet pitkäaikaiseen tutkimusyhteistyöhön kohdeorganisaation kanssa.

3. Hankitaan teoreettista ja käytännöllistä tutkimusaiheen tuntemusta.

   Kolmannessa vaiheessa hankitaan tietoa tutkimusaiheesta. Tietoa hankitaan kirjallisesta aineistosta, havainnoimalla sekä haastattelemalla. Tutkimustyön tekijän on tiedettävä myös tutkimusaiheesta aikaisemmin julkaistut teoriat, jotta kehitystyötä voidaan tehdä sen perustalle. Myös tutkimuksesta saatavaa tulosta on verrattava aikaisempiin teorioihin. Näin saadaan tietoa tutkimustuloksen vaikutuksesta.

4. Innovoidaan ratkaisu ja kehitetään ongelmaan ratkaisu.

   Neljännessä vaiheessa keksitään ongelman ratkaisuun malli ja kehitetään ongelman ratkaiseva konstruktio. Konstruktion kehittämisen tulisi olla ryhmätyötä, jossa ovat mukana sekä tutkimustyön tekijä että asiakkaan työntekijät. Kehitystyössä käytetään sekä teoreettista että käytännön tietoa asiasta. Kehitystyön prosessi voi olla iteratiivinen ja siksi viedä paljon aikaa.

5. Toteutetaan ja testataan ratkaisu.

   Viidennessä vaiheessa toteutetaan ratkaisu ja testataan sen toimivuus. Tämä on eräs tärkeimmistä vaiheista konstruktiivisessa tutkimuksessa. Testauksessa koko tutkimusprosessin toimivuus testataan, ei pelkästään tekninen toteutus.

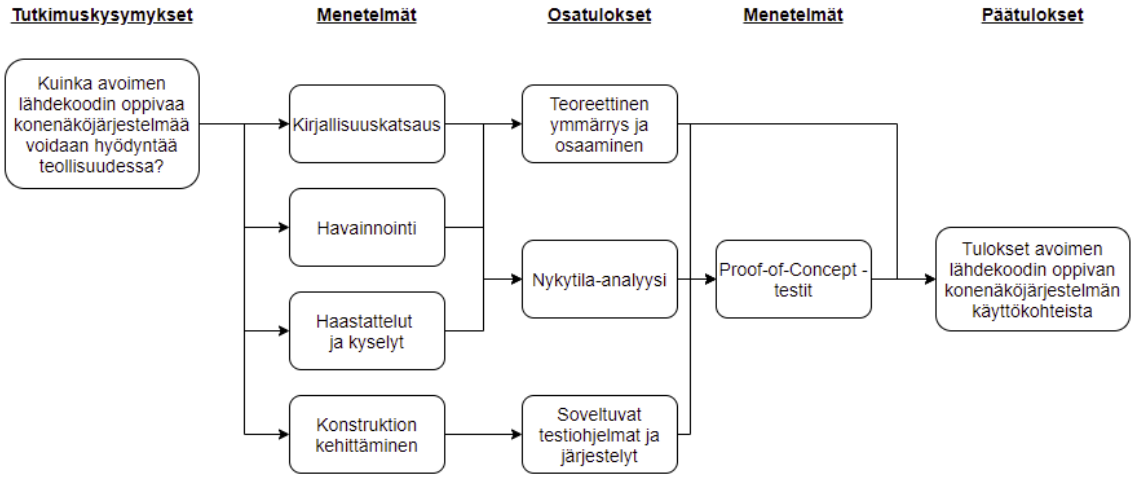
6. Pohditaan ratkaisun soveltamisalaa.

   Kuudennessa vaiheessa selvitetään mihin ratkaisua voidaan soveltaa. Innovoitua konstruktiota voi olla mahdollista soveltaa myös muihin tapauksiin. Tutkijan pitää ottaa etäisyyttä tutkimustyöhönsä ja tarkastella miten tutkimusprosessi on edennyt.

7. Tunnistetaan ja tutkitaan teoreettinen vaikutus.

   Seitsemännessä vaiheessa peilataan tuloksia aikaisempaan teoriaan aiheesta. Miten tutkimustulos suhtautuu teoriaan. Myös tässä vaiheessa tutkijan on tärkeää ottaa etäisyyttä tutkimustyöhönsä.

Tässä diplomityössä ratkaistava ongelma on kallis konenäköjärjestelmä. Kehitettävä konenäköjärjestelmä on uusi konstruktio. Konstruktiolla on myös käytännön vaikutusta tuotannon kehitykseen siitä saatavan tiedon perusteella. ABB SP:lla on intressejä tutkia ja kehittää konstruktiota tulevaisuudessa. Saatavaa konstruktiota on mahdollista muokata ja monistaa useaan eri käyttökohteeseen. Yhteistyö kohdeorganisaation kanssa toimi tutkimuksessa hyvin, aikaisemman yhteistyön toimiessa perustana. Ratkaisun innovoinnissa tehtiin yhteistyötä ABB SP:n tuotannonkehitystiimin kanssa. Saatua ratkaisua testataan käytännön testeillä. Testien jälkeen pohditaan mihin kaikkialle ratkaisua voidaan hyödyntää ABB SP:n tuotannossa. Myös peilausta aikaisempaan teoriaan suoritetaan pohdintaosion yhteydessä.



Kuva 2. *Tutkimuksen eteneminen kaaviona*

Työn teoriaosuudessa tutkimustyötä tehdään käyttämällä menetelminä kirjallisuuskatsausta, jossa perehdytään aihepiirin tutkimuksiin, artikkeleihin ja oppaisiin. Näillä hankitan tarvittavaa teoreettista osaamista. Työn käytännön osuudessa tutkimusmenetelminä käytetään havainnointia, haastatteluja ja kyselyitä sekä käytännön testejä. ABB SP:n tuotantoon suoritettiin nykytila-analyysi, jossa kerättiin tietoa havainnoimalla tuotantoa ja haastattelemalla tuotannon työntekijöitä. Lisäksi tietoa kerättiin myös kyselylomakkeilla. Näiden tietojen perusteella määritettiin käyttökohteet, joissa konstruktiota testataan käytännössä. Lisäksi kehitetään ratkaisuksi sopivaa konstruktiota tutkimalla ohjelmistoja ja määrittämällä niistä parhaiten käyttöön soveltuvat. Näiden menetelmien perusteella saadaan osatuloksina osaaminen ja ymmärrys aiheesta. Käytännön Proof-of-Concept-testien ja teoreettisen osaamisen perusteella saadaan sitten päätuloksena ymmärrys ja käyttökohde-ehdotukset tutkitulle järjestelmälle. Tutkimuksen eteneminen on kuvattu kuvassa 2.

1.5 Työn rakenne

Tämä diplomityö rakentuu teoreettisesta tutkimuksesta sekä käytännön osasta. Teoreettisen tutkimuksen osuudessa tehtiin ongelman kartoitus, kirjallisuuskatsaus, ja selvitettiin erilaisia ratkaisukeinoja ongelmalle. Käytännön osuus pohjautuu teoreettisessa osuudessa saatuun tietoon. Käytännön osuudessa suunnitellaan ja toteutetaan konstruktio, joka ratkaisee ongelman.

Luvussa 1 esitellään työn taustatiedot, tavoitteet, tutkimusmenetelmä sekä työn rakenne ja rajaukset.

Luvussa 2 käsitellään työtä koskettavaa teoriaa, keskittyen tekoälyyn ja konenäköön.

Luvussa 3 käsitellään käytännön sovellus, alkaen sen määrittämisestä, jatkuen laitteiston ja käyttökohteiden valinnalla sekä päättyen sovellusten esittelyyn.

Luvussa 4 esitellään työn tulokset.

Luvussa 5 käydään läpi päätelmät sekä kehitysehdotukset.

Luku 6 on yhteenveto tutkimuksesta.

2. TEORIA

Tutkittavana olevat käsitteet tekoäly ja konenäkö ovat hyvin ajankohtaisia teollisessa tuotannossa. Aihealueen teoriaan tutustuttiin kirjallisuuskatsauksella, jossa luettiin lukuisia artikkeleita aihepiiristä. Teoriaosuudessa käsitellään ensin yleisenä kattokäsitteenä tekoäly. Tämän jälkeen on koneoppimisen osuus. Koneoppiminen on tekoäly yksi osa-alueista. Seuraavaksi käsitellään neuroverkot, jotka ovat yksi koneoppimisen väline. Konenäön ja avoimen lähdekoodin teoria ovat täydentämässä tutkimustyön teoreettista pohjaa.

2.1 Tekoäly

Käsite tekoäly (engl. artificial intelligence, AI) esitettiin ensimmäisen kerran 1956 John McCarthyn toimesta. Sillä käsitetään koneen kykyä ajatella kuin ihminen. Tekoäly on hyvin laaja käsite, ja vaikka sitä on tutkittu jo vuosikymmenien ajan, ei oikeasti toimivaa tekoälyä ole vielä kehitetty. [6]

Alan Turing on ollut merkittävässä roolissa tekoälyn perusteiden luomisessa. Turing on esittänyt monia tekoälyn tutkimuksen peruskysymyksiä vuonna 1950 julkaistussa julkaisussaan. Turing kehitti niin sanotun Turingin testin, jota pidetään yhtenä ensimmäisistä määritelmistä sille, miten voidaan sanoa mikä on oikeasti tekoälyä. [7]

Tekoälyn kehitys on kokenut nousu ja laskukausia, mutta 2000-luvulle tultaessa neuroverkkoihin perustuvat syväoppivat algoritmit ovat osoittautuneet merkittäviksi kehitysaskeleiksi. Neuroverkkoja hyödynnetäänkin nykyisin monissa sovelluksissa. [6]

Tekoäly jaetaan nykyään kahteen eri tasoon, heikkoon tekoälyyn ja vahvaan tekoälyyn. Heikko tekoäly kykenee älykkäältä vaikuttavaan toimintaan mutta sillä ei kuitenkaan ole omaa tietoisuutta tai tahtoa. Vahvalla tekoälyllä olisi vuorostaan ihmisen kaltainen tietoisuus ja ymmärrys. Kaikki nykyajan tekoälyt ovat heikkoa tekoälyä eikä vahvaa tekoälyä ole vielä saavutettu. [6]

Suomen valtioneuvoston asettaman työryhmän vision mukaan Suomi on vuonna 2030 saavuttanut tunnustetun aseman tekoälyn vahvana edistäjänä ja osaamiskeskuksena.[2] Suomalaisilla yhtiöillä on vahva osaamispohja tietoliikenne- ja tietotekniikan alueilla, joita voidaan hyödyntää edistämään asemaamme tekoälyn osaajana.[7]

Tekoälyn sovelluksia

Tekoälysovelluksia kehitetään tällä hetkellä hyvin monella rintamalla. Puheentunnistus on yksi voimakkaasti kehitetty tekoälyn sovellus. Myös puheen kääntämisessä on saavutettu merkittäviä kehitysaskelia viime vuosina. [7]

Tekoälyn avustamat itseohjautuvat ajoneuvot ovat olleet jo pidemmän aikaa kehityksen kohteena. Tunnetuimpana kirjoitushetkellä lienee sähköautoja valmistavan Teslan kehitys autonomisen ajoneuvon saralla. Teslan Autopilot-tekoälyohjelmisto hyödyntää kameroita sekä neuroverkkoja analysoimaan mitä ajoneuvon ympärillä tapahtuu sen liikkuessa tiellä. [8]

B. Staarin [9] johtaman tutkijaryhmän jatkokehittämällä Triplets-neuroverkkomallilla saavutettiin pinnan laadun tarkastuksessa lupaavia tuloksia. Heidän johtopäätöksensä neuroverkon opetuksesta on, että jos opetuksessa käytetään myös kuvia pintatekstuureista pelkkien kappaleiden kohdekuvien lisäksi, voi neuroverkko oppia tunnistamaan viat paremmin.

Monet suuret teollisuuden ohjaus- ja tuotantolaitteiden valmistajat ovat aloittaneet omien tekoälysovellusten kehittämisen. Esimerkiksi Fanuc, joka valmistaa mm. teollisuusrobotteja, on ottamassa tekoälyä käyttöön robottien opettamiseen. Fanucin tavoitteena on kehittää teollisuusrobotti, joka itse oppii parhaiten toistetuista suoritteista. Lisäksi useampi robotti voisi jakaa oppimisen, jolloin robotit oppivat yhdessä nopeammin kuin yksittäinen robotti. [10]

Saksalainen monialainen teollisuusjätti Siemensin tavoitteena on kehittää järjestelmä, joka valvoo, tallentaa ja analysoi kaiken valmistuksen eri vaiheista. Tästä tiedosta voidaan löytää ongelmia ja ratkaisuja, joita ihminen ei löytäisi. [10][11]

2.2 Koneoppiminen

Koneoppimisella ymmärretään tekoälyohjelmistoa, joka kykenee oppimaan sille syötetystä datasta. Esimerkiksi internetin hakukoneet ovat koneoppivia, ja kykenevät tarjoamaan käyttäjälle hänen etsimiään tietoja perustuen aikaisempiin hakuihin. Koneoppiva laitteisto kehittyy paremmaksi saadessaan lisää dataa käsiteltäväksi. [6]

Koneoppimisen yleisimmät muodot ovat Hännisen [6] mukaan ohjaamaton oppiminen, ohjattu oppiminen, vahvistusoppiminen sekä syväoppiminen. Ohjatussa oppimisessa tekoälylle syötetään sekä opetusdata että myös siihen liittyvä oikea tulos. Tekoäly pyrkii löytämään opetusdatasta kohteita, ja vertaa niitä oikeisiin tuloksiin.

Ohjaamattomassa oppimisessa tekoälylle syötetään vain haluttua tunnistettavaa asiaa sisältävää opetusdataa. Tekoäly tunnistaa datasta säännönmukaisuuksia ja suorittaa päättelyn niiden perusteella. [6]

Vahvistetussa oppimisessa tekoälylle ei anneta oikeita vastauksia, vaan saadun vastauksen perusteella annetaan joko palkkio tai rangaistus. Näistä tekoäly oppii tekemään oikeita päätöksiä, vaikka syötetty data muuttuisi. [6]

Syväoppiminen on neuroverkkoihin perustuva koneoppimisen muoto ja se esitellään seuraavassa luvussa tarkemmin.

2.3 Neuroverkot

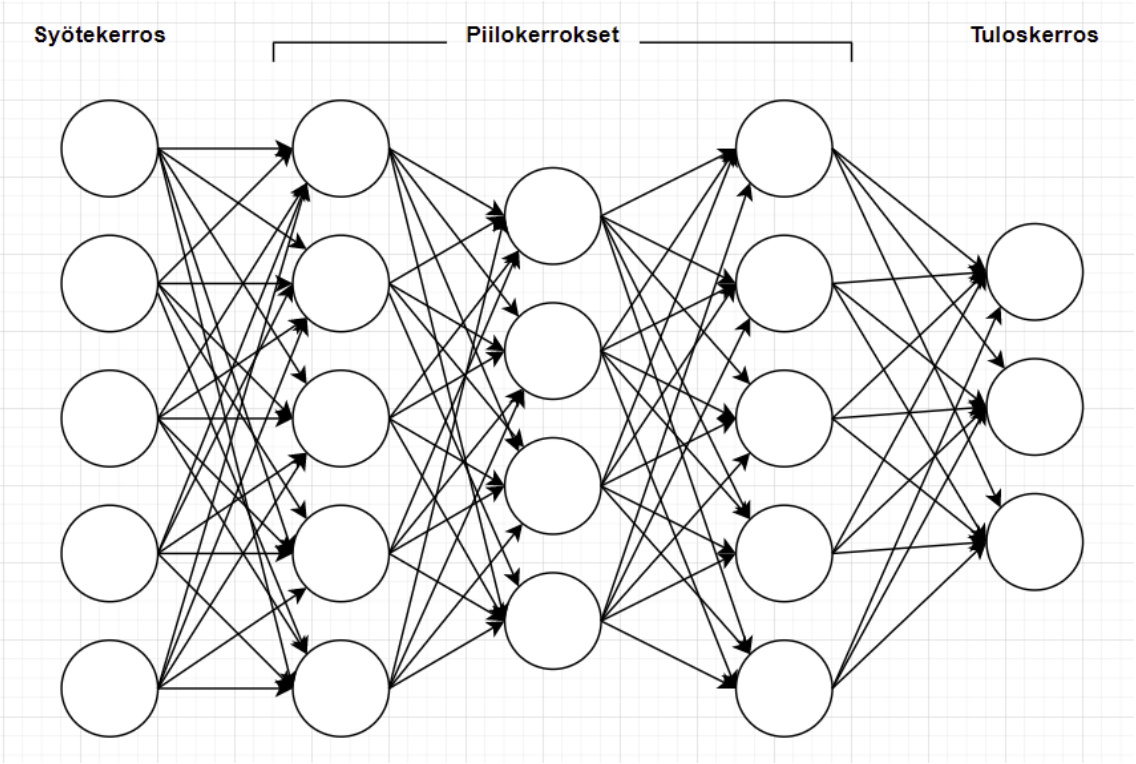
Syväoppiminen on koneoppimisen osa-alue, ja se perustuu neuroverkkoihin. Neuroverkot rakentuvat nimensä mukaisesti monikerroksisista neuroverkoista. Kerrokset rakentuvat yksittäisistä solmuista, neuroneista, jotka ovat yhteydessä toisiinsa. Neuroverkkoa ei ohjelmoida toimimaan tiettyjen algoritmien mukaisesti, vaan se oppii tuottamaan halutun lopputuloksen [6].

Neuroverkkojen hyviä ominaisuuksia Hännisen [6] mukaan hyvä laskentateho, vikasietoisuus, kyky jäsentää tietoa sekä kyky yleistää. Neuroverkkoja voidaan hyödyntää useissa erilaisissa käytännön sovelluksissa, esimerkiksi sääennusteiden laadinnassa, internetsivujen hakupalveluissa sekä itseohjautuvien ajoneuvojen reittien laskennassa.

Seuraavissa kappaleissa on esitelty mistä neuroverkko rakentuu ja minkälaisia neuroverkkoja on olemassa.

2.3.1 Neuroverkon rakenne

Neuroverkot voidaan ajatella kerroksittaiseksi rakenteeksi, jossa kerrokset ovat yhteydessä toisiinsa. Rakenne riippuu neuroverkon käyttötarkoituksesta, mutta perusperiaate on kaikissa sama.



Kuva 3. *Eteenpäin syöttävän neuroverkon rakenne. Mukaillen [12]*

Kuvassa 3 on esitetty neuroverkon rakenne. Ensimmäisenä on syötekerros, johon syötetään neuroverkossa käsiteltävä tieto. Tuloskerros vuorostaan kertoo neuroverkossa tehdyn tulkinnan tuloksen. Syötekerroksen ja tuloskerroksen välisiä kerroksia kutsutaan piilokerroksiksi. Nimitys piilokerros tulee siitä, että kerrokset eivät ”näy” ulospäin neuroverkosta. Piilokerroksia voi olla useita, riippuen mitä ominaisuuksia neuroverkolta halutaan. [12][13]

Neuronit ovat yksinkertaisia tietoa käsitteleviä yksikköjä, jotka ovat kytketty toisiinsa. Näitä kytkentöjä kutsutaan synapseiksi, aivojen hermojen kytkentöjen mukaan. Jokaiselle synapsille määritetään painoarvo, kerroin, jolla painotetaan kyseisen syötteen merkitystä. Kaikki neuronin syötteet kerrotaan painoarvoillaan ja lasketaan yhteen. Saatu summa sijoitetaan aktivointifunktioon. Aktivointifunktio määrittää aktivoituuko neuroni sen saamilla syötteillä, ja mikä on sen arvo.[14]

Neuronin suorittama laskutoimitus voidaan kuvata seuraavalla matemaattisella kaavalla:

$$ulostulo = f\left(\sum_{i}^{n} w_i x_i\right) + b$$

missä *f* on aktivointifunktio, x_i on syötteen arvo, w_i on kyseisen syötteen kerroin ja *b* on vakiotermi.

Neuroverkkojen kerroksittainen rakenne mahdollistaa kerroksien opettamisen muokkaamalla neuronien painoja. Opetettaessa haetaan jatkuvasti parempaa tunnistustulosta, jolloin neuroverkon antama tulos paranee. Neuroverkkojen opetetaan yleensä tietyllä rajatulla datalla, ja opetuskertojen välissä neuroverkon toimintaa testataan tunnetulla datajoukolla. Tuloksen perusteella neuroverkkoa säädetään toimivampaan suuntaan.

Neuroverkkoja on erilaisia malleja ja uusia kehitetään jatkuvasti. Seuraavassa on esiteltynä muutama erityyppinen neuroverkkomalli.

Perseptroni

Perseptronia voidaan pitää ensimmäisenä neuroverkon mallina. Persetronin mallin on ensimmäisen kerran esittänyt Frank Rosenblatt vuonna 1962. Käytännössä se on yksittäinen neuroni, jonka aktivointifunktiona on binääriaskelfunktio. Persetronia voidaan käyttää yksinkertaisissa luokittelijoissa, joissa tulokseksi riittää kyllä tai ei -tyyppinen vastaus. [14]

Eteenpäin syöttävä neuroverkko

Eteenpäin syöttävässä neuroverkossa kerrosten neuronit ovat kytketty suoraan seuraavan kerroksen syötteiksi. Kerrokset suorittavat laskutoimitukset omilla syötteiden painoilla, vakiotermeillä ja aktivointifunktioilla. Toisin kuin perseptronissa, aktivointifunktio voi olla muukin kuin binääriaskelfunkio. Aktivointifunktio valitaan neuronin tarkoituksen mukaan. Lopuksi tulos saadaan ulostulokerroksesta.[14]

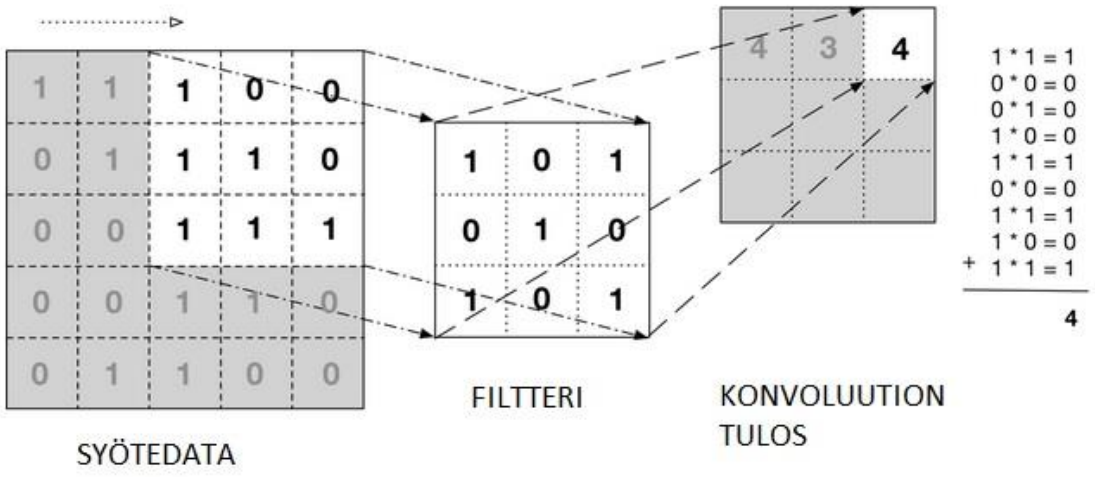
Konvoluutioneuroverkot

Konvoluutioneuroverkkoja (Convolutional neural networks, CNN) käytetään pääasiassa konenäössä, kuvantunnistuksessa ja piirteiden tunnistuksessa. Konvoluutioneuroverkon nimitys tulee erityisistä konvoluutiokerroksista. Konvoluutioneuroverkko poikkeaa tavallisesta neuroverkosta siten että siinä suoritetaan matemaattinen konvoluutio-operaatio tavallisemman matriisien kertomisen sijaan. [15]

Konvoluutioneuroverkkojen perusrakenne koostuu 3 eri kerrostumasta. Ensimmäisenä on syötekerros, jonka jälkeen tulevat tunnistuskerrokset. Lopuksi ovat luokittelukerrokset. Tunnistuskerrokset koostuvat konvoluutiokerroksista, ja näiden jälkeisestä näytteistyskerroksesta.[14]

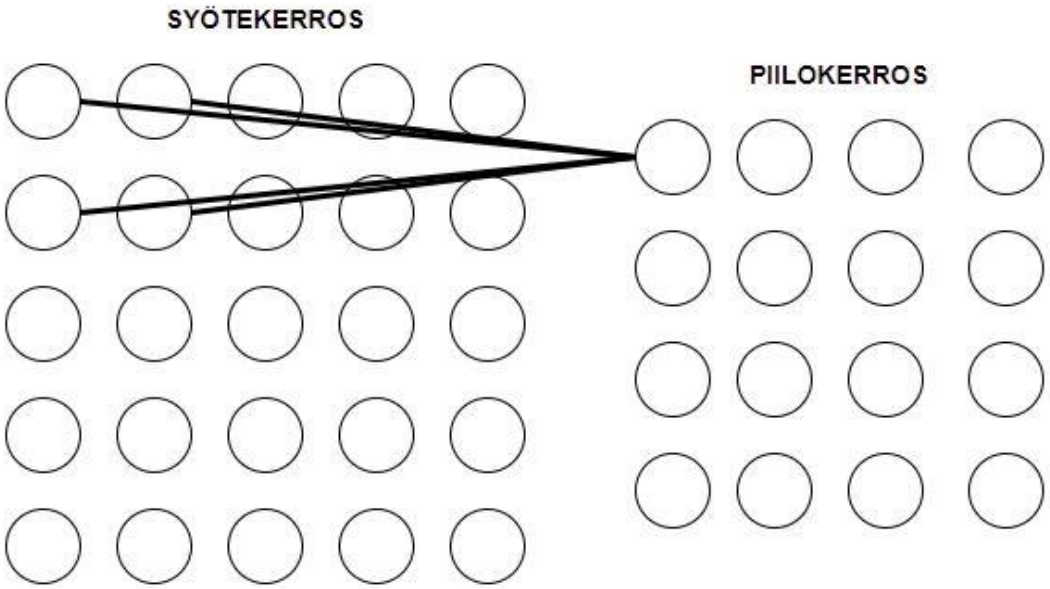
Kuvien käsittelyyn tarkoitetussa neuroverkossa syötekerroksen neuronien määrä määräytyy kuvan pikselien mukaan. Kuvan leveys ja korkeus muodostavat syötekerroksen kaksi ulottuvuutta ja pikselien väriarvot RGB-arvoilla ilmoitettuna kolmannen ulottuvuuden. Esimerkiksi 300x300 kuva tarkoittaa 270000 neuronia. Eteenpäin kytketyssä neuroverkossa tämä tarkoittaisi, että jokaista piilokerroksen neuronia kohti olisi vuorostaan

270 000 kytkentää. Neuroverkosta tulisi siis hyvin raskas. Konvoluutio-neuroverkko kiertää tämän ongelman käyttämällä konvoluutio-operaatiota kuvassa 4 esitetyllä tavalla. Konvoluutiokerroksessa syötedatasta etsitään piirteitä erillisten filtterien avulla. Filtteriä liikutetaan syötedatakerroksen yli, ja se tunnistaa tiettyjä piirteitä. Filtterin ja syötteen konvoluutio-operaatiosta saadaan yksi lukema, joka kertoo, onko haluttu piirre havaittu kohdealueessa.[14]



Kuva 4. *Konvoluutio-operaatio [14]*

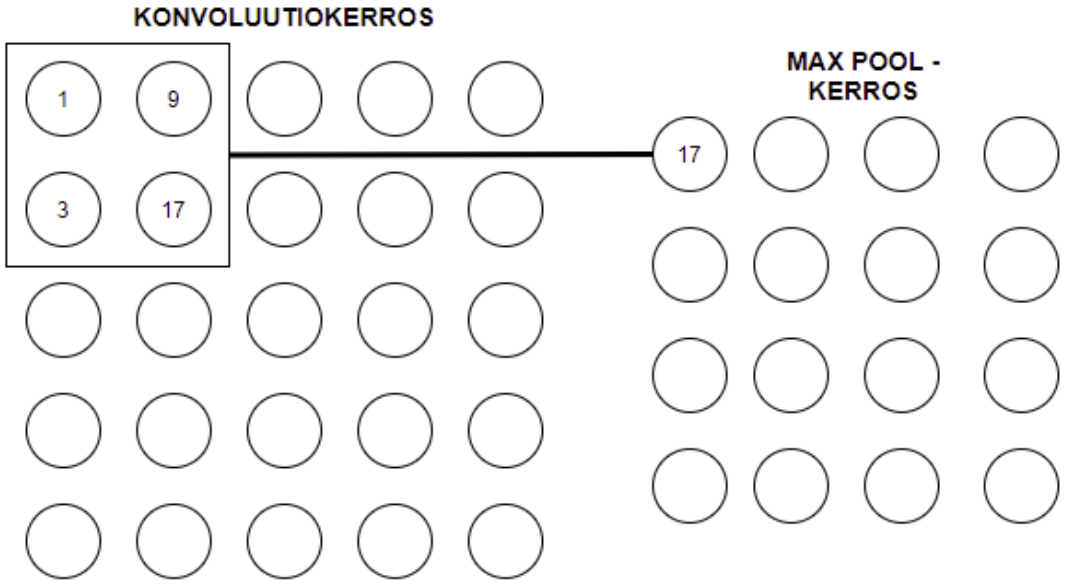
M. Nielsenin mukaan [13] konvoluutioneuroverkkojen toiminta perustuu kolmeen ideaan. Nämä ovat paikalliset vastaanottoalueet, yhteiset painoarvot ja näytteistyskerrokset.



Kuva 5. *Konvoluutioneuroverkon paikallisen vastaanottoalueen kuvaus. Mukaillen [13]*

Konvoluutioneuroverkon syötekerros voidaan kuvata kaksiulotteisena matriisina. Konvoluutiokerrosten neuronit ottavat syötteenään pienen alueen, jota kutsutaan paikalliseksi vastaanottoalueeksi. Tämä on esitetty kuvassa 5. Tällöin seuraavat kerrokset ovat huomattavasti pienempiä kuin syötekerros. Tämä keventää neuroverkon rakennetta ja vähentää sen suorituksessa vaadittavaa tietokoneen suorituskykyä. [13]

Konvoluutiokerroksessa kaikki neuronit käyttävät samoja painoarvoja, jolloin koko kerros tunnistaa vain yhtä tiettyä piirrettä. Tämä mahdollistaa kohteen tunnistamisen riippumatta siitä missä se sijaitsee käsiteltävässä syötteessä. Tästä on erityisesti hyötyä konenäkösovelluksissa, koska tunnistettava kohde voi olla kuvassa missä tahansa. Jokaista piirrettä kohti tarvitaan oma konvoluutiokerros. [13]



Kuva 6. *Max pool -yhdistyskerroksen toiminta. Mukaillen [13]*

Näytteistyskerroksissa (eng. pooling layers) etsitään tietyn kokoisesta alueesta esimerkiksi suurin arvo. Tällaista kerrosta kutsutaan max pool -kerrokseksi. Kuvassa 6 on esitetty miten max pool -kerros toimii. Esimerkissä 2x2 kokoisesta neuronimatriisista valitaan suurin arvo max pool -kerroksen neuronin arvoksi. Yhdistyskerroksia on myös muilla periaatteella toimivia, esimerkiksi min pool, joka poimii syötteistä pienimmän arvon. [13]

Konvoluutioneuroverkot ovat tehokkaita koneoppimisen välineitä. Konvoluutioneuroverkon etuja tavallisiin neuroverkkoihin ovat pienempi parametrien tarve, suurempi suoritusnopeus sekä hyvä tunnistuskyky. [13]

2.4 Neuroverkon opetus

Toimiakseen oikein halutulla datalla neuroverkko on ensin opetettava. Neuroverkon opetuksessa neuroverkko viritetään tunnistamaan haluttua dataa syötteestä. Opetuksessa neuronien kytkentöjen painoarvoja muutetaan siten että neuroverkon antama tulos vastaa haluttua tulosta. Opetusta varten tarvitaan suuri joukko dataa, jonka on oltava laadullisesti hyvää. Laatuun vaikuttavat datan vaihtelevuus ja selkeys. Seuraavassa on esitetty tarkemmin ohjatun oppimisen periaate, koska sitä käytettiin tutkimuksen käytännön työn osuudessa.

Ohjatussa oppimisessa neuroverkolle syötetään sekä analysoitava data että oikea vastaus mitä datasta halutaan tunnistaa. Neuroverkko tunnistaa datasta piirteitä ja vertaa niitä oikeaan vastaukseen. Näiden välistä eroa voidaan mitata funktiolla ja sitä kutsutaan

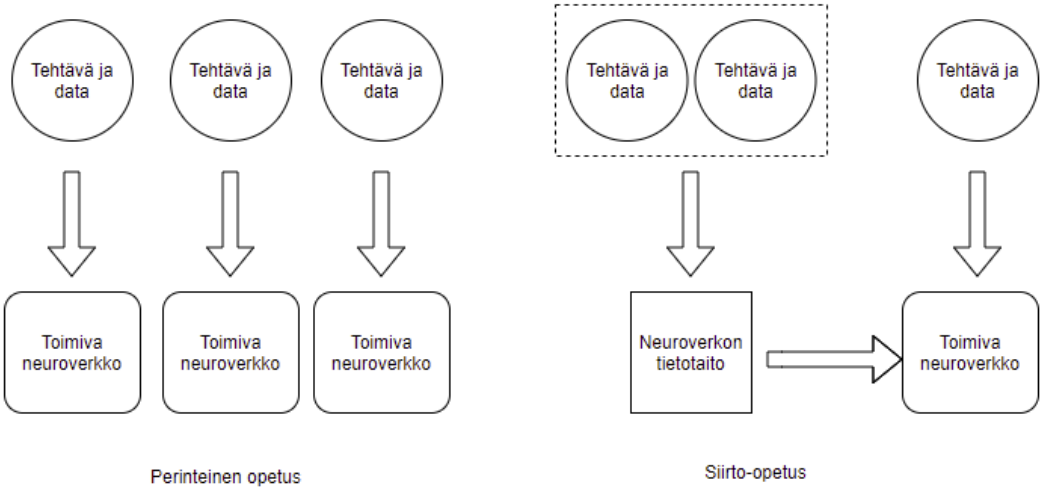
neuroverkon häviöfunktioksi (engl. loss function). Häviöfunktion tulos on lukuarvo, joka kertoo, miten hyvin neuroverkko suoriutui tunnistuksesta. Opetettaessa neuroverkkoa häviöfunktio pyritään minimoimaan, jolloin neuroverkon opetus voidaan ajatella optimointiongelmaksi. Häviöfunktioita on useita erilaisia, erilaisiin käyttötarkoituksiin sopivia. Yleisesti käytettyjä ovat keskimääräinen neliövirhe, ristientropiavirhe sekä logaritminen virhe. [14]

Häviöfunktion arvon pienentämiseksi neuroverkon opetuksessa käytetään funktion gradienttia. Gradientti kertoo mihin suuntaan funktion arvo kasvaa voimakkaimmin. Opetettaessa käytetään siis negatiivista gradienttia häviöfunktion arvon pienentämiseen. Neuroverkon painoarvoja muutetaan askel askeleelta, kunnes saavutetaan häviöfunktion minimi. Painoarvojen muuttaminen aloitetaan viimeisenä olevasta tuloskerroksesta. Painoarvojen muuttaminen etenee tästä kerroksittain kohti neuroverkon alkua vastavirta-algoritmin avulla. [14]

Neuroverkko pyritään opettamaan siten että se toimii hyvin sekä opetuksessa käytettävällä datalla, että myös opetusdatan ulkopuolisella datalla. Neuroverkko voi opetuksessa päätyä tilanteeseen, jossa se on ali- tai ylisovittunut opetusdataan. Alisovittunut neuroverkko ei tuota oikeaa tulosta riittävän hyvin, ja vaatii lisää opetusdataa. Ylisovittunut neuroverkko vuorostaan on optimoitunut opetusdatan tunnistamiseen, mutta ei toimi opetusdatan ulkopuolisella datalla. Osa opetusdatasta varataan testausta ja validointia varten sovitusongelmien välttämiseksi. Nämä datat ovat neuroverkon opetusdatan ulkopuolella, eikä neuroverkko siten ole ”nähnyt” niitä opetuksen aikana. Validointidatalla neuroverkon suoritusta testataan jokaisen opetuskerran jälkeen ja tehdään parametrien hienosäätöä. Testidatalla toiminta kokeillaan neuroverkon viimeisen opetuskerran jälkeen.[14]

Siirto-opetus

Neuroverkot ovat osoittautuneet hyvin toimiviksi monenlaisissa tehtävissä. Neuroverkon opetus vaatii kuitenkin paljon dataa ja aikaa. Jos neuroverkon data muuttuu, se on opetettava uudelleen. Uuden neuroverkon kohdalla tämä tarkoittaisi koko neuroverkon uudelleen rakentamista ja opettamista uudella datalla. Tämä ei ole välttämättä mahdollista tai kannattavaa. Ratkaisuksi tähän ongelmaan on kehitetty neuroverkkojen siirto-opetus (englanniksi transfer-learning). [16]



Kuva 7. *Perinteisen opetuksen ja siirto-opetuksen ero, mukaillen [16]*

Kuvassa 7 on esitetty perinteisen neuroverkon opetuksen ja siirto-opetuksen ero. Perinteisessä opetuksessa neuroverkko opetetaan toimimaan tietyssä tehtävässä siihen liittyvällä datalla. Näin saadaan neuroverkko, joka toimii vain opetuksessa käytetyllä datalla. Siirto-opetuksessa aikaisemmista tehtävistä ja datasta saatu tietotaito yhdistetään uuteen tehtävään ja dataan. Siirto-opetus nopeuttaa neuroverkkojen käyttöönottoa, koska koko opetusprosessia ei tarvitse suorittaa alusta alkaen.[16]

Siirto-opetusta käytettiin kaikissa tämän tutkimustyössä käytettyjen neuroverkkojen opetuksissa.

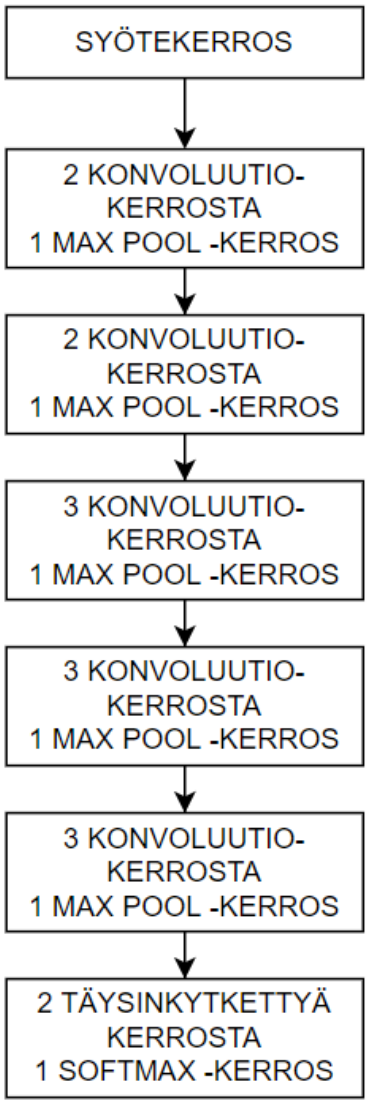
2.4.1 Erilaisia neuroverkkoja

Tutkimusta varten selvitettiin erilaisia neuroverkkoja, millä käyttökohteita voitaisiin tunnistaa. Käytetyt neuroverkot ovat esiteltynä alla, julkaisujärjestyksessä:

VGG16

VGG16-neuroverkko on konvoluutioneuroverkko, joka esiteltiin vuonna 2015. VGG16 tunnetaan myös alkuperäisellä nimellään ConvNet. VGG16 nimi tulee sen kehittäneen työryhmän nimestä, Visual Geometry Group, sekä neuroverkon mallissa olevasta 16 parametreja sisältävästä kerroksesta. Kokonaisuudessaan neuroverkko koostuu 22 kerroksesta. VGG neuroverkkoperheeseen kuuluu myös muita neuroverkkojen konfiguraatioita, joissa kerrosten lukumäärä vaihtelee. Suosituin näistä on tässä esitelty VGG16. [17]

VGG16 on julkaistu Creative Commons Attribution lisenssillä. Lisenssi mahdollistaa koodin jakamisen ja muokkaamisen sekä omaan että kaupalliseen tarkoitukseen. Alkuperäiselle tekijälle on annettava tunnustus sekä osoitettava tehdyt muutokset.



Kuva 8. *VGG16-neuroverkon rakenne. Mukaillen [18], sivu 4*

VGG16 rakenne koostuu pääosin konvoluutiokerroksista ja max pool -kerroksista. Neuroverkon rakenne on esitetty kuvassa 8. Kerrokset ovat ryhmitelty 2-3 konvoluutiokerroksen ja max pool-kerroksen ryhmiin. Konvoluutiokerroksissa syötteestä tehdään laskutoimitukset. Max pool -kerroksissa edellisten kerrosten tuloksista suodatetaan suurimmat arvot. Samalla neuroverkon parametrimäärä pienenee ja käsittely nopeutuu. VGG16 neuroverkon viimeiset kerrokset suorittavat syötteestä tunnistetun kohteen luokittelun. Ne rakentuvat kahdesta täysin kytketystä kerroksesta sekä softmax -kerroksesta [18].

VGG16 ja muilla VGG-neuroverkkoperheen jäsenillä voidaan suorittaa hyvin monenlaisia tehtäviä. Tutkimustyössä suoritettu artikkelikatsaus toi esille mm. erilaisia kuvan tunnistussovelluksia, henkilöiden tunnistussovelluksia sekä kirjoituksen ja merkkien tunnistussovelluksia. VGG16-neuroverkkoa käytetään myös joidenkin muiden neuroverkkomallien osana, kuten esimerkiksi alla esiteltävässä SSD-neuroverkossa.

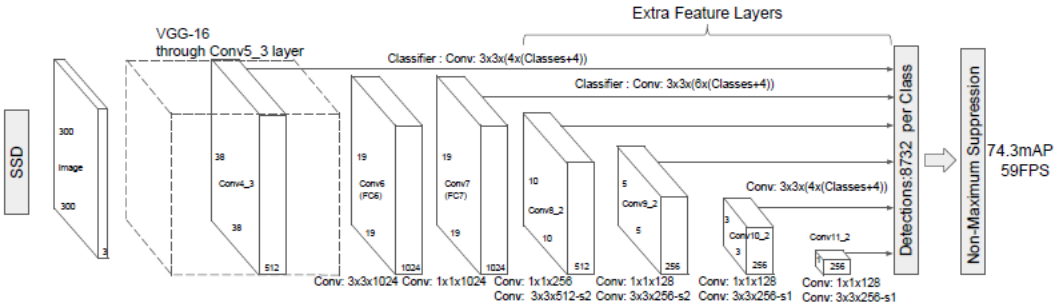
Iitti *et al.* tekemässä tapaustutkimuksessa [19] kehitettiin konenäkösovellus luokittelemaan tiiliä erilaisiin laatuluokkiin. Sovelluksessa käytetään VGG16 neuroverkkoa, joka siirto-opetetaan tunnistamaan tiiliä. Neuroverkon viimeinen kerros muutetaan vastaamaan tutkimuksessa käytettäviä luokkia. Neuroverkkoa testatessa se tunnisti kappaleet 88 % varmuudella. Tutkimuksessa todetaan, että neuroverkot ja tekoäly soveltuvat kyseisten tuotteiden laaduntarkastukseen ja jatkokehitys luokittelun toteuttamiseksi automaattilinjastoon kannattaa toteuttaa.

SSD

Single Shot Multibox Detector, SSD, on vuonna 2016 kehitetty neuroverkkomalli. Vapaasti suomentaen nimi tarkoittaa kertatunnistuksella toimivaa monikohdetunnistinta. SSD on eteenpäin syöttävä konvoluutioneuroverkko, joka tuottaa tietyn kokoisia tunnistuslaatikoita sekä todennäköisyyden kohteen luokalle. SSD kykenee tunnistamaan useita eri kohteita syötteestä. [20]

SSD jakaa tunnistettavan kuvan ruudukkoon, josta sitten tutkitaan ruutu kerrallaan. Ruudusta tunnistetaan mahdolliset muodot sekä todennäköisyydet kaikille tunnistettaville kohteille. Neuroverkko tunnistaa kohteelle useita mahdollisia sijainteja ruudukossa, ja näistä parhaan todennäköisyyden omaava valitaan lopulliseksi tunnistetuksi kohteeksi. [20]

Neuroverkkoa opetettaessa neuroverkolle kerrotaan myös kuvasta tunnistettavien kohteiden tiedot ns. totuuslaatikkoina. Totuuslaatikko kertoo kohteen tyypin, sekä sijainnin kuvassa. Sijainti ilmoitetaan neljän pisteen x- ja y-koordinaatteina. Kohde sijaitsee näiden pisteiden sisässä.[20]



Kuva 9. *SSD-neuroverkon rakenne. Lainattu lähteestä [20], sivulta 4*

SSD-neuroverkon malli on esitetty kuvassa 9. Neuroverkko ottaa syötteenään 300x300 pikselin kokoisen kuvan. Neuroverkon seuraavat kerrokset ovat osa VGG16-neuroverkosta, sen konvoluutiokerrokset. VGG16-neuroverkon täysin kytketyt kerrokset, jotka tuottavat luokittelun on jätetty pois. Tämän jälkeen SSD-neuroverkossa ovat luokitteluun käytettävät konvoluutiokerrokset. Lopussa ovat luokittelutuloksen määrittävät kerrokset.

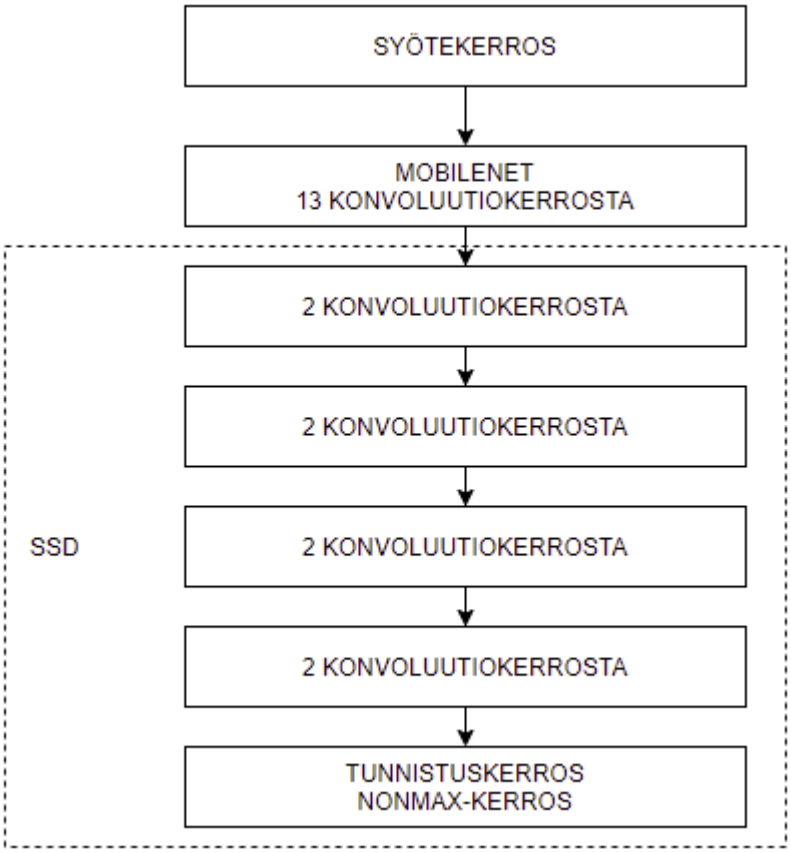
SSD-neuroverkkoa voidaan käyttää monenlaisissa käyttökohteissa. Artikkelikatsauksessa SSD-neuroverkkoa on käytetty esimerkiksi laaduntarkastussovelluksissa ja erilaisissa kuvan- ja hahmontunnistussovelluksissa.

Mobilenet-SSD

Mobilenet-SSD on eteenpäin syöttävä neuroverkkomalli, joka on suunniteltu kevyeksi ja nopeaksi kohteidentunnistusmalliksi. Mobilenet-SSD on jatkokehitelmä Mobilenet-neuroverkkomallista, jonka Google julkaisi 2016. Mobilenet-SSD neuroverkko rakentuu Mobilenet-neuroverkkoon yhdistetystä SSD-neuroverkosta. Mobilenet-osa suorittaa kohteen etsinnän, ja SSD-osa suorittaa tunnistuksen. [21]

Mobilenet-SSD on julkaistu MIT-lisenssillä, joten se on vapaasti käytettävissä, muokattavissa ja levitettävissä, kunhan lisenssi säilyy ohjelman lähdekoodissa.

Mobilenet-SSD:n rakenne on esitetty kuvassa 10. Syötekerrokselle syötetään 300x300 pikselin kuva. Alussa Mobilenet-neuroverkon 13 konvoluutiokerrosta suorittavat piirteiden etsinnän syötteestä. Tämän jälkeen SSD-osan 8 konvoluutiokerrosta suorittavat kohteiden tunnistuksen. [21]



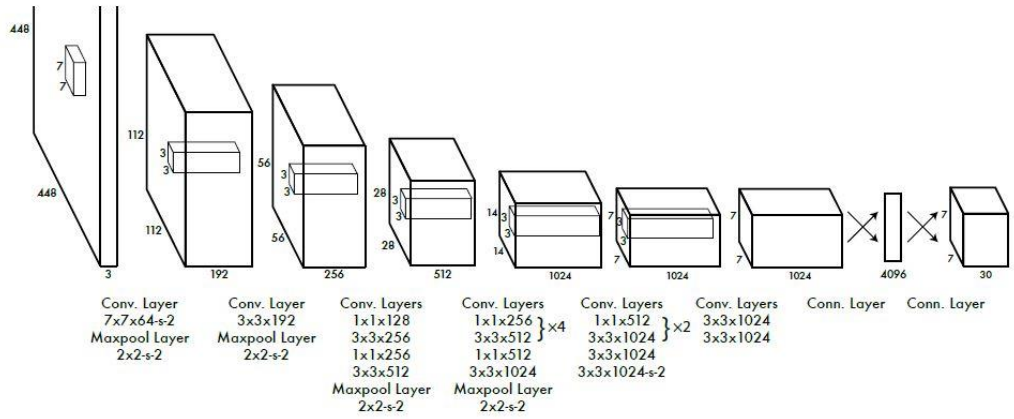
Kuva 10. *Mobilenet-SSD rakenne, mukaillen [21], sivu 9*

Y. Li et al [21] ovat saaneet Mobilenet-SSD-neuroverkolla hyviä tuloksia tuotteiden laadunvalvontasovelluksessa. Heidän tutkimuksessaan kehitettiin sovellus, jota tunnistaa pakkauksen suljinpinnan mahdolliset valmistusviat. Tutkimuksen mukaan Mobilenet-SSD saavutti tunnistuksessa 95 % tunnistusvarmuuden ja suoriutui käytetyllä laitteistolla tunnistuksesta 120 millisekunnissa. Tutkimuksessa suoritetussa kolmen eri neuroverkon vertailussa Mobilenet-SSD saavutti parhaimman tunnistusvarmuuden. Muut tutkitut neuroverkot olivat VGG16 ja Mobilenet. VGG16 oli vain hieman heikompi tunnistusvarmuudessa, mutta lähes 10 kertaa hitaampi. Mobilenet vuorostaan on hieman nopeampi, mutta huonompi tunnistusvarmuus kuin Mobilenet-SSD:ssä. Merkittävin etu Mobilenet-SSD-neuroverkon käytöstä on sen keveys yhdistettynä hyvään tunnistusvarmuuteen.

YOLO

YOLO on kohteentunnistusalgoritmi, joka on lyhenne sanoista You Only Look Once. Vapaasti suomennettuna se tarkoittaa ”Katsot vain kerran”. YOLO julkistettiin 2016. YOLO on erityisesti kehitetty kohteiden tunnistamiseen kuvista ja se kykenee tunnistamaan kuvasta useita eri kohteita. Syötetty kuva käsitellään neuroverkossa vain kerran, jolloin saavutetaan pienempi viive tuloksen saamiseen. YOLOn kehittäneiden Redmon *et al* [22] mukaan YOLO oli julkaisuhetkellä huomattavasti nopeampi kuin muut kuvantunnistusmallit. Heidän käyttämällään laitteistolla päästiin tunnistusnopeuteen 45 ruutua/sekunnissa, ja kevyemmällä Fast YOLO neuroverkolla tunnistusnopeuteen 155 ruutua/sekunti.

YOLO on avoimen lähdekoodin neuroverkkomalli, joka on julkaistu vapaaseen käyttöön GNU GPL-lisenssillä [23]. Ohjelman lähdekoodi ja lisenssit ovat saatavilla GitHub-verkkosivustolta. [24] Lisenssin mukaan ohjelmaa saa vapaasti käyttää, kehittää ja levittää.



Kuva 11. *YOLO neuroverkon rakenne. Lainattu lähteestä [22]*

YOLO neuroverkko on rakenteeltaan yksi putki. Kuvassa 11 on esitetty neuroverkon rakenne. Neuroverkko koostuu 24 konvoluutiokerroksesta ja 2 yhdistyskerroksesta. Konvoluutiokerrosten välissä on maxpool-kerros, jolla kerätään edellisen kerroksen matriiseista suurimmat arvot. Neuroverkon lopussa oleva täysin kytketyt kerrokset esittävät tuloksen.

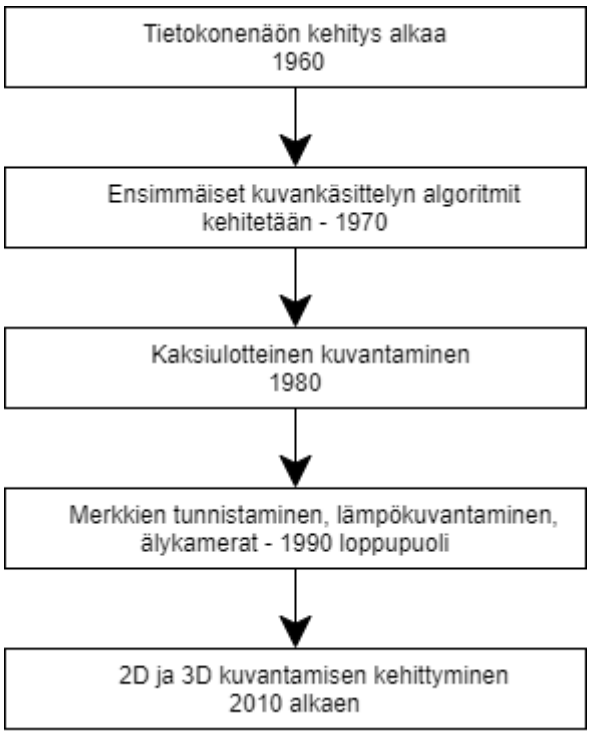
YOLOn toimintaperiaate on, että kuvasta tunnistetaan kaikki kohteet yhdellä kertaa. YOLOn tunnistusalgoritmissa kuva ensin pienennetään sopivaan kokoon. Tämän jälkeen kuva jaetaan $S \times S$ kokoiseen ruudukkoon. Kuvasta etsitään piirteitä, joiden perusteella tunnistetaan kohteita. Se ruutu, johon tunnistetun kappaleen keskipiste osuu, on vastuussa kohteen tunnistuksesta. Jokaisella ruudulla voi olla B kappaletta tunnistettuja kohteita. Tunnistetuilla laatikoilla on 5 arvoa. x, y, w, h ja tunnistusvarmuus. Koordinaatit (x, y) kertovat tunnistuslaatikon keskipisteen ruudussa. Leveys w ja korkeus h ovat tunnistuslaatikon mitat suhteutettuna kuvan kokoon. Tunnistusvarmuus on arvo sille, kuinka varmasti kohde on tunnistettu.

Redmon et al [22] mukaan YOLO-neuroverkon suurin vahvuus on sen nopeus. Nopeutta tarvitaan heidän mukaansa esimerkiksi itseohjautuvien ajoneuvojen kameralaitteistoissa tunnistamaan liikenteessä olevia kohteita. YOLOn tunnistusvarmuus on kuitenkin heikompi ja sillä on ongelmia löytää ja tunnistaa pienet kohteet kuvista.

2.5 Konenäkö

Tietokonenäkö määritelmänä kattaa Anand ja Priya [25] mukaan perusteknologian kuvan analysointiin missä tahansa sovelluksessa. Konenäkönä ymmärretään tietokonenäön liittämisen teollisuusautomaatioon. Konenäön avulla teollisuuden prosessissa voidaan mm. suorittaa prosessin ohjausta ja laadunvalvontaa. Teollisuuden konenäkösovellukset ovat yleensä kaksiulotteisia kuvia käyttäviä, koska kameralaitteisto on edullisempi ja kuvia on helpompi käsitellä kuin 3-ulotteisia kuvia käytettäessä. 3D-kamerajärjestelmien hankintahinta on kuitenkin laskenut, ja niiden käyttö lisääntyy.

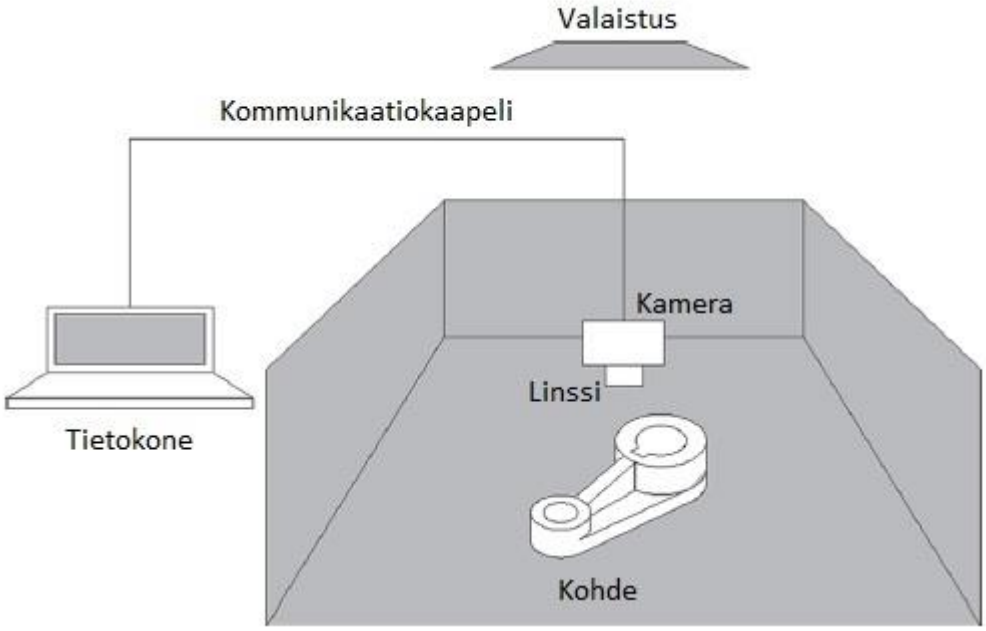
Konenäön tutkimisen ja kehittämisen tarkoituksena on tuottaa tietokoneille ihmisen kaltainen havainnointikyky. Tällä kyvyllä tietokoneet voisivat aistia ympäristöä, ymmärtää siitä saatavaa dataa ja tehdä tarvittavia toimenpiteitä. Lisäksi koneet oppisivat tästä kaikesta ja kehittäisivät suorituskykyään. [26]



Kuva 12. *Tietokonenäön kehitys (mukaillen [25])*

Tietokonenäön ja sen osa-alueen konenäön kehittämisen voidaan katsoa alkaneen vuonna 1960. Silloin esitettiin ensimmäisen kerran käsite tietokonenäkö. Tietokonenäköä ajateltiin tarvittavan robottien älykkään käyttäytymisen mahdollistamiseksi. Kuvassa 12 esitetään tietokonenäön kehityksen vaiheet. Ensimmäiset algoritmit kuvan käsittelyä varten kehitettiin 1970 luvulla. Nämä algoritmit ovat toimineet pohjana hyvin mo-

nille nykyajan tietokonenäön algoritmeille. 1980-luvulla kehitystä tapahtui 2D-kuvantamisessa ja merkkien tunnistamisessa. Älykamerat keksittiin 1980-luvun loppupuolella. Nykyään kehitystä tapahtuu paljon 3D-kuvantamisen osa-alueella. [25]



Kuva 13. *Konenäköjärjestelmän rakenne (Mukaillen [25])*

Konenäköjärjestelmän pääkomponentit ovat kamera, linssi, valonlähde, kuvankäsittelyohjelmisto ja kommunikaatiorajapinnat. Komponentit valitaan sovelluksen mukaan Kuvassa 13 on esitetty laitteiston pääkomponentit. Kuvattava alue ja kohde valaistaan. Kohteesta heijastuva valo kerätään linssillä kameran kennolle. Digitaalinen kuva siirretään tietokoneelle käsittelyä ja tulkintaa varten. Kuvan tulkinnan perusteella voidaan ohjausjärjestelmällä tehdä tarvittavat toimenpiteet prosessin seuraavissa vaiheissa.[25]

Teollisuudessa konenäkösovelluksia on hyvin monenlaisia. Yleisimpiä ovat laaduntarkastukseen liittyvät sovellukset. Konenäköjärjestelmiä valmistavan Cognex-yhtiön kotisivuilla [27] on useita erilaisia esiteltynä. Kokoonpanoon liittyvät sovellukset ovat myös hyvin yleisiä, näissä konenäöllä voidaan esimerkiksi paikantaa käytettävien komponenttien sijainteja ja tarkistaa niiden asemointia toisiinsa nähden.

Koneoppiminen mahdollistaa konenäön automaattisen mukautumisen sovelluksen mukaan. Konenäkösovellus voi muuttaa parametrejaan ja algoritmejaan siten että järjestelmä oppii tunnistamaan mitä kuvista halutaan löytää. [26]

N. Sebe [26] *et al.* toteavat, että oppivilta konenäköjärjestelmiltä odotetaan parempaa suorituskykyä ja parempaa yleistä sopivuutta. Oppiva järjestelmä voidaan heidän mukaansa kehittää toimimaan yhdessä sovellusalueessa, ja siitä saatavalla kokemuksella järjestelmä voidaan siirtää toimimaan toisessa sovellusalueessa.

Vaativat konenäön sovellukset käyttäen tekoälyä sekä kehittyneitä tekniikoita yleistyvät. Esimerkiksi A. Tellaeche [28] et al. tekemässä tapaustutkimuksessa yhdistettiin stereokonenäköjärjestelmä yhteistyörobottiin. Stereokonenäköjärjestelmän tuottaman tiedon perusteella robotti osaa asentaa komponentit oikein, vaikka kohteen sijainti vaihtelisikin.

B. Staar [9] et al. tutkimuksessa konvoluutioneuroverkolla tunnistetaan kuvasta kappaleen pinnassa olevia virheitä. Neuroverkkoa ei opeteta tunnistamaan tiettyä virhettä, vaan tulkitsemaan kuvasta normaalista pinnasta poikkeavaa muotoa. Tällöin neuroverkko toimisi myös uusilla tuotteilla automaattisesti. Tämän kaltaisille yleisille tunnistimille voidaan ajatella olevan tarvetta hyvin monessa laaduntarkastussovelluksessa, missä tuote itsessään pysyy ulkomuodoiltaan vakiona.

2.6 Avoin lähdekoodi

Tutkittaessa avoimen lähdekoodin sovelluksia on myös tarpeen tarkastella, mitä avoin lähdekoodi tarkoittaa. Avoimella lähdekoodilla ymmärretään ohjelmistoja, joiden lähdekoodi on vapaasti saatavilla ja jotka ovat vapaasti käytettävissä, muokattavissa ja levitettävissä. Avoimen lähdekoodin kehitystyötä voi tehdä kuka tahansa. [29]

Avoimen lähdekoodin ohjelmointia tehdään yleensä hajautetusti ja yhteistyönä. Koodia tarkastetaan muiden tekijöiden toimesta. Avoimen lähdekoodin ohjelmat ovat yleensä halvempia, joustavampia ja pidempi-ikäisiä kuin kaupalliset verrokkinsa koska kehitystyöhön osallistuu yhteisö eikä yksittäinen toimija tai yritys. [29]

Tyypillisesti kaupalliset ohjelmat myydään valmiiksi tehtyinä ohjelmistoina, jotka asennetaan käyttäjien tietokoneille. Ohjelman kehittäjä huolehtii ohjelman päivittämisestä ja ylläpidosta. Ohjelmalla on kuitenkin usein rajallinen elinkaari, jonka jälkeen kehittäjä ei enää päivitä tai tue ohjelman käyttöä. Eri käyttöjärjestelmissä ohjelman käyttö vaatii yleensä oman lisenssin kullekin järjestelmälle. Myös käyttäjien lukumäärää voi olla rajoitettu lisenssissä. Avoimen lähdekoodin käyttäjä voi asentaa ohjelman vapaasti, ja saada ohjelmaan tukea muilta käyttäjiltä. Pääsy ohjelman lähdekoodiin mahdollistaa myös ongelmien ratkaisun itse muokkaamalla ohjelmaa. [30]

Avoimeen lähdekoodiin liittyy oikeudellinen lisenssi, jolla ohjelman tekijä antaa ohjelmansa muiden käyttäjien vapaaseen muokkaukseen ja käyttöön. Tekijänoikeus säilyy kuitenkin ohjelman alkuperäisellä tekijällä.

Avoimen lähdekoodin käyttöä ja tietoisuutta edistävä Open Source Initiative -järjestö on määrittänyt avoimen lähdekoodin piirteet seuraavasti: [31]

1. Ohjelman myyntiä tai levitystä ei saa rajoittaa. Jos ohjelma on komponenttina toisessa ohjelmassa, ei sen käytöstä tai levityksestä saa periä maksua.
2. Ohjelman pitää sisältää lähdekoodi tai se pitää olla saatavilla tunnetulla tavalla. Lähdekoodin pitää olla muodossa missä siihen voidaan tehdä muutoksia.
3. Ohjelmaan on sallittua tehdä muutoksia tai jatkokehitystä, ja näin muokattua ohjelmaa on sallittua levittää samalla lisenssillä.
4. Muokattua lähdekoodin levitystä voidaan rajoittaa vain, jos lisenssi sallii erillisten korjaustiedostojen ja niiden lähdekoodin levittämisen. Korjaustiedostoilta voidaan vaatia erilaista nimeä tai versionumerointia kuin alkuperäisessä ohjelmassa.
5. Ohjelman lisenssi ei saa asettaa ihmisiä tai ryhmiä eriarvoiseen asemaan.
6. Ohjelmaa on oikeus käyttää kaikenlaisiin käyttötarkoituksiin.
7. Kaikilla ohjelman käyttäjillä on samat oikeudet.
8. Ohjelman oikeudet eivät saa riippua isommasta ohjelmistokokonaisuudesta, jos ohjelma on sen osana. Ohjelman oikeudet säilyvät vaikka ohjelma irrotettaisiin tästä kokonaisuudesta.
9. Lisenssi ei voi asettaa ehtoja muille ohjelmille mitä käytetään lisenssin alaisen ohjelman kanssa. Esimerkiksi ei voida vaatia, että muut ohjelmat olisivat avointa lähdekoodia.
10. Lisenssi pitää olla riippumaton teknologiasta.

Seuraavana on esiteltynä työssä käytettyjen avoimen lähdekoodin neuroverkkojen lisenssit.

GNU General Public License (myöhemmin GNU GPL) on vapaiden ohjelmien lisenssi, joka on julkaistu alun perin GNU-projektia varten. Lisenssin kolmas versio on julkaistu 2007. Lisenssissä annetaan oikeudet käyttää ohjelmaa vapaasti kaikkiin tarkoituksiin, muokata ohjelmaa sopimaan käyttäjän tarpeisiin, sekä oikeus jakaa ohjelmaa ja siihen tehtyjä muutoksia vapaasti. GNU GPL on hyvin yleisesti käytetty avoimen lähdekoodin lisenssi [32]. GNU GPL sallii ohjelman käyttämisen myös osana rajatun lisenssin ohjelmistoa. [33]

MIT-lisenssi on avoimen lähdekoodien ohjelmien lisenssi, jota kutsutaan myös X11-lisenssiksi. Lisenssin on rinnastettavissa GNU GPL -lisenssiin. [33]

Creative Commons -lisenssi mahdollistaa tekijänoikeuksilla suojatun materiaalin vapaan levittämisen. Lisenssin neljäs versio on julkaistu 2013. Lisenssi säilyttää tekijänoikeudet alkuperäisellä tekijällään, mutta sallii materiaalin levittämisen, kunhan alkuperäiselle tekijälle annetaan tunnustus tehdystä työstä. [34]

3. SOVELLUS

Työn sovellusosuudessa tutkittiin erilaisia avoimen lähdekoodin konenäön neuroverkko-ohjelmia sovelluksissa, jotka määritettiin yhdessä ABB SP:n kanssa. Tutkimus suoritettiin käyttämällä menetelminä havainnointia, dokumentointia ja haastatteluja. Nykytila-analyysi on esitetty luvussa 3.1. Tutkimuksen koejärjestelyt ovat esitettynä luvussa 3.2.

3.1 Nykytila-analyysi

Sovelluksen toteuttaminen aloitettiin suorittamalla ABB SP:n tuotannossa nykytila-analyysi. Tätä varten suoritettiin kierros ABB SP:n tuotannossa yhdessä tuotannon kehityshenkilöiden kanssa. Tuotantotiimien toimintaa käytiin läpi kartoittaen tuotannon vaiheita, joissa konenäkö voisi toimia ihmisen apuna. Tämän selvityksen perusteella saatiin arvio tuotannon nykyisestä tilasta ja siitä mitä käyttökohteita voidaan löytää. Myös tuotantotiimien työntekijöiltä kyseltiin heidän näkemyksiään ja ajatuksiaan käyttökohteista konenäkölaitteistolle kyselylomakkeilla ja haastatteluilla. Kyselylomakkeet ovat esitetty liitteissä A ja B.

Nykytila-analyysin perusteella ABB SP:n tuotanto on hyvin organisoitua, tehokasta sekä automatisoitua. Automatisointiprojektien pohjalta myös tuotannon kehitystiimillä on hyvä kokemus ja näkemys siitä, miten ja mihin suuntaan tuotantoa halutaan kehittää. Tämän tutkimustyön aihe on yhden kehitysidean tutkiminen käytännön testeillä.

3.1.1 Käyttökohteet

Tuotannosta löydettiin useita käyttökohteita, joissa kameralaitteistoa voidaan käyttää. Käyttökohteiden valinta suoritettiin vertailemalla niiden vaikuttavuutta tuotantoon sekä toteutuskelpoisuutta.

Löydetyt käyttökohteet olivat seuraavanlaisia:

- Kokoonpanossa tapahtuvien laatuvirheiden tunnistaminen
- Kokoonpanon vaiheen onnistumisen tarkistus
- Robotin poimiman kappaleen asennon tarkistus
- Robotin suorittaman poiminnan tarkistus
- Tuotteiden merkintöjen tarkistaminen
- Huonosti toimivan konenäkökameran korvaaminen
- Automaattisessa kokoonpanossa käytettävän anturin korvaaminen
- Tuotepakkauksen tarkistus keräilyn jälkeen
- Tuotteiden keräilyalustojen tarkastus väärien tuotteiden varalta
- Yleinen siisteyden tarkistus työpisteessä
- Työkalujen ja tarvikkeiden seuranta
- Automaattivaraston tuotepaikkojen tarkistus väärien tuotteiden varalta

Kohteet voidaan jakaa kahteen kategoriaan:

1. Prosessin tai tuotteen laadun valvonta
2. Tuotantoprosessin edistämiseen liittyvät kohteet.

Pääasiassa kohteet olivat erilaisia kokoonpanon laadunvalvontakohteita. Esimerkiksi tarkistukset, onko jokin komponentti mennyt kokoonpanossa oikealle paikalleen. Myös ulkoisia laaduntarkastuksia kappaleen ulkonäölle ja merkinnöille oli useita. Prosessia edistäviin kohteisiin kuuluvat vuorostaan erilaiset robotin tekemien työvaiheiden varmistaminen tai ohjausprosessiin kuuluvan anturin korvaaminen. Joissakin kokoonpanoissa anturilla tehtävä tarkistus ei ole riittävän tarkka, vaan virheellisiä tunnistuksia esiintyy. Konenäöllä voidaan tehdä sama tarkistus.

Käyttökohteiden vaikutusta tuotantoon ja toteutuskelpoisuutta arvioitiin määritettäessä mitkä niistä valitaan tarkempaan tarkasteluun ja Proof-of-Concept-testaukseen. Käyttökohteen vaikutusta arvioitiin seuraavien ominaisuuksien perusteella: kuinka hyvin konenäkö helpottaa tuotannon työntekijöiden työtä tässä kohteessa sekä kuinka moneen tuotantolinjaan tätä voidaan hyödyntää. Toteutettavuutta arvioitaessa tarkasteltiin, kuinka helposti laitteisto voidaan sijoittaa tuotantolinjastolle ja miten vaikeaksi sovelluksen tekeminen arvioitiin. Vertailutaulukko on esitetty alla olevassa taulukossa 1.

Taulukko 1. *Käyttökohteiden vertailu*

Kohde	**Vaikuttavuus tuotantoon**	**Toteutettavuus**
Kokoonpanon laatuvirheiden tarkastus	Hyvä	Hyvä
Kokoonpanon vaiheen onnistumisen tarkastus	Kohtalainen	Hyvä
Robotin poimiman kappaleen asennon tarkistus	Vähäinen	Heikko
Robotin suorittaman poiminnan tarkistus	Kohtalainen	Heikko
Tuotteiden merkintöjen tarkastaminen	Kohtalainen	Hyvä
Huonosti toimivan konenäkökameran korvaaminen	Hyvä	Heikko
Automaattisessa kokoonpanossa käytettävän anturin korvaaminen	Kohtalainen	Heikko
Tuotepakkauksen tarkistus keräilyn jälkeen	Hyvä	Hyvä
Tuotteiden keräilyalustojen tarkastus väärien tuotteiden varalta	Kohtalainen	Hyvä
Yleinen siisteyden tarkistus työpisteessä	Heikko	Hyvä
Työkalujen ja tarvikkeiden seuranta	Kohtalainen	Hyvä
Automaattivaraston tuotepaikkojen tarkistus väärien tuotteiden varalta	Kohtalainen	Hyvä

Näistä paras toteutettavuus ja vaikutus arvioitiin olevan seuraavissa kohteissa:

1. Tuotteen loppupakkauksen tarkastus
2. Tuotteen kokoonpanon jälkeinen laaduntarkastus

Loppupakkauksen tarkastukselle on tarvetta ABB:n tuotannossa, koska sille suoritetaan ainoastaan ihmisen tekemä visuaalinen tarkastus pakkausta tehtäessä. Automaattisella pakkauksen sisällön tarkastuksella voidaan vähentää väärien osien ja ohjeiden päätymistä pakkaukseen ja siten vähentää palautuvien tuotteiden määrää. Tämä tuotannon vaihe koskettaa jokaista tehtaasta valmistuvaa tuotetta ja siksi sillä arvioitiin olevan hyvin suuri vaikutus. Myös toteutuskelpoisuus on hyvä, koska tarkastus suoritetaan tuotannon työntekijöiden toimesta normaalilla työpisteellä. Työpisteellä on hyvin tilaa kameralle ja tietokoneelle.

Tarve tuotteen kokoonpanon laaduntarkistukselle mahdollisten vikojen varalta oli suuri eräässä ABB SP:n kokoonpanolinjassa. Linjastolla on ollut aikaisemmin lopputarkastuskamera, joka on kuitenkin poistettu vikaantumisen takia käytöstä. Nyt tuotteen tarkistus on käyttäjien vastuulla. Tuotteita valmistuu päivässä suuri määrä, joten tarkistustyötä on paljon ja siksi laitteiston vaikutus on suuri. Laaduntarkistus suoritetaan automaattisessa kokoonpanosolussa tehtävän osuuden jälkeen, jolloin laitteisto voidaan sijoittaa työntekijän käyttämälle työpisteelle. Myös muilla kokoonpanolinjoilla on tarve tarkistukselle, mutta ne jätettiin työn ulkopuolelle. Jos testaus osoittautuisi toimivaksi, ideaa voitaisiin soveltaa myös näihin linjoihin.

3.1.2 Testilaitteisto

Testilaitteiston oli tarkoitus olla kuluttajamarkkinoilta saatavilla oleva kameralaitteisto, tietokone sekä tarvittava valaistus ja kaapelointi. Markkinoilta on saatavilla hyvin monenlaisia tietokoneita, jotka ovat käyttökelpoisia alustoja tälle laitteistolle. Tekoälysovellusten vaatimaa laskentatehoa on tutkimusten mukaan parhaiten käytettävissä grafiikkasuorittimissa. Grafiikkasuorittimet taas vaativat enemmän tehoa sekä jäähdytyskykyä koko tietokoneen laitteistolta, johtaen isompiin kustannuksiin. Tutkimuksessa haluttiin samalla tutkia pienten ja edullisten tietokoneiden soveltuvuutta konenäkökäyttöön, koska edulliselle järjestelmälle on helpompi perustella toissijaisiakin käyttökohteita kuin kalliimmalle. Jos järjestelmän hankintahinta ei ole este, voidaan konenäköä hyödyntää kohteissa, joissa sitä ei normaalisti ole edes harkittu.

Valintaa varten tarkasteltiin muutamaa vaihtoehtoa. Nämä ovat esiteltynä seuraavaksi.

Raspberry PI 4

Raspberry Pi on pienoistietokone, jonka tarvitsema elektroniikka on mahdutettu yhdelle piirilevylle. Raspberry Pi on Raspberry Pi Foundation -säätiön ja sen omistaman Raspberry Pi Ltd -yhtiön kehittämä. Raspberry Pi-tietokoneet ovat tarkoitettu halvaksi alustaksi erilaisissa opetuskäytöissä tai projekteissa tietokoneista, elektroniikasta ja ohjelmoinnista kiinnostuneille käyttäjille. [35]

Raspberry Pi 4 Model B vaikutti ominaisuuksien ja hinnan puolesta mielenkiintoiselta vaihtoehdolta, mutta sen suorituskykyä ei pidetty riittävänä laskentatehoa vaativaan tekoälysovelluksen reaaliaikaiseen suorittamiseen. Markkinoilta on saatavilla erillisiä tekoälykiihdyttimiä, joilla saadaan Raspberry Pi:n laskentatehoa nostettua tekoälysovelluksia varten. Näiden saatavuus osoittautui kuitenkin diplomityön teon aikaan hyvin heikoksi, joten niiden käyttäminen työhön jätettiin pois. Erillisen tekoälykiihdyttimen käyttäminen olisi ollut tutkimisen arvoinen vaihtoehto.

Raspberry Pi-tietokoneet ovat hyvin suosittuja erilaisien harrastelijaprojektien toteutuksiin. Erilaisia projekteja on jaossa internetissä hyvin suuri määrä. Dokumentaatiot ja käyttöohjeet ovat pääsääntöisesti kattavat.

Jetson Nano Developer Kit

Toisena vaihtoehtona tutkittiin Nvidia Jetson Nano Developer Kit -pienoistietokoneen käyttöä. Jetson Nano on tekoälyn tutkimus-, oppimis- ja kehityskäyttöön kehitetty pienoistietokone. Jetson Nano on suhteellisen uusi tähän käyttöön tarkoitettu tietokone. Developer Kit on varsinaisen Jetson Nano -kortin lisäosakortti. Tässä kortissa on liittimet virran syötölle, USB- ja verkkoliittimet sekä näytön liittimet. Jetson Nanossa on erillinen grafiikkasuoritin, mikä antaa sille tehoa vaativaan laskentaan. Jetson Nano Developer Kit on kuitenkin suhteellisen pienellä tehontarpeella, maksimitehoksi ilmoitetaan 10 wattia.[36]

Jetson Nanon tekoälysovelluksiin tarkoitetut ominaisuudet olivat hyvin mielenkiintoinen mahdollisuus kokeilla laitteistoa. Jetson Nanolle on Nvidian toimesta tehty kattavat valmiit ohjelmistot ja käyttöohjeet. Lisäksi internetistä löytyy monia valmiita projektiesimerkkejä erilaisiin tekoälysovelluksiin.

Laitteistojen vertailu

A. Süzen *et al* [37] ovat tehneet vertailun Jetson Nanon, Raspberry Pin ja Jetson TX2 välillä syvien neuroverkkojen käytössä. Jetson TX2 on Nvidian valmistama yhden piirilevyn tietokone, joka kuuluu samaan Jetson-tuoteperheeseen. TX2 on tehokkaampi ja kal-

liimpi kuin Nano, ja ei ole hintansa vuoksi mukana tässä vertailussa. Heidän tutkimuksessaan todetaan, että Jetson Nano on huomattavasti nopeampi kuin Raspberry PI 4. Konvoluutioneuroverkolla suoritetussa testissä tunnistusvarmuudet pysyivät suurinpiirtein samana, mutta Jetson Nano oli parhaimmillaan kuusi kertaa nopeampi. Jetson TX2 oli samoissa testeissä noin kaksi kertaa nopeampi kuin Jetson Nano. Jetson Nanon hinta on noin kaksi kertaa enemmän kuin Raspberry Pi 4:n. Tutkimuksen mukaan Nvidian Jetson-tuoteperheen laitteet ovat parempi vaihtoehto tekoälysovelluksiin.

Laitteistojen tärkeimpien ominaisuudet ovat kirjattu alla olevaan taulukkoon 2. Raspberry Pi 4B:n prosessori ja grafiikkasuoritin ovat integroitu yhdeksi siruksi, kun taas Jetson Nanossa on erillinen täysikokoinen grafiikkasuoritin. Tämä takaa Jetson Nanolle paremman suorituskyvyn. Raspberry Pi 4B:n prosessori on tehokkaampi kuin Jetson Nanossa.

Taulukko 2. *Tutkittujen tietokoneiden ominaisuuksien vertailu. [35][36][37]*

Ominaisuudet	**Raspberry Pi 4B**	**Nvidia Jetson Nano**
Prosessori	4-ytiminen ARM A72, 1,5 GHz	4-ytiminen ARM A57, 1,43 Ghz
Grafiikkasuoritin	Broadcom VideoCore VI, 500 MHz	128 ytiminen Maxwell, 951 MHz
Keskusmuisti	1/2/4/8 Gb LPDDR4	2/4 Gb LPDDR4
Massamuisti	Micro-SD -muistikortti	Micro-SD -muistikortti

Laitteisto päätettiin toteuttaa Jetson Nanolla sen parempien ominaisuuksien vuoksi. Jetson Nanon suurin etu Raspberry Pi:hin nähden on sen tehokkaampi grafiikkasuoritin. Grafiikkasuoritin soveltuu hyvin tekoälyn vaatimaan laskentaan. Varsinkin grafiikkasuorittimen tuoma laskentateho on tarpeen reaaliaikaista tunnistusta tavoiteltaessa. Jetson Nano on arviolta kaksi kertaa kalliimpi kuin Raspberry Pi 4, mutta huomattavasti tehokkaampi. Hankintahinnaltaan Jetson Nano on kuitenkin edelleen suhteellisen edullinen tietokoneeksi, joka kykenee suorittamaan neuroverkko-ohjelmistoja lähes reaaliajassa.

Laitteiston kameraksi valittiin Logitechin valmistama C920S web-kamera. Kamera on ominaisuuksiltaan laadukas nykyaikainen web-kamera. Kameran resoluutio on 1920x1080 pikseliä, ja se kykenee 30 kuvaan sekunnissa. Kamerassa on automaattinen tarkennus. Kamera kytketään USB-kaapelilla testilaitteistoon.

Käyttöjärjestelmä asennetaan Micro-SD-muistikortille. Kortin vaatimuksena on vähintään 32 gigatavun kapasiteetti sekä UHS-1 luokan kirjoitusnopeus. Muistikortin käyttö massamuistina on hyvin tyypillistä tällaisissa yhden piirilevyn tietokoneissa.

Testilaitteiston kokonaiskustannukset asettuivat 350 euron paikkeille, sisältäen tietokoneen, muistikortin, kameran ja virtalähteen. Koska käytettävät ohjelmistot ovat avoimen lähdekoodin ohjelmistoja ja siten ilmaisia, voidaan tämän olevan koko järjestelmän hankintahinta, jos työn osuutta ei huomioida. Tämä on ABB SP:ltä saadun tiedon mukaan murto-osa heillä käytössä olevien kaupallisten konenäköjärjestelmien hankintahinnasta.

Ohjelman kehitystyötä tehtiin lisäksi työn tekijän kannettavalla tietokoneella, mutta se ei vaikuta POC-testien suorittamiseen, joten sitä ei esitellä.

3.1.3 Testiohjelmistot

Jetson Nanon käyttöä varten laitteistolle asennettiin seuraavat ohjelmistot.

JetPack SDK

Nvidia on julkaissut laitteilleen valmiin ohjelmistokehityspaketin, JetPack SDK:n. JetPack SDK sisältää Linux-käyttöjärjestelmän tarvittavine ajureineen, ja mahdollistaa Jetson Nanon sekä muiden laitteistojen käyttämisen. Muita JetPack SDK sisältyviä ohjelmia tai ohjelmakirjastoja ovat TensorRT ja CUDA. Nämä ovat esiteltynä alla. JetPack SDK sisältää myös valmiita esimerkkejä laitteiston ja ohjelmien käytöstä. [38]

TensorRT

TensorRT on Nvidian kehittämä syväoppimisen malli, jossa on myös päättelyketjun optimointityökalu. Tensor RT on tarkoitettu kohteen tunnistamiseen ja segmentointiin käytettäviin neuroverkkojen suorittamiseen. TensorRT suoriutuu Nvidian [39] mukaan 36 kertaa nopeammin neuroverkon päättelyketjun laskennasta kuin pelkkä prosessorilla suoritettava laskenta. Tämä kuitenkin tarkoittaa, että Tensor RT:llä suoritettu neuroverkko-ohjelma voi olla tulokseltaan epätarkempi.

CUDA

CUDA on NVidian kehittämä alusta ja ohjelmointirajapinta grafiikkasuorittimilla tehtävää yleistä laskentaa varten. CUDA:n käyttö mahdollistaa tehokkaan grafiikkasuorittimen käytön normaalisti prosessorilla suoritettavaan laskentaan.

Labelimg

Labelimg on avoimen lähdekoodin graafinen kuvien annotaatiosovellus, jolla voidaan merkitä kuviin kohteita. Kohteiden merkitsemisellä mahdollistetaan neuroverkkojen opetusdatan luominen ohjattua oppimista varten. Labelimg tuottaa merkitystä kuvasta XML-tiedoston, jossa kohdedata on joko PASCAL VOC-, YOLO- tai CreateML-formaateissa.[40]

3.1.4 Robotti

ABB Smart Powerilla on testauskäytössä Yumi – IRB 14000 -yhteistyörobotti. YuMi yhteistyörobotti on kahdella käsivarrella varustettu, hyvin ihmismäiseen työskentelyyn soveltuva laite. Tämä robotti on mahdollista hankkia usealla erilaisella valmiilla tarttujamoduulilla. Tarttujassa on aina sormitarttuja, mutta sen lisäksi voi olla imukuppitarttujia ja/tai konenäkökamera. ABB SP:lle hankittu YuMi on varusteltu imukuppitarttujilla ja kameralla.

Kameran resoluutio on 1.3 megapikseliä, linssin polttoväli 6,2 mm ja aukon koko f/5. Kamerassa on kiinteä LED-valaistus. Kameran ohjelmisto on Cognexin In-Sight, jota voidaan ohjelmoida joko ABB Integrated Vision tai Cognex In-Sight Explorer-ohjelmistoilla. [5]

Robotin kameralla otettavien kuvien käsittely on siirrettävä erilliselle tietokoneelle neuroverkolla suoritettavaa tunnistusta varten. Tämä voidaan tehdä millä tahansa tietokoneella, jolla on yhteys robotin kanssa jaettuun verkkolevyyn.

Robotti otettiin yhdeksi testilaitteistoksi sen kameran vuoksi. Tarkoituksena oli selvittää voisiko robottia käyttää sovelluksissa kuvien ottamiseen. Kuvista saatavilla tiedoilla voitaisiin vuorostaan ohjata robotin prosessia. Robotilla suoritettua testiä käsitellään kappaleessa 3.3.2.

3.2 Tuotepakkauksen sisällön tarkistus

Tuotepakkauksen sisällön tarkistus on toinen työssä toteutettu PoC-testaus. Kyseisessä tapauksessa tarkistetaan ihmisen suorittamaa työtä tuotannon loppuvaiheessa. Tuotepakkaukseen kerätään itse tuotteen lisäksi myös käyttöohjeet, kiinnitystarvikkeet sekä muut tarvittavat lehtiset. Tuotepakkauksen sisältö vaihtelee tuotteen mallin mukaan. Tässä PoC-testauksessa rajattiin testaus käsittämään yksi tuote, johon kuuluu muutama eri komponentti. Nämä ovat:

- kytkin, ”Switch”,
- vaiheiden merkitsemisarkki, ”Label set”,
- kiinnitystarvikepakkaus, ”Mounting set”,
- ohjelehtinen, ”Instructions”.

Komponentit ovat esiteltynä alla olevassa kuvassa 14. Kytkin on kuvassa vasemmalla ja on piirteiltään helposti tunnistettavissa. Vaiheiden merkitsemisarkki on paperinen valkoinen arkki, jossa on kaapeleiden merkitsemiseen tarkoitettuja irroitettavia merkkejä. Kiinnitystarvikepakkaus (kuvassa toinen oikealta) on läpinäkyvään muovipakkaukseen pakattuja metalliosia. Pakkauksen päällä on valkoinen etikettilappu. Ohjelehtinen on paperinen taiteltu asennus- ja käyttövihko.

Kuva 14. *Tuotepakkauksen komponentit*

Testissä haluttiin kokeilla, miten hyvin neuroverkko kykenee tunnistamaan pakkauksen komponentit, ja miten Jetson Nanon suorituskyky riittää neuroverkon suoritukseen. Tu-

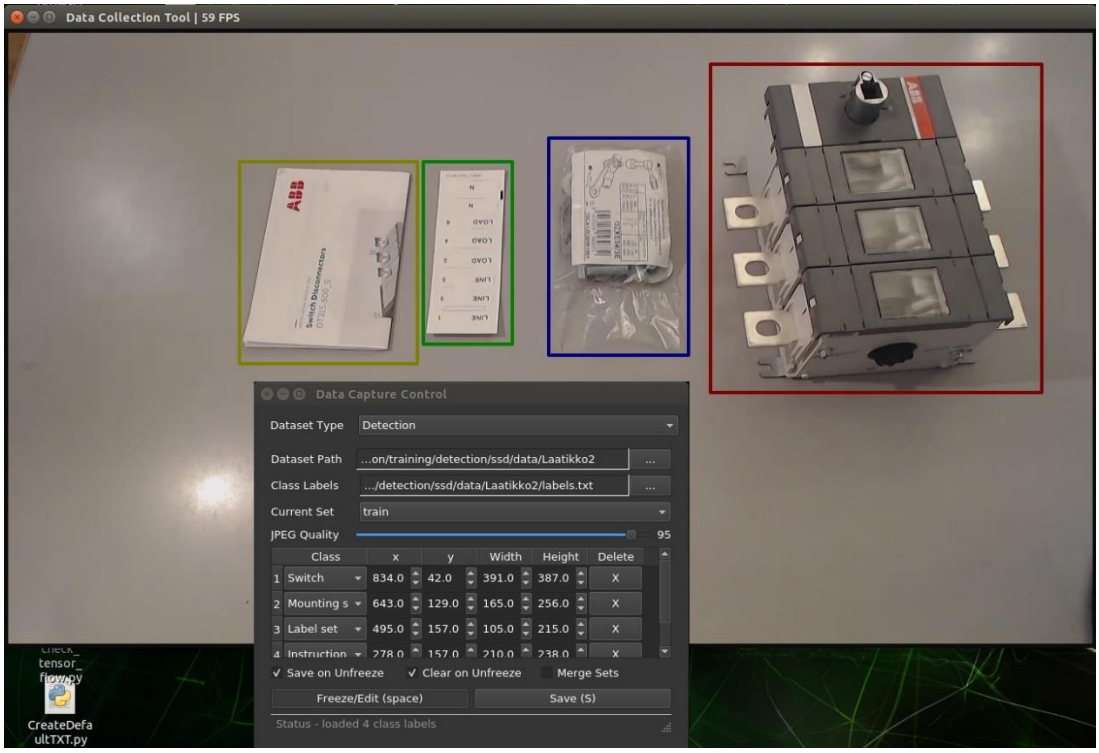
losten mittareina käytetään tunnistusvarmuutta ja ruudunpäivitysnopeutta. Komponenttien tunnistus kokeillaan niiden ollessa selkeästi esillä ja myös tilanteissa, missä komponentti ei ole täysin näkyvillä.

Käyttötarkoitukseen soveltuvia neuroverkkoja määritettäessä hankittiin aiheeseen liittyvää tietoa ja esimerkkisovelluksia internetistä. Tästä aineistosta analysoitiin erilaisia tapoja suorittaa haluttu tunnistussovellus. Koska tämän työn tekijällä ei entuudestaan ollut kokomusta neuroverkkosovellusten tekemisestä, valittiin käytettävät neuroverkot niiden dokumentaation ja esimerkkisovellusten laadun perusteella. Testit suoritettiin kahdella eri neuroverkolla, jotta voidaan tehdä vertailua ja saada kokemuksia erilaisista neuroverkoista.

3.2.1 Mobilenet-SSD

Ensimmäiseksi tutkittiin Mobilenet-SSD-neuroverkon käyttöä sovelluksessa. NVIDIA on laatinut kattavat ohjeet Mobilenet-SSD:n käytöstä Jetson Nanolla. [41] Tässä sivustossa on erilaisia ohjeita, miten Jetson Nanolle asennetaan tarvittavat sovellukset sekä miten neuroverkkoa voidaan käyttää kohteiden tunnistukseen, luokitteluun ja segmentointiin. Mobilinet-SSD-neuroverkolle löytyy valmiiksi opetetut mallit, joita voidaan hyödyntää siirto-opetusta varten. Neuroverkko on opetettu Microsoft COCO datasetillä. MS COCO sisältää 91 yleistä kohdetta eri kategorioista, esimerkiksi eri eläimiä, esineitä ja ajoneuvoja. Neuroverkolle suoritettiin osittainen uudelleen opetus käyttämällä ohjatun oppimisen menetelmää.

Mobilenet-SSD:n tutkiminen aloitettiin opetusdatan hankkimisella. Tässä tapauksessa opetusdata on valokuvia tunnistettavista komponenteista. Kuvausta varten laitteisto ja komponentit sijoitettiin pöydälle ja valaistus ja ympäristö vakioitiin.



Kuva 15. *Komponenttien merkitseminen kuvaan Data Collection Tool -työkalulla*

Jetpack SDK sisältää esiasennettuna myös kuvien opetukseen tarkoitetun työkalun, Data Collection Tool. Tällä työkalulla voidaan ottaa laitteeseen liitetyllä kameralla kuvia, sekä merkitä kuvaan mitä siitä löytyy. Kuvassa 15 on esitetty, miten kuvaan merkitään kohteille tunnistuslaatikot. Kohteelle valitaan luokka, joita tässä tapauksessa ovat eri komponenttien tyypit. Ohjelma määrittää lisäksi kohteelle sijainnin kuvassa.

```
<annotation>
    <filename>20220209-183730.jpg</filename>
    <folder>Laatikko2</folder>
    <source>
        <database>Laatikko2</database>
        <annotation>custom</annotation>
        <image>custom</image>
    </source>
    <size>
        <width>1280</width>
        <height>720</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>Label set
</name>
        <pose>unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>966</xmin>
            <ymin>196</ymin>
            <xmax>1114</xmax>
            <ymax>432</ymax>
        </bndbox>
    </object>
```

Kuva 16. *XML-tiedosto kohteista*

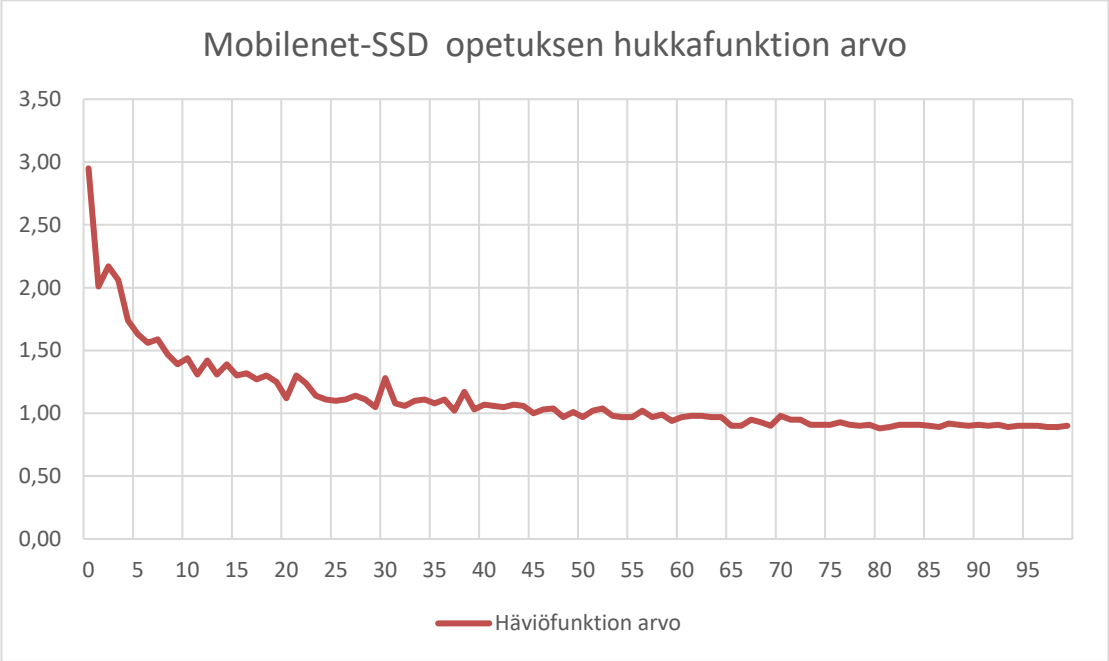
Tallennettaessa ohjelma luo xml-tiedoston, joka sisältää kuvaan merkittyjen kohteiden tiedot. Kuvassa 16 on esitetty osa XML-tiedoston sisällöstä. Ensimmäisillä riveillä ovat XML-tiedostoon liittyvän valokuvan tiedot, kuten nimi, tallennussijainti sekä kuvan koko. Tämän jälkeen on kuvassa olevien kohteiden tiedot. Tässä tapauksessa kuvasta löytyy kohde ”Label set”, jonka tunnistuslaatikon kaksi nurkkaa ovat x- ja y-koordinaateilla 966, 196 ja 1141, 432.

Komponenteista otettiin 566 kuvaa. Näistä kuvista noin 80 % on opetukseen käytettäviä kuvia, 10 % opetuksen testaukseen ja 10 % opetuksen validointiin. Kuvien jakauma on taulukon 3 mukainen.

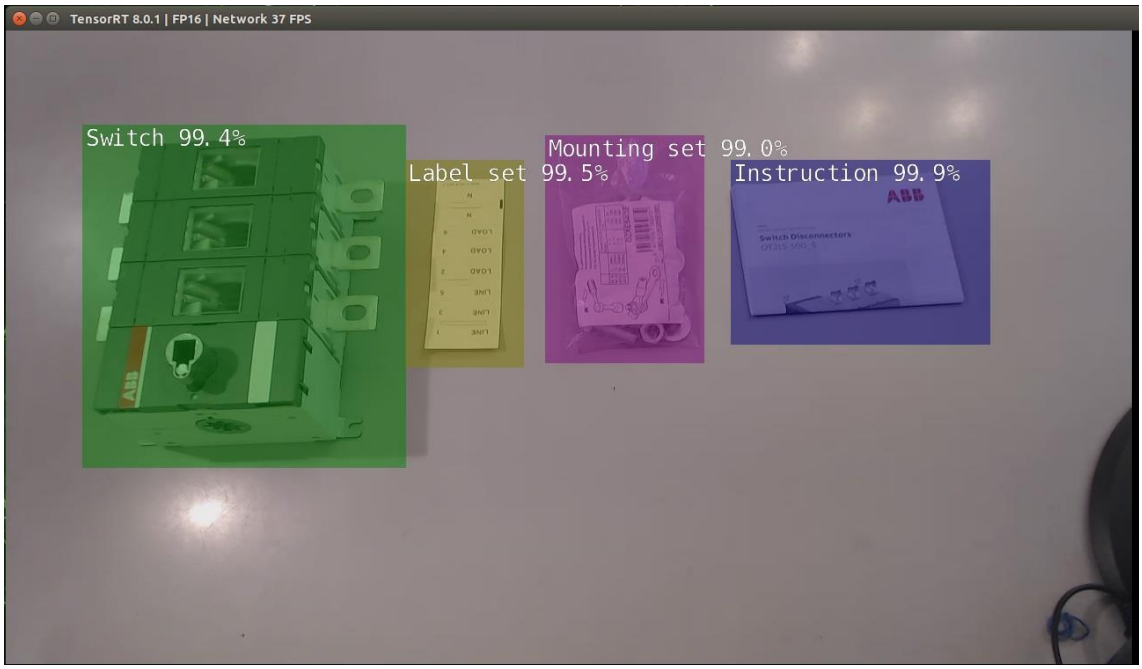
Taulukko 3. *Mobilenet-SSD, kuvien jakauma*

Komponentti kuvassa	**Opetuskuvat**	**Validointi-kuvat**	**Testikuvat**	**Yhteensä**
Kytkin	100	10	10	120
Ohjelehtinen	100	10	10	120
Kiinnitystarvi-kepakkaus	100	10	10	120
Vaiheiden merkitsemisarkki	100	10	10	120
Ohjelehtinen + vaiheiden merkitsemisarkki	20	2	2	24
Ohjelehtinen + kiinnitystarvi-kepakkaus	20	2	2	24
Vaiheiden merkitsemisarkki + kiinnitystarvi-kepakkaus	20	2	2	24
Kaikki kohteet	10	2	2	14

Neuroverkon opetus suoritettiin Jetson Nanolla. Opetuskierroksia (englanniksi epochs) suoritettiin 100, ja opetukseen kului aikaa noin kolme tuntia. Tällä päästiin neuroverkon oppimista kuvaavalla häviöfunktion arvoon 0,89. Häviöfunktion kehitys on kuvattu alla olevassa kuvassa 17. Opetuksen ensimmäisillä kierroksilla häviöfunktion arvo pienenee nopeasti, mutta tasaantuu 50 kierroksen jälkeen. Paras häviöfunktion arvo saavutettiin opetuskierroksella 93. Häviöfunktion arvo 0,98 ei ole vielä mitenkään merkittävän hyvä, mutta riittävä suoritettavaan testaukseen. 100 opetuskierrosta on neuroverkoille suhteellisen pieni, joten suorittamalla enemmän opetuskierroksia, saadaan hukkafunktion arvoa myös pienennettyä. Myös opetusdatan määrällä ja laadulla on vaikutus hukkafunktion arvoon.



Kuva 17. *Mobilenet-SSD häviöfunktion kehitys opetuksessa*



Kuva 18. *Mobilenet-SSD Kohteiden tunnistustestaus*

Opetuksen jälkeen tunnistusta testattiin samassa ympäristössä kuin opetuskuvat olivat otettu. Tunnistusvarmuus komponenteille oli erittäin hyvä. Kuvasta 18 näemme miten neuroverkko tunnistaa kaikki komponentit lähes 100 % varmuudella. Tunnistus tapahtuu reaaliajassa, ruudunpäivityksen nopeuden ollessa koko ajan yli 30 ruutua sekunnissa. Komponenttien siirtämisen aikana niitä ei kyetä tunnistamaan, mutta välittömästi liikkeen loputtua ja komponentin ollessa esillä se tunnistetaan. Osittain peitossa olevat komponentit jäävät tunnistamatta, joten voidaan todeta tämän neuroverkon toimivan vain sovelluksissa, joissa tunnistettava kohde on hyvin esillä.

Testauksen perusteella neuroverkko suoriutuu sille asetetusta tunnistustehtävästä kiitettävästi. Jetson Nanon suorituskyky riittää sekä opettamaan että suorittamaan neuroverkkoa hyvällä ruudunpäivitysnopeudella. Käytetty web-kamera toimii tässä käyttökohteessa hyvin, kun halutaan kuvata työpöydällä olevia kohteita. Kuva-ala on riittävän suuri, ja automaattitarkennus toimii hyvin tarkentuen pienen viiveen jälkeen oikealle syvyydelle.

3.2.2 YOLO V4

Mobilenet-ssd kokeilun jälkeen siirryttiin kokeilemaan YOLO V4 -neuroverkkoa kappaleiden tunnistamiseen. Opetus neuroverkolle suoritettiin Google Colab-pilvipalvelussa. Colab tarjoaa tehokasta suorituspalvelua tutkimus- ja kehitysprojekteihin, ja sopii erityisesti tekoälyn, datan käsittelyn ja koneoppimisen tutkimiseen. Colab-pilvipalvelusta on saatavilla käyttöön eritehoisia grafiikkasuorittimia.

Opetusta varten komponenteista otettiin 377 kappaletta. Kuvien kuvaukseen käytettiin älypuhelimen kameraa. Kuvien hankintaan kului aikaa kaikkinensa noin 5 tuntia. Tässä testissä haluttiin kokeilla, voidaanko YOLO saada tunnistamaan komponentit myös hyvin erilaisissa ympäristöissä ja tilanteissa, joissa komponentista on näkyvillä vain osa. Tätä varten kuvia otettiin erilaisia taustoja vasten ja erilaisissa valaistuksissa. Alla olevassa kuvassa 19 on esitetty joitakin opetusdatan kuvia.

Kuva 19. *Opetuskuvia tuotepakkauksen sisällöstä erilaisia taustoja vasten*

Kuviin merkittiin tunnistuslaatikot käyttäen Labelimg-sovellusta. Toiminnaltaan Labelimg on hyvin samanlainen kuin aikaisemmin käytetty Data Collection Tool. Tunnistettavia luokkia on neljä. Komponenttien määrä ja tausta vaihtelee kuvien välillä. Labelimg-sovelluksella saadaan joka kuvalle opetusdatan sisältävä XML-tiedosto suoraan YOLOn ymmärtämässä formaatissa. Kuva 20 esittää Labelimg-sovelluksella tehtävän kuvien merkitsemistä.

Kuva 20. *Labelimg annotaatiosovelluksen käyttö*

Kuten Mobilenet-SSD:llä suoritetussa testissä, myös tässä tapauksissa kuvista käytettiin noin 10% testaukseen ja validointiin. Taulukko 4 esittää kuvien jakauman. Tällä kertaa validointi- ja testikuvat olivat samat, koska haluttiin tutkia minkälainen vaikutus sillä on neuroverkon oppimiseen. Joka tapauksessa nämä kuvat ovat opetusmateriaalin ulkopuolelta, ja siten estävät neuroverkon ylisovittumista.

Taulukko 4. *YOLO, kuvien jakauma*

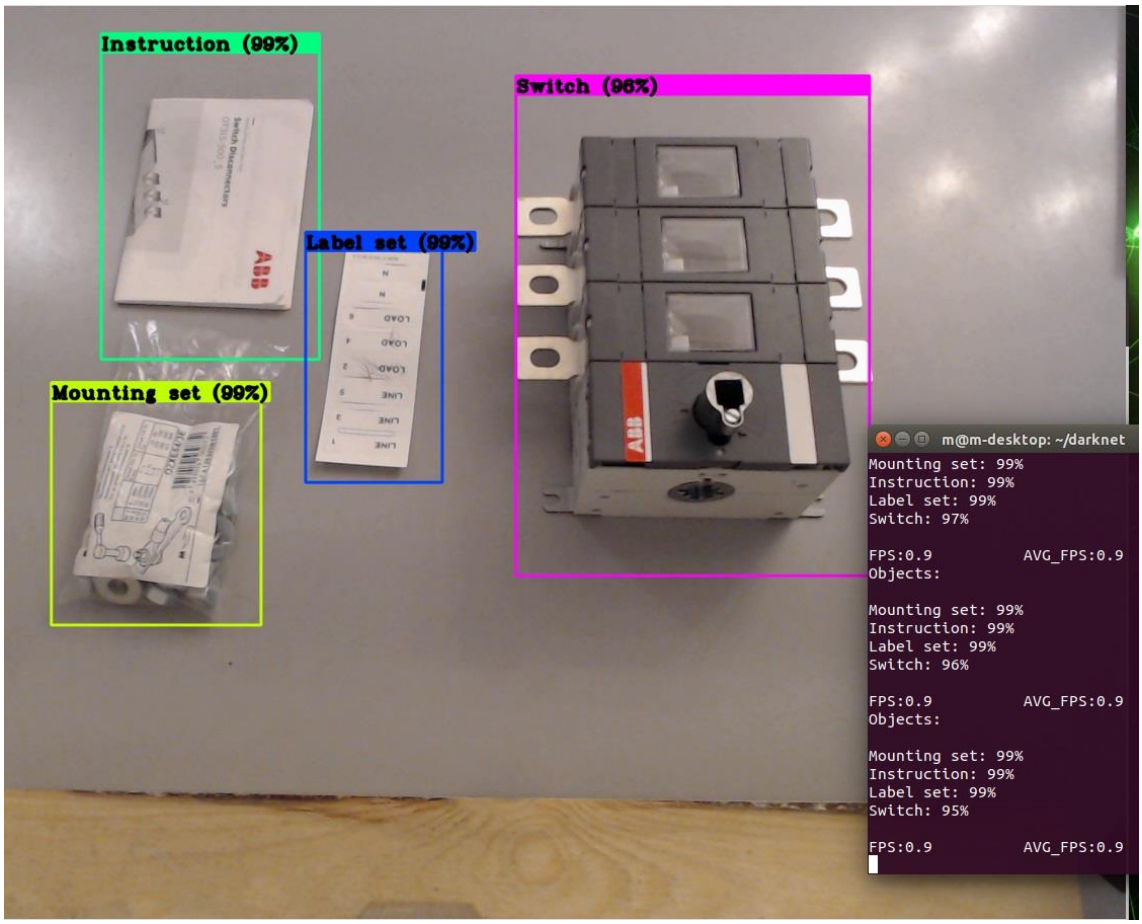
Komponentti kuvassa	**Opetuskuvat**	**Validointikuvat**	**Testikuvat**	**Yhteensä**
Kytkin	58	6	6	70
Ohjelehtinen	69	7	7	83
Kiinnitystarvikepakkaus	84	10	10	104
Vaiheiden merkitsemisarkki	66	7	7	80
Kytkin + ohjelehtinen	1	0	0	1
Kytkin + kiinnitystarvikepakkaus	1	0	0	1
Kytkin + vaiheiden merkitsemisarkki	3	2	2	7
Kiinnitystarvikepakkaus + vaiheiden merkitsemisarkki	3	0	0	3
Kiinnitystarvike-pakkaus + ohjelehtinen + vaiheiden merkitsemisarkki	5	0	0	5
Kaikki kohteet	49	6	6	61

YOLO V4 opetusta varten alkuperäiseen konfiguraatiotiedostoon tehtiin seuraavat muutokset:

- Kuvien koko muutetaan 608x608 pikseliin
- tunnistettavien kohteiden määrä muutettiin 4 kappaleeseen
- opetuserien maksimimäärä muutettiin 8000 erään
- opetuksen oppimisarvon muuttamisen kohdat 6400 ja 7200 opetuserän kohtaan

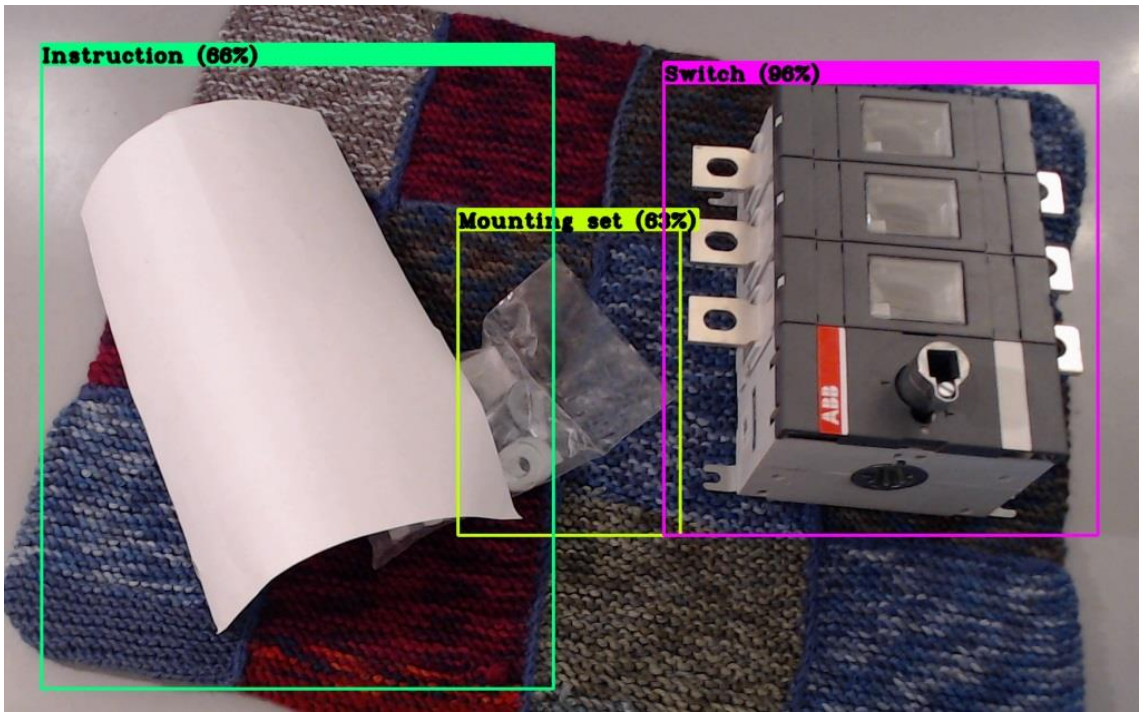
Näillä muutoksilla muutettiin syötettävän kuvan koko isommaksi, vastaamaan datan kuvakokoa. Tunnistettavien kohteiden lukumäärän muutoksella neuroverkolle kerrotaan montako erilaista kohdetta sen pitää löytää. Opetuserien määrää rajattiin, jotta opetus saataisiin valmistumaan yhden päivän aikana. Oppimisarvon kohtien muuttamisella neuroverkon opetuksessa käytettävien parametrien muuttamiskohdat muutettiin sopimaan käytettävään 8000 opetuserään.

Opettamiseen kului aikaa n. 8 tuntia, ja täydelliseen opetukseen ei parin opetuskerran aikana päästy. Colab-pilvipalvelun ilmaisen version käyttäminen katkeaa, jos sen käyttöliittymänä olevalla verkkosivulla ei tapahdu mitään tietyn ajan kuluessa. Tämä pysäyttää opetusprosessin. Opetusprosessia piti jatkaa muutamaan otteeseen aikakatkaisun jälkeen. Häviöfunktion arvoja tai kuvaajaa ei tallenneta, jos opetustapahtuma aikakatkaistaan. Siksi sitä ei ole tässä tapauksessa mahdollista esittää. Opetuksen tuloksena saatu neuroverkon painot kuitenkin osoittautuivat testissä toimivan todella hyvin.



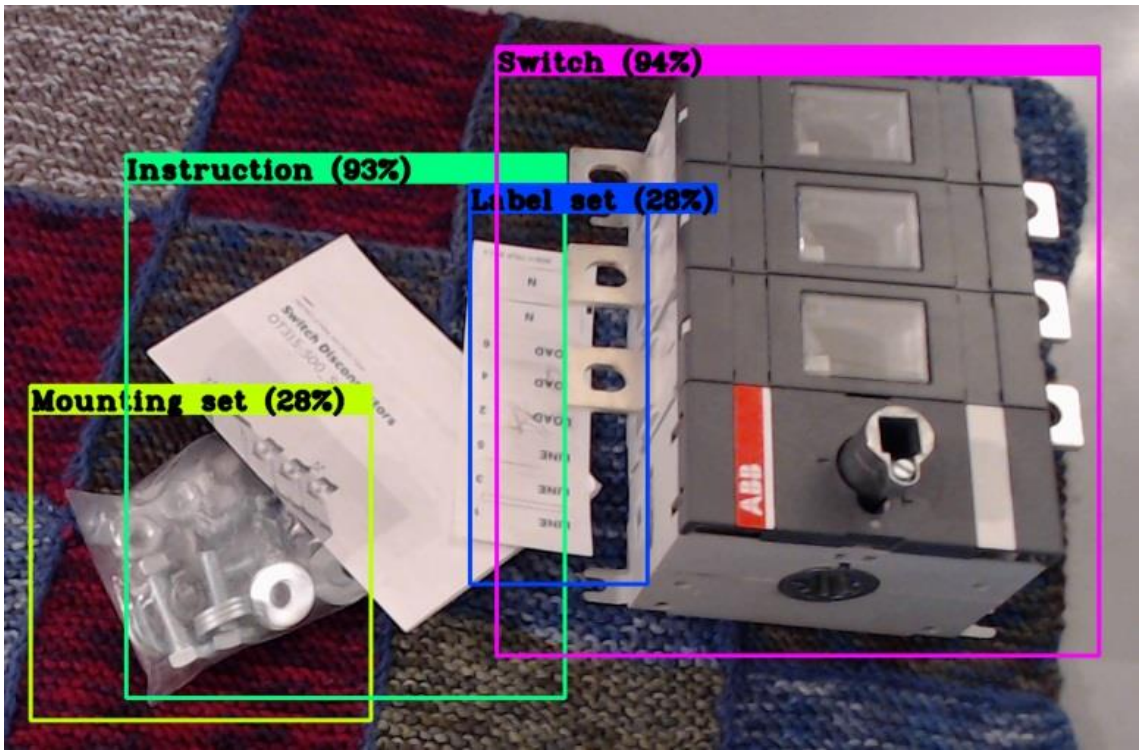
Kuva 21. *YOLO V4 kohteiden tunnistustestaus, kuva 1*

Testattaessa neuroverkon toimintaa kamera asetettiin ensin kuvaamaan pöytää ja sillä olevia kappaleita. Kuvasta 21 näemme, miten tunnistusvarmuus oli hyvää luokkaa, pysyen 90 % - 100 % välillä. Jetson Nanon suorituskyky ei kuitenkaan riittänyt kuin 1–2 ruudun sekuntivauhtiin, mikä aiheutti tuntuvaa viivettä ja kuvan nykimistä. Testillä kuitenkin voidaan todeta, että YOLO toimii tässä käytössä.



Kuva 22. *YOLO V4 kohteiden tunnistustestaus, kuva 2*

Seuraavaksi kuvaan asetettiin erilaisia taustoja, ja kokeiltiin tunnistuksen luotettavuutta. Kuva 22 esittää kohteiden tunnistuksen moniväristä taustaa vasten. Kuvassa on myös ylimääräisenä kappaleena valkoinen A4-kokoinen paperi. Kytkin löytyy kuvasta ongelmitta. Kiinnitystarvikepakkaus tunnistetaan 63 prosentin todennäköisyydellä, vaikka siitä on lähes puolet piilossa. Sen sijaan valkoinen paperi luokitellaan virheellisesti ohjelehtiseksi. Ohjelehtinen tosin on pääasiassa valkoista paperia, joten virheellinen tunnistus on ymmärrettävissä. Lisäopetuksella tähän voitaneen vaikuttaa.

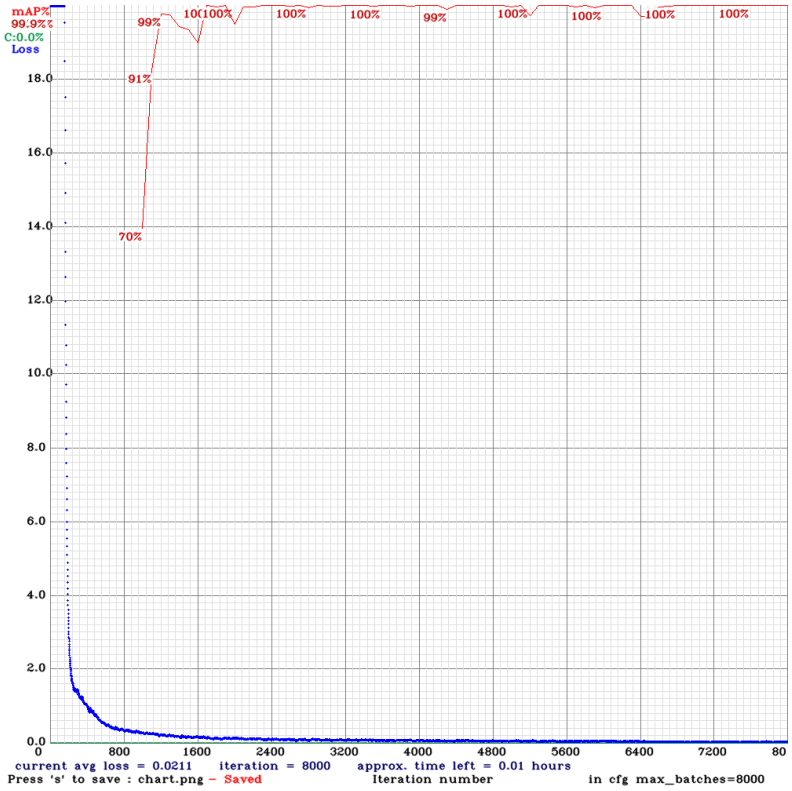


Kuva 23. *YOLO V4 kohteiden tunnistustestaus, kuva 3*

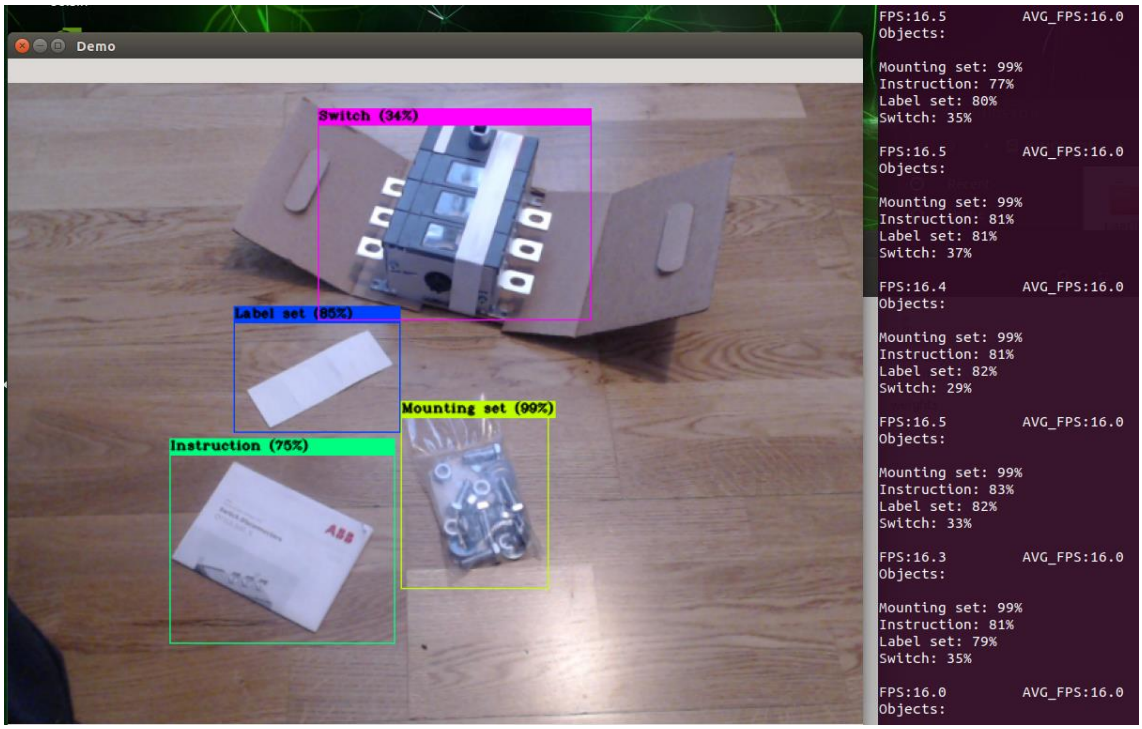
Kuvassa 23 kohteet ovat asetettu hyvin lähekkäin, osittain toistensa päälle. Neuroverkko suoriutuu näiden tunnistamisesta kohtuullisesti, selkeästi esillä olevat kytkin ja käyttöohje tunnistetaan hyvin. Sen sijaan kiinnitystarvikepakkaus ja vaiheiden merkitsemisarkki tunnistetaan vain vaivoin. Testien perusteella havaittiin, että tunnistus tapahtuu, jos kohteesta on näkyvillä vähintään puolet.

YOLOsta on myös saatavilla kevyempiä neuroverkkomalleja, ja näistä kokeiltiin YOLO V4-tiny -mallia. Neuroverkon asetustiedostoon tehtiin vastaavat muutokset kuin YOLO V4 neuroverkkomallin asetustiedostoon. Opetuskuvat olivat samat. Neuroverkon opetus suoritettiin Colab-pilvipalvelussa. Opetus tapahtui huomattavasti nopeammin kuin isomman V4 neuroverkkomallin opetus. Aikaa kului n. 4 tuntia. Aikakatkaisua ei tällä kertaa esiintynyt, ja opetus suoriutui yhdellä kerralla.

Häviöfunktion kuvaaja nähdään kuvasta 24. Häviöfunktion arvo pienenee hyvin nopeasti muutaman sadan opetuskierroksen aikana, mutta tasaantuu tämän jälkeen. Häviöfunktion arvo asettuu 0.0211 opetuksen päätyttyä. Tätä voidaan pitää hyvänä arvona, joten neuroverkko on oppinut tunnistamaan kohteet hyvin.



Kuva 24. *YOLO V4 tiny -neuroverkon opetuksen häviöfunktion kuvaaja.*



Kuva 25. *YOLO V4-tiny kohteiden tunnistustestaus*

YOLO V4-tinyn testissä päästiin tunnistusnopeuteen 16 ruutua sekunnissa, mutta tunnistusvarmuudet olivat kohteille huonommat, keskimäärin 50 ja 70 prosentin välillä. Lähempänä olevat kappaleet tunnistetaan paremmin, mikä näkyy kuvasta 25. Etäisyyden ja sitä kautta kappaleen koon pienentyessä tunnistusvarmuus heikkenee. YOLO V4-tinyä voidaan kuitenkin pitää toimivana neuroverkkona Jetson Nanon kaltaisessa laitteessa, jonka resurssit ovat rajalliset.

3.3 Tuotteen laadun tarkistus

Toiseksi käyttökohteeksi neuroverkoille valittiin tuotteen laadun tarkistus kokoonpanon loppuvaiheessa. Tuotteiden kokoonpanolinja on nykyaikainen roboteilla toteutettu tuotantolinja. Kokoonpanon eri vaiheissa on laadun varmistamiseksi tarkistuksia sekä erilaisilla sähköisillä antureilla että myös konenäön avulla. Tuotteen lopputarkastukselle ei kuitenkaan ole tarkistusta, vaan operaattorit tarkistavat tuotteen silmämääräisesti ennen varastointia. Tuotteessa voi olla virheitä, joita ei ole aikaisemmissa tarkistuksissa huomattu tai jotka ovat syntyneet tarkistuksen jälkeen.

Laadun tarkistuksella on hyvin merkittävä osa tuotannossa. Viallisen tuotteen päätyminen asiakkaalle aiheuttaa kustannuksia ja johtaa pahimmillaan ison tuotantoerän takaisinvetoon. Ihminen toimii hyvin laaduntarkastuksessa ja tarkistaa tuotteen kunnon helposti silmäillen. Ihminen kuitenkin väsyy ja turtuu tekemäänsä työhön, jolloin viallisuus

voi jäädä huomaamatta. Konenäöllä suoritettava laaduntarkastus sen sijaan toimii väsymättä ja luotettavasti yhä uudelleen.

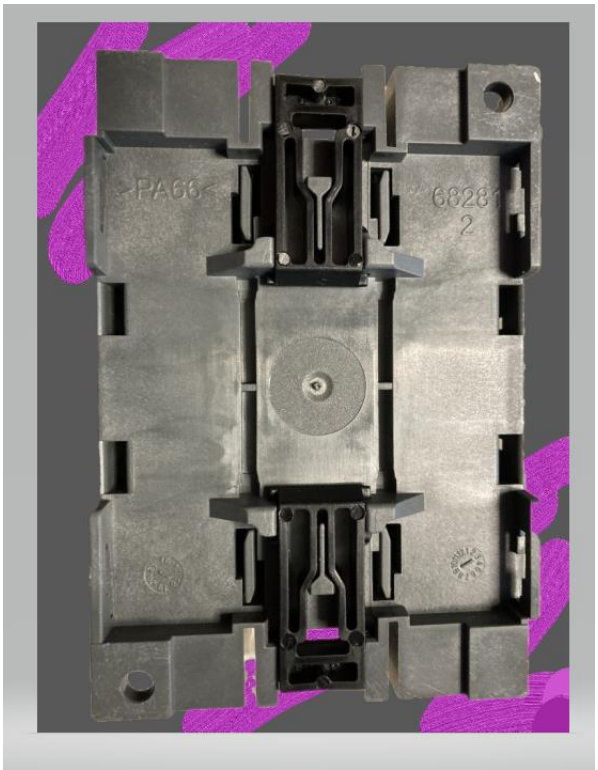
Konenäkösovelluksen on kuitenkin oltava informatiivinen tilanteissa, joissa ihmisen on osallistuttava prosessiin tai selvitettävä mahdollisia ongelmatilanteita. Keskusteluissa ABB henkilöstön kanssa tuli esille toive, että tähän sovelluksen informatiivisuuteen kiinnitettäisiin huomiota. Ratkaisuksi tähän sopisi erilaiset symbolit, merkinnät ja värien käyttö.

Laaduntarkistuksen PoC-testin tavoitteena oli löytää neuroverkko, joka kykenee tunnistamaan viallisia tuotteita opettamalla sille vain hyviä tuotteita. Tällä tavalla tuotteissa esiintyvät viat voisivat olla mitä tahansa ja miten harvinaisia tahansa, mutta neuroverkko tunnistaisi tuotteen olevan viallinen koska se poikkeaa opetetusta hyvästä tuotteesta. Lisäksi haluttiin yllä kuvattua informatiivisuutta tuloksen esitykseen. Neuroverkkojen valintaa varten suoritettiin tässäkin testissä olemassa olevien neuroverkkojen ja esimerkkisovellusten kartoitus ja analysointi. Dokumentaation ja esimerkkisovellusten laatu olivat tärkeimmät valintakriteerit testin neuroverkkojen valintaan.

3.3.1 YOLO V4

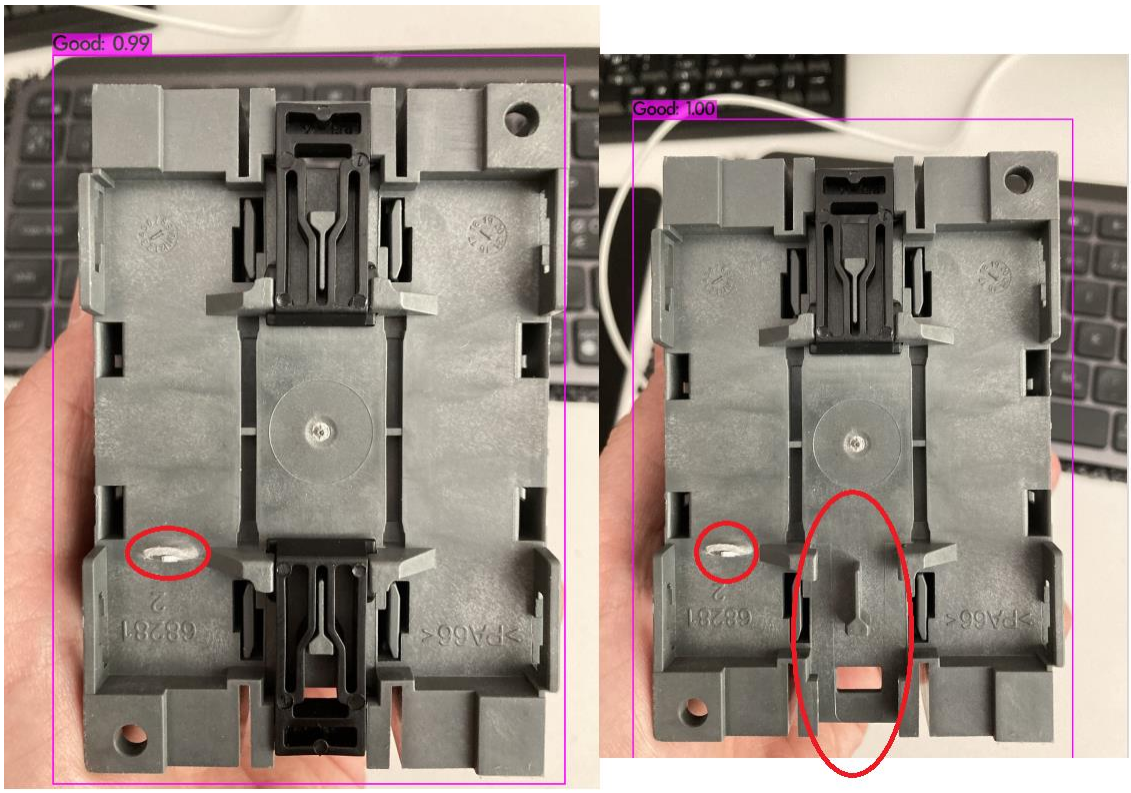
YOLO V4-neuroverkolla suoritettavan testin tarkoituksena oli kokeilla, voidaanko neuroverkko opettaa pelkillä hyvillä kuvilla. Jos neuroverkolle näytetään tuotteen kuva, jossa on jokin vika, saadaanko pienempi tunnistusvarmuus. Tätä varmuuslukemaa voitaisiin käyttää tunnistettaessa, onko kappale hyvä tai huono. Hyvän kappaleen tunnistamisen ehdoksi asetettiin varmuuslukema, joka olisi vähintään 98 prosenttia. Tämän alle olevat tulkittaisiin viallisiksi.

Tuotteen laadun tarkistukseen opetettiin YOLO V4 neuroverkko samaan tapaan kuin laatikon sisällön tarkistukseen. Testiä varten kerättiin 134 erilaista kuvaa tuotteen pohjasta. Kuvia tuotettiin myös keinotekoisesti muokkaamalla kuvan taustaa erilaisilla tehosteilla. Kuvassa 26 on esimerkki taustan muokkauksesta. Taustaan on lisätty satunnaisesti valittuja värejä ja muotoja. Näin kuvia saatiin lisää opetusta varten.



Kuva 26. *Muokattu taustakuva*

Valokuvien koko pienennettiin ja niihin merkittiin tunnistettava kohde yksinkertaisesti nimellä ”Good”, eli haluamme tunnistaa vain hyvän tuotteen. Neuroverkon opetusta varten asetustiedostoon muutettiin tunnistettavien luokkien määräksi 1. Tässä tapauksessa halutaan tunnistaa vain hyvä tuote. Opetuskertojen määräksi asetettiin 2000. Opetus suoritettiin Google Colab -pilvipalvelussa, ja se kesti n. 3 tuntia. Pilvipalvelun automaattisen aikakatkaisun vuoksi tästäkään opetuksesta ei ole saatavissa häviöfunktion arvon kuvaajaa.



Kuva 27. *Virheellisten tuotteiden testaus opetetulla neuroverkolla. Virheelliset kohdat merkitty punaisella värillä*

Opetuksen jälkeen neuroverkon painotiedosto ladattiin Jetson Nanolle kokeilua varten. Neuroverkon tunnistuskykyä kokeiltiin 20 kuvalla, joista 10 oli viallisia. Neuroverkko kyllä tunnistaa tuotteet hyvin, mutta ei erottele viallisia ja hyviä. Tämän näemme kuvasta 27, missä on esitetty kaksi kuvaa, joissa molemmissa on jonkinlainen vika. Vasemmanpuoleisessa tapauksessa tuotteen sisäinen komponentti on rikkonut tuotteen pohjan. Oikeanpuoleisessa on lisäksi poistettu musta DIN-kiinnike. Molemmissa tapauksissa tunnistusprosentti oli hyvin suuri. Tämä ei ollut toivottu tulos, koska tunnistusprosenttia ei voida käyttää erottelemaan hyvät kappaleet huonoista. Selitys tälle on se, että YOLO V4 neuroverkko on optimoitu kohteen tunnistamiseen. YOLO tunnistaa kohteen, vaikka siinä olisi poikkeamia opetetusta. Kuvissa esiintyvät virheet ovat niin pieniä, että ne eivät muuta kappaleen ulkomuotoa paljoakaan, joten YOLO tunnistaa ne hyviksi.

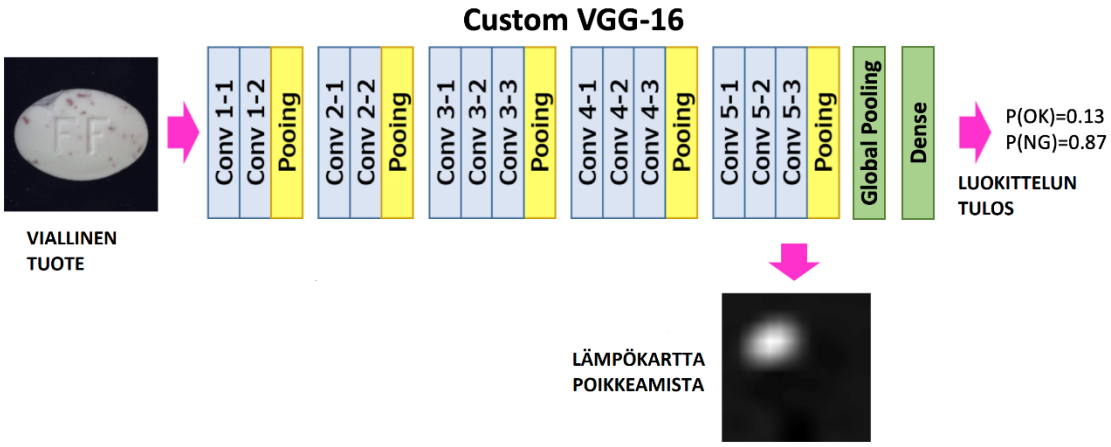
Tästä tultiin johtopäätökseen, että YOLOn opettaminen laaduntarkastukseen pitäisi toteuttaa eri tavalla. Opetusta varten jokaiselle virheelle pitäisi tehdä oma luokkansa, ja määrittää ne kuviin tunnistuslaatikoilla. Siten neuroverkko saadaan etsimään kuvista virheitä. Jos tuotteessa esiintyy myöhemmin virheitä, jotka eivät sisälly neuroverkon opetusdataan, neuroverkko ei tunnista niitä. Opetus joudutaan tekemään uudelleen datalla, joka sisältää kuvia näistä virheistä, jotta ne kyetään tunnistamaan. YOLO-neuroverkolla

on toteutettu laadunvalvonnan sovelluksia, esimerkiksi A. LI *et al* [43] tutkimuksessa tunnistetaan teräslevystä virheitä. Tutkimuksessa neuroverkko opetetaan tunnistamaan kuusi erilaista virhettä käyttäen ohjattua oppimista. Neuroverkko kykenee tunnistamaan 99 prosenttia virheistä. Tämä lähestymistapa ei kuitenkaan ollut tavoiteltu, koska neuroverkko kykenisi tunnistamaan vain sille opetettuja virheitä. Muut virheet jäisivät tunnistamatta.

3.3.2 VGG16

Lisätutkimus neuroverkkojen käytöstä laaduntarkastukseen toi vastaan O. Chernytskan [44] kehittämän sovelluksen VGG16 neuroverkolle. Tämä valittiin tarkempaan tutkimiseen ja testaukseen koska sovellus vaikutti toiminnaltaan halutulta ja dokumentaatio oli kattava.

Hänen käyttämänsä neuroverkko pohjautuu VGG-16 neuroverkkoon, joka on opetettu ImageNet-kuvakirjastolla. Neuroverkon luokitteluosa on muutettu siten että alkuperäinen kolme kerrosta on vaihdettu yhdistyskerrokseen ja tuloskerrokseen. Kuva 28 esittää neuroverkon rakenteen. Opetettaessa neuroverkon tunnistuskerroksen painot pidetään muuttumattomina, ja vain loppuosan luokittelukerrokset ja viimeinen tunnistuskerros opetetaan uudelleen siirto-opetuksella. Tämä mahdollistaa pienemmän opetusdatan käytön ja nopeamman opetuksen. Neuroverkko opetetaan tunnistamaan vain hyviä kuvia, jolloin se kykenee tunnistamaan kaikenlaiset poikkeamat. [44]



Kuva 28. *Muokattu VGG16 neuroverkko ja lämpökartta, muokattu lähteestä [44]*

Lisäksi sovelluksessa on visualisointi, jolla voidaan havainnollistaa kappaleista löydettävät virheet. Tämä perustuu neuroverkon viimeistä luokittelukerroksesta saatavaan läm-

pökarttaan havaituista poikkeamista. Kuvasta 29 nähdään, miten lämpökartan perusteella sovelluksessa luodaan vian sijainnin osoittava merkintä kuvaan. Tämä informatiivinen ominaisuus oli yksi asia mitä laadunvalvontasovelluksessa tavoiteltiin.

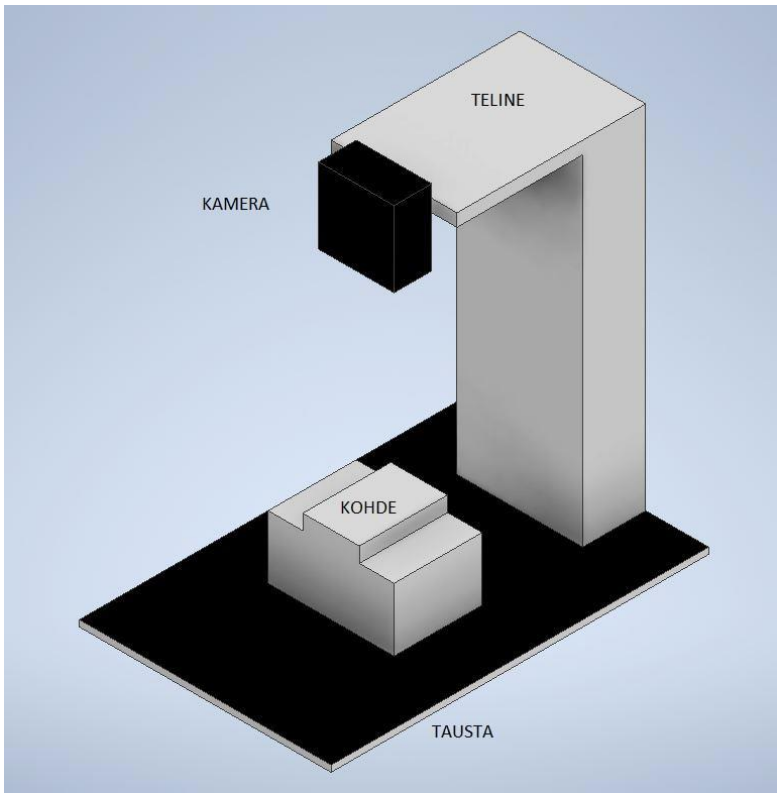
Kuva 29. *Lämpökartan perusteella tehtävä vian merkintä, muokattu lähteestä [44]*

Datan hankinta

Laaduntarkistuksessa datan laadulla on suurempi merkitys kuin kohteen tunnistuksessa. Laaduntarkastuksessa tasalaatuisesta tuotteesta halutaan tunnistaa pienetkin poikkeavuudet. Opetusdatana käytettävien kuvien laatuun ja tasaisuuteen on kiinnitettävä huomiota, ettei neuroverkko opetuksessa opi tunnistamaan vääriä asioita. Esimerkiksi kohteen taustan vaihtelu voi aiheuttaa neuroverkon oppivan tunnistamaan vain kohteen eikä poikkeamaa kohteessa.

O. Chernytskan [44] mallissa käytetään MVTEC Anomaly Detection Dataset -kuvakirjastoa [45], jossa 15 erilaista kohdetta, kuten esimerkiksi pullo, lääkepilleri ja mutteri. Yhdestä kohteesta on noin 400 kuvaa, jotka ovat jaettu hyviin ja huonoihin suhteessa 70/30. Kuvakirjaston on julkaistu Creative Commons 4.0 lisenssillä vapaaseen ei-kaupalliseen käyttöön. Kuvakirjastoa ei kuitenkaan käytetty tässä tutkimuksessa muuten kuin mallina luotaessa dataa neuroverkon opettamista varten.

Datan keräämistä varten kuvausolosuhteet vakioitiin käyttämällä kiinteää kameran telinettä sekä tummaa taustaa. Taustan vaihtelu ja kappaleen koko kuvassa saatiin näin vakioitua. Kuvassa 30 on esitetty havainnemalli telineestä. Kamera sijaitsee suoraan kuvattavan kappaleen yläpuolella. Kuvausalusta on musta, jolloin taustan vaihtelu minimoidaan sekä kappale erottuu taustasta hyvin. Tässä tapauksessa laajakulmalinssillä varustettu kamera ei ole paras valinta kameraksi, koska linssi aiheuttaa vääristymiä kuvan reuna-alueille. Kuvat ovat kuitenkin hyödynnettävissä, kun kappale on keskellä kuvaa.



Kuva 30. *Havainnemalli kuvaustelineestä*

Kuvia kerättiin yhteensä 129. Osa näistä luotiin muokkaamalla otettuja kuvia erilaisin tehostein kuvankäsittelyohjelmalla. Aikaa tähän koko prosessiin kului noin 16 tuntia. Kuvien jakauma on esitetty alla olevassa Taulukko 5. Opetus tapahtuu vain hyviksi luokitetuilla kuvilla. Huonoiksi luokiteltuja kuvia on 28 prosenttia kaikista kuvista.

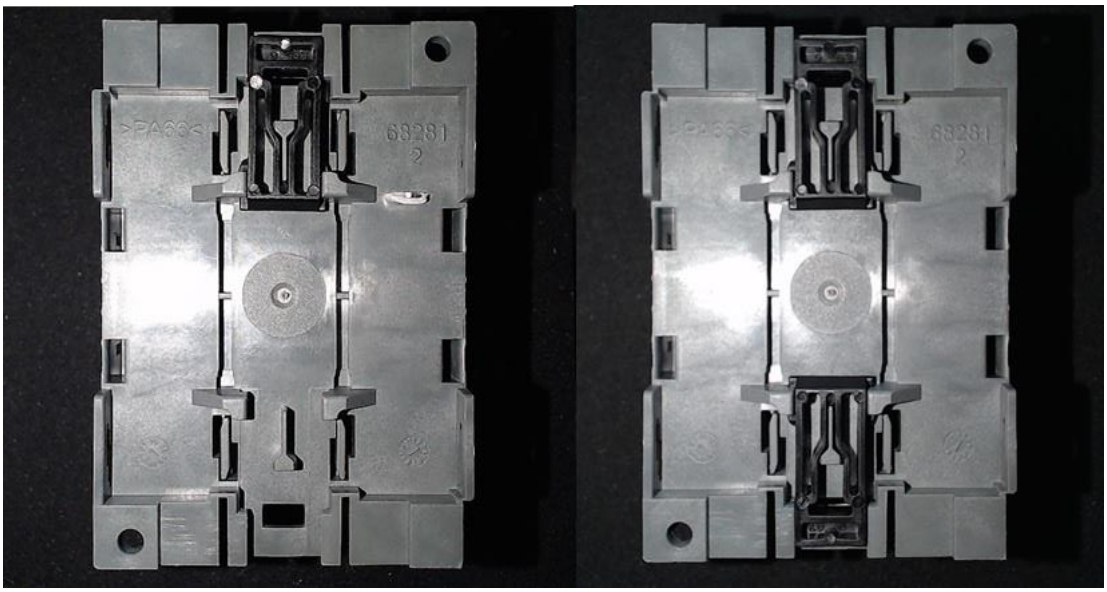
Taulukko 5. *VGG16 neuroverkon opetuksessa käytettyjen kuvien jakauma*

	Opetuskuvat	**Testikuvat**	**Yhteensä**
Hyvä	75	18	93
Huono	0	36	36

Huonojen kappaleiden viat olivat taulukon 6 mukaisesti. Yleisin vika kuvassa on puuttuva DIN-kiinnike. Tämä vika löytyy 75 prosentista kuvia. Rikkinäinen pohja esiintyy 42 prosentissa kuvista ja aukinaisen kannen osoittava vika 25 prosentissa kuvista.

Taulukko 6. *VGG16 vikojen jakauma*

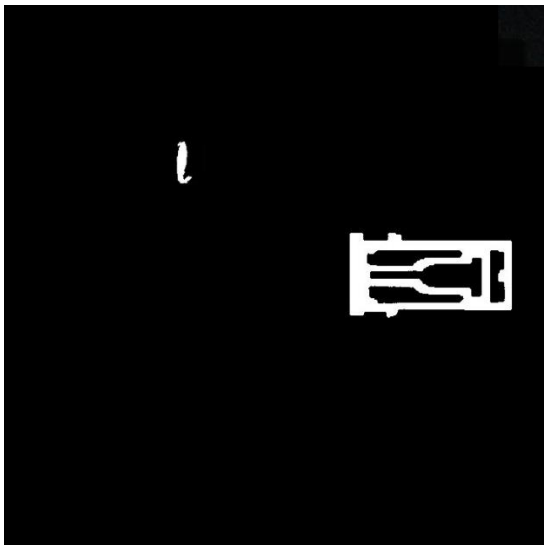
Vika	**Kuvien määrä**
Pohja rikki	6
DIN-kiinnike puuttuu	6
Molemmat DIN-kiinnikkeet puuttuvat	6
DIN-kiinnike puuttuu + pohja rikki	6
DIN-kiinnike puuttuu + kansi auki	3
Molemmat DIN-kiskot puuttuvat + kansi auki	3
Molemmat DIN-kiskot puuttuvat + pohja rikki	3
Kansi auki	3



Kuva 31. *Hyvän ja huonon kappaleen pohja*

Kuvat tallennettiin käsittelyä varten tietokoneelle. Kuvassa 31 on esitetty sekä hyvän että huonon kappaleen kuvat. Tummalla taustalla ja sopivalla valaistuksella saadaan kappale erottumaan hyvin. Ylemmässä hyvän kappaleen pohjassa on näkyvillä molemmat mustat DIN-kiskokiinnikkeet ja kappaleen pohja näyttää virheettömältä. Alemmassa kuvassa esitetty viallinen kappale, josta voidaan havaita selkeästi puuttuva musta DIN-kiskokiinnike sekä pohjan vaurio.

Tässä sovelluksessa VGG16 neuroverkon opetusta varten tarvitaan myös maskiksi kutsuttu kuva, johon on merkitty tunnistettavat, tässä tapauksessa vialliset, kohdat. Kuvassa 32 on maski, josta nähdään kuvassa esiintyvät virheelliset kohdat. Kuvaan on merkitty valkoisella värillä virheelliset kohdat. Tätä maskia käytetään neuroverkon opetuksessa tarkistukseen, kuinka hyvin neuroverkko löytää vialliset kohdat. Jokaisesta viallisen tuotteen kuvasta tehtiin erillinen maskikuva.

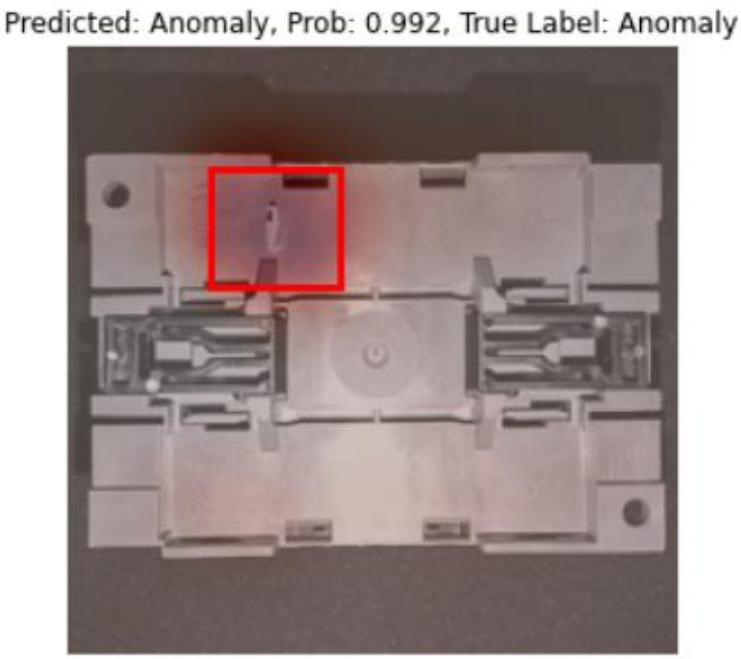


Kuva 32. *Viallisten kohtien maski*

Neuroverkon opetus ja testaus

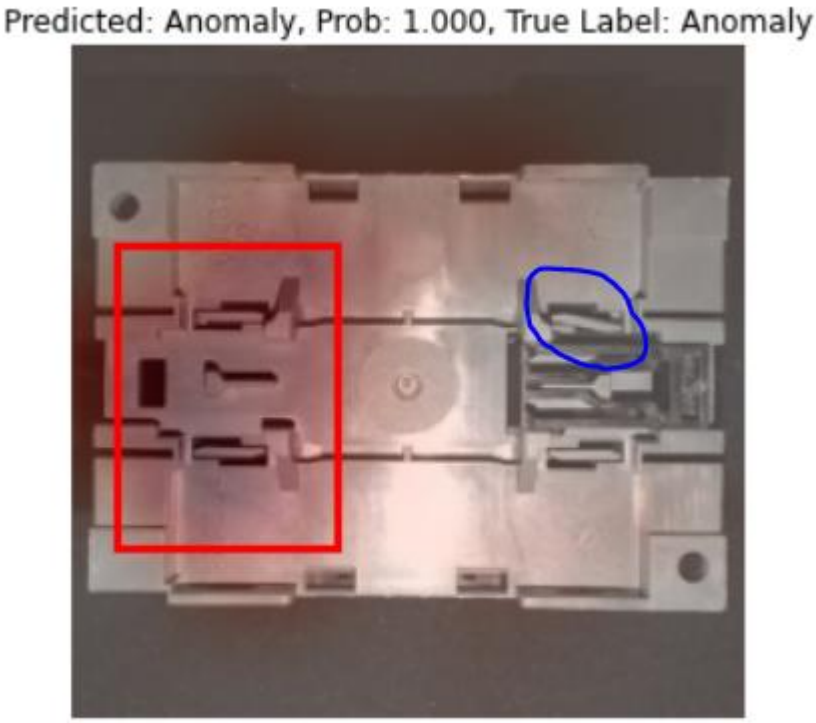
VGG16 neuroverkon opetus suoritettiin Google Colab-pilvipalvelussa käyttäen O. Chernytskan [46] valmista työkirjaa. Tämä työkirja on jaossa avoimen lähdekoodin periaatteiden mukaisesti. Työkirjassa on valmiiksi määritettynä asetukset neuroverkon opettamiseksi, ja lähinnä käyttäjän tehtäväksi jää määrittää oikeat polut käytettävän datan hakemistoille.

Työkirjan loppuun on ohjelmoitu myös neuroverkon testausmahdollisuus, jota käytettiin tässä testissä neuroverkon kokeiluun. Neuroverkolle syötettiin tunnistettavaksi uusia kuvia sekä virheellisistä että virheettömistä kappaleista. Kuvassa 33 on esitetty testaus yhdelle vialliselle kappaleelle. Kappaleen pohjassa oleva rikkoutunut kohta on hyvin tunnistettu, ja sen ympärille on piirretty ruutu. Tunnistus on myös hyvin luotettava, todennäköisyyden ollessa 99,2 prosenttia.



Kuva 33. *VGG16 neuroverkolla tunnistettu viallinen kappale, kuva 1*

Myös puuttuva musta DIN-kiskokiinnike havaitaan luotettavasti, kuten kuvasta 34 voidaan nähdä. Sen sijaan toinen virhe, vajaa kannen lukittuminen kiinnityskielekkeen asennon perusteella, jää neuroverkolta huomaamatta. Tämä on merkitty kuvaan sinisellä ympyrällä. Ohjelmaa tarkemmin tutkimalla tämän todettiin olevan ominaisuus, ohjelma kykenee tunnistamaan vain yhden virheen kerrallaan. Kannen lukittuminen voidaan havaita paremmin sivusta otetusta kuvasta, joten tämä vika voidaan havaita varmemmin toisesta kuvasta.



Kuva 34. *VGG16 neuroverkolla tunnistettu viallinen kappale, kuva 2*

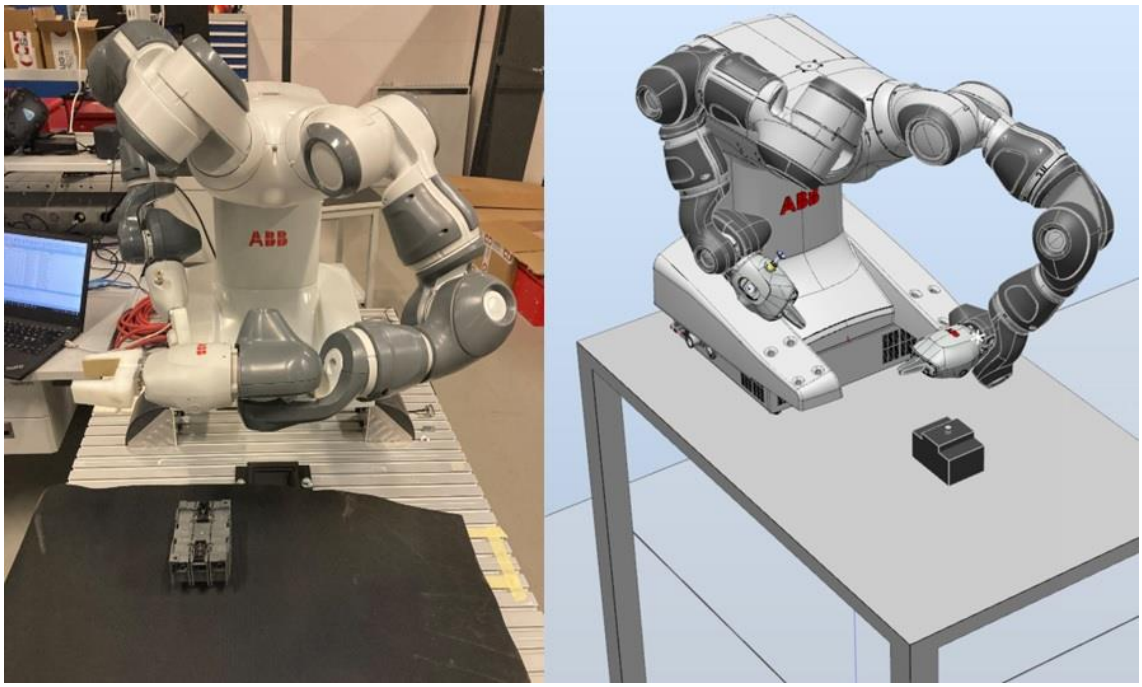
Testauksen perusteella voidaan sanoa VGG16 neuroverkon toimivan hyvin tässä käytössä. Neuroverkon tunnistusvarmuutta voidaan parantaa opettamalla neuroverkko uudelleen isommalla määrällä lähtödataa. Jos laitteisto otetaan tuotantokäyttöön, voidaan dataa tallentaa myös normaalin käytön aikana. Siten saadaan helposti suuri määrä opetusdataa.

Robotin testaus

Seuraavassa vaiheessa haluttiin kokeilla robotin tarttujan kameralla otettuja kuvia neuroverkon tunnistuksessa. Lisäkannustimena tähän testiin oli mahdollisuus löytää robotille soveltuva käyttökohde laaduntarkastustehtäviin.

Kuvausjärjestelyt

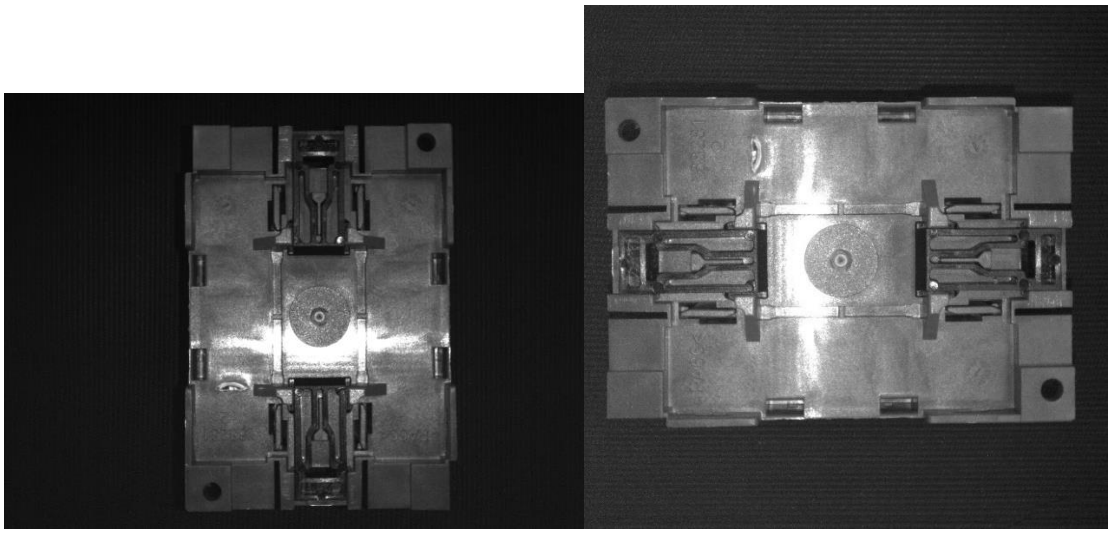
Robotin käsivarren tarttujat soveltuvat suoraan tai pienillä muutoksilla tarttumaan tuotteeseen. YuMi-robotin käsivarsissa olevilla kameroilla voidaan kuvata kappale sen ollessa joko tarttujassa tai tarvittaessa kappale voidaan laskea pöydälle kuvausta varten. Riittävän hyvään valaistukseen on kiinnitettävä huomiota, jotta kuvasta saadaan laadukas.



Kuva 35. *YuMi-robotti kuvaamassa tuotetta*

Kuvassa 35 on testijärjestelyistä kuva, vasemmalla puolella oikeasta robotista ja oikealla puolella RobotStudio-sovelluksella luotu kuva. Tuote sijaitsee tummalla taustalla robotin edessä. Tummalla taustalla saadaan kuvan tausta häivytettyä. Robotin käsivarressa oleva kamera on käännetty kappaletta kohti. Kuvausasetelma olisi mahdollinen myös tuotannossa, tosin kappaleen pohjaa kuvatessa olisi syytä asettaa kappale telineeseen, ettei se kaadu.

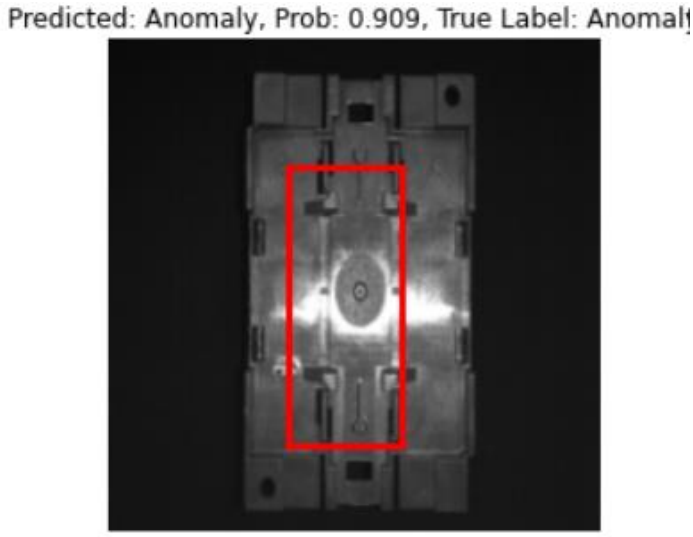
Robotilla tehdyn testin perusteella robotin integroitu kamera kykenee ottamaan riittävän laadukkaita kuvia neuroverkolla suoritettavaa tunnistusta varten. Kuvassa 36 on esitetty robotin kameralla otettu kuva. Koska tarttujan kamerassa sijaitseva valo on suoraan kameran vieressä, se aiheuttaa herkästi ylivalotuksen heijastaviin pintoihin kuten kuvasta nähdään. Laadunvalvonnan sovelluksen kannalta kiinnostavat piirteet ovat kuitenkin kuvasta hyvin havaittavissa. Kuvankäsittelyllä voidaan entisestään parantaa kuvan käyttökelpoisuutta. Kuvassa oikeanpuoleinen kuva esittää kuvankaappausohjelmalla muokattua kuvaa kappaleesta. Kuvan valoisuutta on lisätty, kuva on käännetty 90 astetta ja rajattu. Kuvan valoisuuden muokkauksen myötä olisi mahdollista pienentää kuvauksessa käytettyä valaistusta ja siten materiaalin heijastusta. Kuvan tärkeät piirteet ovat paremmin kuin muokkaamattomasta kuvasta.



Kuva 36. *Yumi-robotin integroidulla kameralla otettu kuva tuotteesta sekä käsitelty kuva*

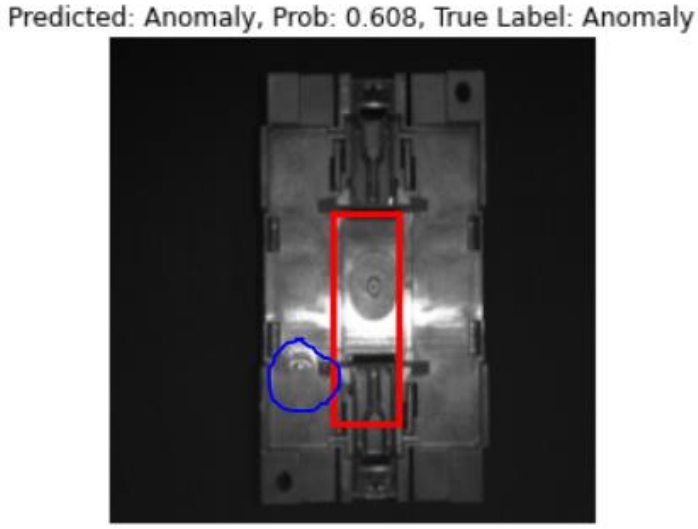
Robotin kameralla otettuja kuvia syötettiin neuroverkkoon tunnistettavaksi. Kuvien tunnistus toimi hyvin, saavutetut tunnistukset virheellisille tuotteille olivat 60 % – 90 % varmuuksien luokkaa. Neuroverkko tunnisti joitakin hyvän tuotteen kuvia väärin, todennäköisesti opetuskuvista merkittävästi poikkeavan materiaalin heijastuksen vuoksi.

Kuvassa 37 on esitetty robotin kameralla otetun kuvan tunnistus. Tunnistusvarmuus on hyvä, 90,9% ja luokittelu on oikein. Puuttuvat DIN-kiskokiinnikkeet on havaittu.



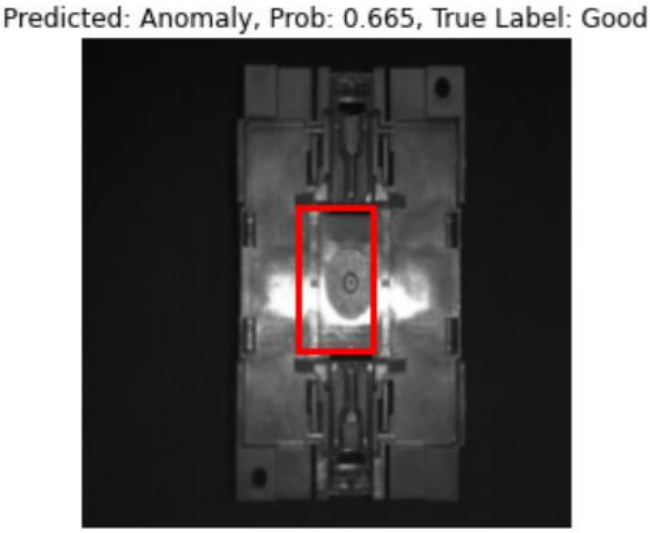
Kuva 37. *Robotin kameralla otetun kuvan tunnistus neuroverkolla, puuttuvat DIN-kiinnikkeet*

Sen sijaan pohjassa näkyvä rikkoutunut kohta jää neuroverkolta löytymättä, ja virheen tulkitaan olevan tuotteen keskellä olevassa kiiltävässä alueessa. Tämä näkyy kuvassa 38. Tunnistettu virheellinen alue on merkitty punaisella, kun todellinen virhe löytyy sinisellä merkityltä alueelta. Tunnistusvarmuudeksi saadaan 60,8 %.



Kuva 38. *Robotin kameralla otetun kuvan tunnistus neuroverkolla, rikkoutunut pohja*

Kuvassa 39 on neuroverkon tunnistama viallinen tuote, joka on todellisuudessa hyvä. Neuroverkko ei siis toimi täydellisesti, vaan pieni opetusdatan määrä tuottaa virheellisiä tunnistuksia. Lisäksi opetusmateriaalissa ei ole yhtään kuvaa tästä kyseisestä kuvakulmasta ja valaistuksesta, mikä vaikuttaa tulokseen. Voidaan kuitenkin sanoa, että tunnistus toimii ja robotin käyttämisessä kappaleiden kuvaukseen on jatkokehityspotentiaalia.



Kuva 39. *Robotin kameralla otetun kuvan tunnistus neuroverkolla, väärä tunnistus.*

4. TULOKSET

Tässä osiossa käsitellään testeistä saatuja tuloksia ja arvioidaan niiden vaikutusta.

Tutkimuskysymykseen, kuinka avoimen lähdekoodin oppivaa konenäköjärjestelmää voidaan hyödyntää teollisuudessa, selvitettiin ratkaisua aiheesta kertovan kirjallisuuden, aikaisemmin tehtyjen tutkimusten ja käytännön kokeilujen kautta. Selvitystyön perusteella löydettiin kaksi käyttökohdetta sekä laitteisto, joilla suoritettiin Proof-of-Concept-testit. Testien tulokset ovat esitelty kappaleissa 4.1 ja 4.2. Tulosten arviointi on kappaleessa 4.3.

4.1 Laatikkopakkauksen PoC tulokset

Laatikkopakkauksen PoC testauksen tuloksina saatiin tietoa kahden neuroverkon toimivuudesta tuotepakkauksen sisällön tarkistuskäyttöön. Molempia neuroverkkoja voidaan käyttää tähän tarkoitukseen, ja siten voidaan myös todeta, että avoimen lähdekoodin tekoälysovelluksia voidaan hyödyntää tämänkaltaisessa käytössä.

Mobilenet-SSD toimii Jetson Nanolla sujuvalla 40 ruudun päivitysnopeudella, kun taas YOLO:n ruudunpäivitys jää parhaimmillaankin 1–2 ruutuun. Molemmat neuroverkot tunnistavat kohteet hyvin, tunnistusvarmuuden ollessa yli 90 prosenttia selkeästi esillä oleville kappaleille. Osittain piilossa olevien kohteiden tunnistuksesta Mobilenet-SSD ei suoriudu, kun taas YOLO V4 kykenee siihen. Tunnistusvarmuus voi tosin olla YOLOn tapauksessa melko pieni tällaisille kohteille. YOLO V4 tiny-neuroverkolla saavutetaan parempi tunnistusnopeus, mutta tunnistusvarmuus ei ole niin hyvää tasoa kuin muilla neuroverkoilla.

Jetson Nanoa käytettäessä Mobilenet-SSD:tä voidaan pitää parempana vaihtoehtona tähän sovellukseen. Jetson Nanon suorituskyky riittää sujuvaan neuroverkon suorittamiseen samalla kun esitetään sujuvaa live-kuvaa. Jos taas käytettävissä on paremman suoritustehon omaava laitteisto, YOLO V4 voidaan olettaa toimivan paremmin.

4.2 Laadunvalvonta PoC tulokset

Toisessa PoC-testissä todettiin YOLO V4-neuroverkon olevan väärä tavoitellessa sovellusta, joka kykenisi tunnistamaan minkä tahansa kosmeettisen virheen. Tähän pyrittiin opettamalla neuroverkolle vain hyvien tuotteiden kuvia. YOLO V4 on optimoitu kohteiden tunnistukseen, jolloin se tunnistaa viallisetkin tuotteet hyviksi. Pieni vika tuotteessa ei riitä muuttamaan tuotteen ulkonäköä niin paljoa että sen tunnistaminen vaikeutuisi.

YOLO-neuroverkkoa käytettäessä laaduntarkastukseen on käytettävä eri lähestymistapaa ja opetettava neuroverkolle eri viat. Tämä tosin tarkoittaa sitä, että jos myöhemmin löydetään uusi vika, neuroverkko on opetettava uudelleen datalla, joka sisältää kuvia näistä vioista.

VGG16 saadut tulokset olivat lupaavia. Neuroverkko kykenee tunnistamaan viallisia tuotteita opettamalla sille vain hyviä tuotteita. Tunnistusvarmuus ei kuitenkaan ole mahdottoman hyvä, vaihdelleen 60 ja 90 prosentin välillä. Lisäksi osa vioista jää tunnistamatta, tai tulkitaan väärin. Opetusdatan määrää voidaan pitää melko pienenä mikä luultavasti selittää osaltaan keskinkertaista tunnistusvarmuutta. Testissä käytetyistä VGG16 voidaan suositella jatkokehityksen pohjaksi.

4.3 Tulosten arviointi

Tuloksia voidaan pitää kohtuullisen hyvinä. Opetuksessa käytettävien kuvien määrä on kuitenkin suhteellisen pieni verrattuna normaalisti tuhansia kuvia ja opetuskierroksia sisältäviin neuroverkkojen opetuksiin. Toki tämä tapahtuu opetettaessa neuroverkkoja kokonaan alusta alkaen, tässä työssä pystyimme hyödyntämään siirto-opetusta ja siten pääsimme parempiin tuloksiin pienemmillä kuvamäärillä.

Ensimmäisen PoC-testin osalta opetetut neuroverkot tunnistavat kappaleet missä tahansa asennossa. Kuvadataa on myös otettu eri kameroilla, jolloin eri laatuiset kuvat eivät aiheuta ongelmia. Toisessa PoC-testissä tilanne on hieman eri, sillä kuvaus on suunnattu tarkasti yhdestä suunnasta. Eri suunnista otetut kuvat eivät toimisi tässä tapauksessa. Eri kameralla otettu kuva vuorostaan toimii riittävän hyvin tunnistusta varten.

Jetson Nanon suorituskykyä voidaan kuitenkin pitää suhteellisen rajallisena neuroverkkojen vaatiman laskentatehon osalta. Se on sovelias laite neuroverkkojen opettelu- ja testauskäyttöön, mutta vakavampaan sovelluskäyttöön voidaan suositella tehokkaampaa laitteistoa. Laitteiston laskentateholla on suora vaikutus neuroverkon prosessointiaikaan ja sitä kautta tuloksen saamiseen.

5. PÄÄTELMÄT JA KEHITYSEHDOTUKSET

Tässä osiossa tarkastellaan mitä tuloksista voidaan päätellä sekä pohditaan tutkimuksessa suoritetulle työlle sopivia jatkokehitysehdotuksia.

5.1 Päätelmät

Avoimen lähdekoodin tekoälysovelluksissa on potentiaalia teollisuuden sovelluksiin. Kohteiden tunnistukseen kehitettyjä neuroverkkoja voidaan hyödyntää myös teollisuudessa. Erilaisia tarkastuksia ja laatupoikkeamien tarkistuksia voidaan suorittaa käyttämällä testeissä käytettyjä laitteistoa ja neuroverkkoja.

Avoimen lähdekoodin sovelluksien yksi suurista haasteista on hyvin laajan ohjelmakirjastojen hallinta. Ohjelmien ja kirjastojen yhteensopivuus on varmistettava huolellisesti. Ohjelmien päivitykset voivat aiheuttaa ennalta arvaamattomia muutoksia yhteensopivuuksiin, jolloin aikaisemmin toimiva ohjelma ei enää toimi oikein. Näiden selvitystyö vie aikaa ja resursseja, ja niihin on varauduttava riittävällä koulutuksella. Muutokset on varmistettava huolellisesti ennen käyttöönottoa.

Neuroverkon opetukseen käytettävän datan hankinta ja käsittely vie paljon aikaa ja resursseja. Kuvien määrä ja laatu vaikuttavat opetustulokseen. Neuroverkon opetuksessa käytetään tavallisesti tuhansia valokuvia. Suuri kuvien määrä varmistaa paremman neuroverkon oppimisen ja sitä kautta varmemman tunnistuksen erilaisissa tilanteissa.

Tutkimustyössä käytetty laitteisto on hankintahinnaltaan edullinen ja hyvin saatavilla. Laitteisto osoittautui toimivaksi neuroverkkoihin perustuvien oppivien konenäkösovellusten opettamiseen ja käyttämiseen. Jetson Nano on pienoistietokoneista parempi vaihtoehto kuin Raspberry Pi neuroverkkojen käyttöön. Grafiikkasuoritin takaa Jetsonille paremman suorituskyvyn monisäikeisen prosessoinnin ansiosta. Parhaiten neuroverkkojen opetukseen soveltuvat kuitenkin tehokkaat useilla grafiikkasuorittimilla varustetut tietokoneet. Näiden laskentatehoa on saatavilla myös pilvipalveluina, jolloin kalliita tietokoneita ei tarvitse hankkia itse. Nykyaikainen web-kameran kuvanlaatu on riittävän hyvä käytettäväksi laitteiston kamerana.

Ensimmäisessä PoC-testissä käytetyistä neuroverkoista Mobilenet-SSD-neuroverkko toimi luotettavasti ja sulavasti tunnistuskäytössä, kun taas YOLO V4-neuroverkon sujuvaan suoritukseen resurssit eivät ole riittävät. Kevyempi YOLO V4 tiny-neuroverkko toimi nopeammin, mutta tunnistusvarmuuden kustannuksella. Jos käytettävissä olisi paremmalla laskentateholla varustettu laitteisto, YOLO V4-neuroverkon toiminta olisi varmasti

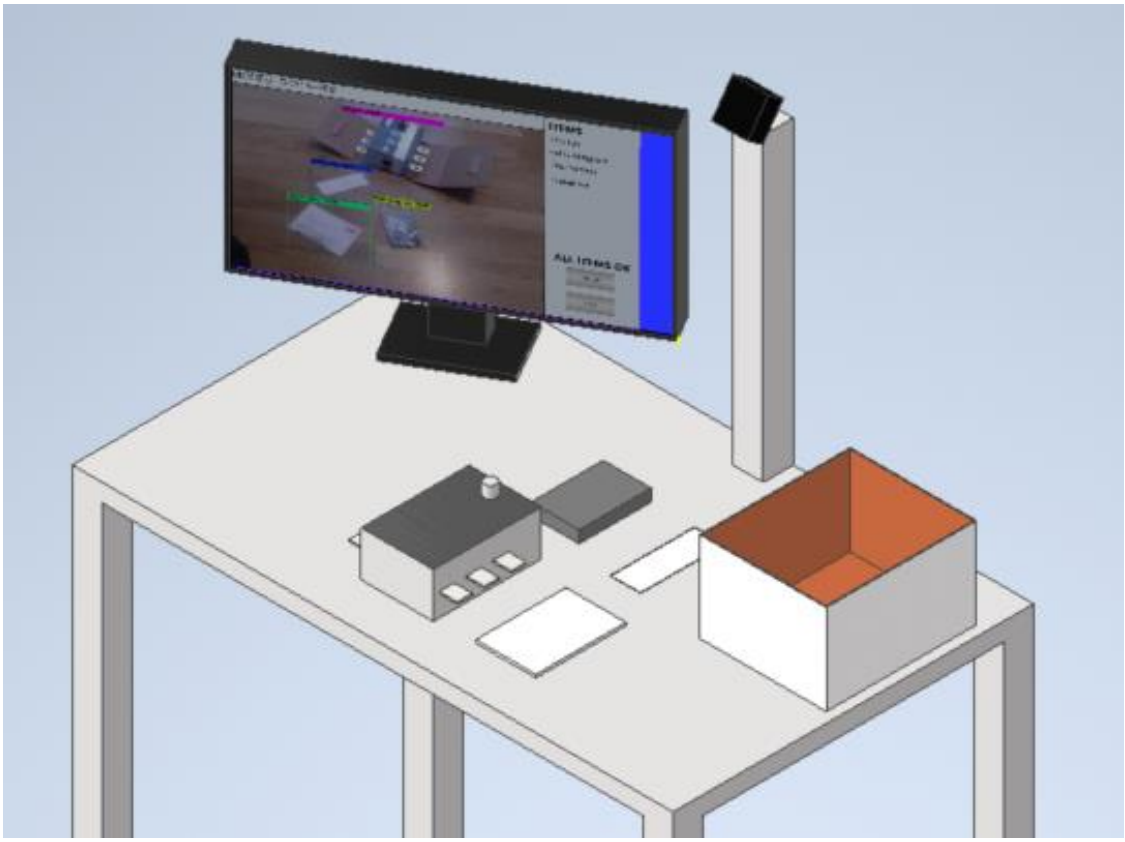
parempaa kuin nyt havaittu. Toisessa PoC-testissä käytetyistä neuroverkoista VGG16 toimii hyvin, ja on sovelias jatkokehityksen pohjaksi.

Tässä tutkimustyössä käytetyt neuroverkot ja ohjelmat eivät sovellu suoraan tuotantokäyttöön, vaan kehitystyötä tarvitaan erityisesti ohjelmien osalta. Käytetyt ohjelmat ovat sopivia neuroverkkojen kokeiluun ja opetteluun, mutta eivät sellaisenaan tarjoa esimerkiksi rajapintoja muihin sovelluksiin. Neuroverkot ovat vaativia tietokoneohjelmia, joiden kehitystyö vaatii sekä aikaa että osaamista. Tästä muodostuu niiden käytölle kustannuksia, jonka määrää on vaikea arvioida. Lisäksi ylläpidosta aiheutuu kustannuksia järjestelmän käytön aikana. Kehitystyö voidaan toteuttaa sisäisenä projektina tai hankkia ulkopuoliselta tekijältä.

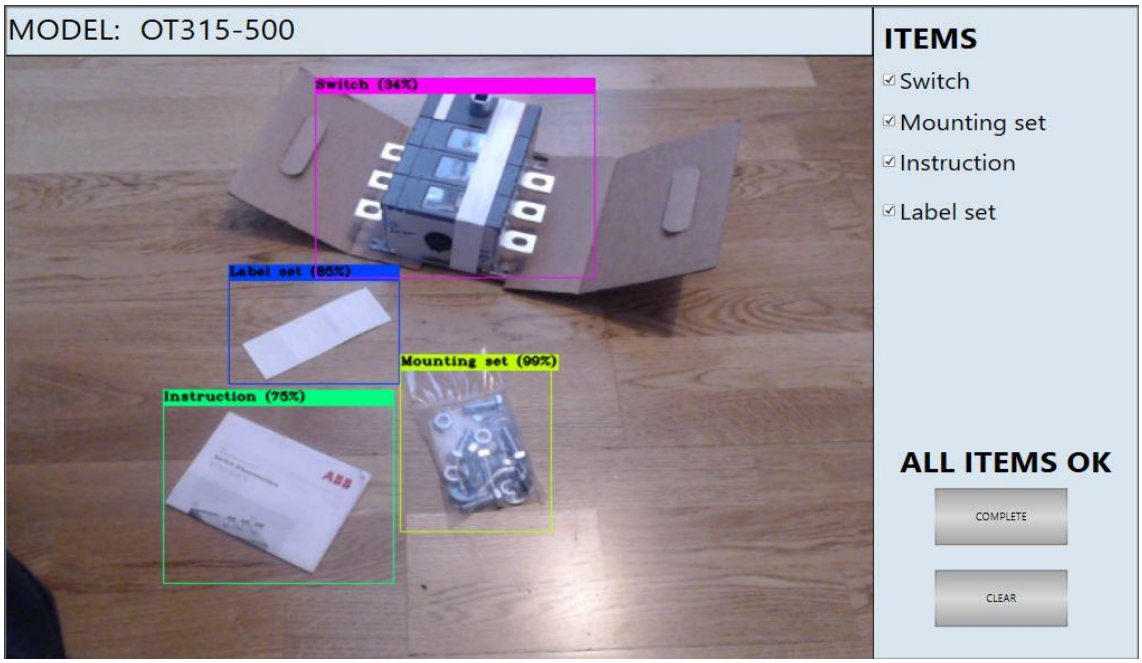
5.2 Kehitysehdotukset pakkauksen tarkistukselle

Tehdyn POC-testin perusteella voidaan todeta, että pakkauksen tarkistus on toteutettavissa testissä käytetyllä laitteistolla. Käytetyillä neuroverkoilla kyetään tunnistamaan kohteet luotettavasti. Tuotantokäyttöön tarkoitetun sovelluksen kehitykseen on kuitenkin syytä varata resursseja yhtiön sisältä tai ostaa työ kolmannelta osapuolelta. Käytetyt ohjelmat voivat toimia perustana ja mallina tälle sovellukselle.

Testin perusteella hahmoteltiin ehdotus, miten testauksen laitteistoa voitaisiin hyödyntää tuotannossa. Kuvaan 40 on luonnosteltu tuotepakkauksen työpisteen järjestely. Työpisteeseen ei tarvitse tehdä suuria muutoksia, kameran ja näytön sijoitus ovat käytännössä ainoat tarvittavat asiat. Kameran sijoittelussa on varmistettava riittävä kuva-ala sekä esteetön näkyvyys työskentelyalueelle.



Kuva 40. *Havainnekuva työpisteestä*



Kuva 41. *Käyttöliittymän hahmotelma*

Käyttöliittymä jossa työpisteen näkymää esitetään voisi olla kuvan 41 mukainen. Käyttöliittymän pääosassa on kameralta saatava videokuva, johon neuroverkko merkitsee havaitut kappaleet. Käyttöliittymän ylälaidasta voidaan valita tuote mitä ollaan pakkaamassa. Tämän valinnan mukaan esitetään pakkaukseen kuuluvat artikkelit käyttöliittymän oikeassa laidassa. Kun kuvassa tunnistetaan pakkaukseen kuuluva artikkeli, se merkitään kerätyksi oikean laidan listaan. Käyttöliittymän alalaidassa on painikkeet, joilla kuitataan pakkaus valmiiksi sekä löydettyjen artikkelien tyhjennys.

Käyttöliittymä on syytä pitää yksinkertaisena, jotta työntekijä pystyy keskittymään oleelliseen suorittaessaan tuotteiden pakkausta. Käyttöliittymään kannattaa tuoda vain työn suorittamiseen liittyvät tarpeelliset tiedot, kuten tässä tapauksessa havaitut kappaleet. Kameran kuvan osuus voisi olla pienempi, ja korostaa löydettyjä artikkeleita.

Valmistuneesta pakkauksesta voidaan lähettää saman käyttöliittymän kautta tieto tuotannonohjausjärjestelmään. Käyttöliittymään voisi lisäksi liittää tietoa tarvittavista pakkausmääristä, tuotantoerän valmistumisen etenemisestä ja muusta tarpeellisesta tiedosta.

Työn tuloksia esiteltiin ABB SP:n tuotannon työntekijöille palautetta varten. Saadun palautteen mukaan laitteistolle olisi hyvin paljon erilaisia käyttökohteita auttamaa päivittäisessä työskentelyssä. Tuotantolinjoilta löytyy paljon erilaisia kokoonpanon vaiheita, joiden onnistumista voitaisiin seurata konenäöllä. Osa työvaiheista suoritetaan käsin, jolloin kokoonpanoa voitaisiin seurata ja kirjata suoritetut vaiheet niiden valmistuttua. Eri-

tyisesti tästä olisi hyötyä, jos jokin komponentti tai työvaihe jää piiloon kokoonpanon edetessä eikä sitä kyetä enää tarkistamaan myöhemmin. Käyttöliittymässä voisi olla tarkistuslistatoiminto, joka kirjaa työvaiheen suoritetuksi, kun käyttäjä näyttää oikein kootun kappaleen kameralle.

5.3 Kehitysehdotukset laadun tarkastukselle

Laadun tarkastukselle on helppo löytää lukuisia käyttökohteita ABB:llä. Manuaalisesti tehtävässä tuotannossa käytetään ns. laatuportteja, mikä pitää sisällään tuotteille suoritetaan visuaalinen tarkastus. Konenäöllä suoritettava laaduntarkastus olisi hyvä tuki ihmisen suorittaman tarkistuksen rinnalle. Myös automaattisissa tuotantolinjoissa on lukuisia kohteita, joissa voidaan suorittaa osakokoonpanon tai tietyn työvaiheen laaduntarkistus.

POC-testissä käytettiin OT125-tuoteperheen tuotteita. Niiden tuotantolinjalla tuotteen lopputarkistus suoritetaan linjaston operaattoreiden toimesta. He tarkistavat linjastolta tulevat tuotteet silmämääräisesti ennen niiden siirtämistä varastointiin ja pakkaukseen. Käytettävissä oleva YUMI yhteistyörobotti soveltuisi ominaisuuksiltaan hyvin OT125-tuotantolinjan tuotteiden lopputarkastukseen. Robotin kameran kuvausvalo ei kuitenkaan ole optimaalinen heijastavan materiaalin kuvaukseen. Ulkoisella valonlähteellä kuvan heijastukset voitaisiin poistaa, esimerkiksi käyttämällä ympyrän muotoista valonlähdettä kameran ympärillä. Ilman ulkoista valonlähdettä ylivalottumista voidaan välttää asettamalla valon voimakkuutta pienemmäksi ja käyttämällä pidempää valotusaikaa.

Robotin ottama kuva voidaan tallentaa esimerkiksi lähiverkkoon jaettuun kansioon, johon myös neuroverkkoa suorittavalla tietokoneella on pääsy. Tähän kansioon tallennettu kuva voidaan ottaa neuroverkon tarkistettavaksi, ja tulos voidaan lähettää esimerkiksi OPC UA-palvelimen kautta takaisin robotille. OPC UA-palvelin on tietokoneella suoritettava ohjelmisto, joka voi kommunikoida useiden erilaisten teollisuudessa käytettyjen ohjauslaitteistojen kanssa.

Tuloksen mukaan robotti voi nostaa hyvän kappaleen kuljettimelle tai laatikkoon. Hylätyn tuloksen saanut kappale voidaan nostaa hylättyjen laatikkoon. Koska tuotantolinjoilla valmistettavilla kappaleilla on yksilöllinen sarjanumero, voidaan kappaleiden viat kohdistaa yksilöllisesti. Testissä käytetty VGG16 neuroverkko ei yksilöi vikaa sellaisenaan, mutta jatkokehityksellä voidaan rakentaa ohjelma, joka tähän kykenee. Vikojen esiintymisestä voidaan kerätä dataa. Tämän datan perusteella voidaan esimerkiksi päätelmiä laitteiston vikaantumisista ja huollon tarpeesta, jos samaa vikaa esiintyy useita kertoja lyhyellä aikavälillä.

Yhteistyörobotin turvaominaisuudet mahdollistavat sen sijoittamisen nykyiseen työpisteeseen ilman että työpistettä tarvitsee muokata merkittävästi. Robotti ei tarvitse ympärilleen erillisiä suojia. Robotin kaksi käsivartta mahdollistavat työskentelyn samoilla järjestelyillä kuin ihminen työskentelee nykyisessä työpisteessään. Työpisteeseen tarvitaan vain robotille sopiva alusta, esimerkiksi tukeva pöytä, sekä teline johon kappale voidaan laskea kuvauksen ajaksi.

5.4 Muut kehitysehdotukset laitteiston käytölle

Työssä tutkittiin myös muita käyttökohteita avoimen lähdekoodin neuroverkoilla toteutetulle konenäölle. Teollisuuden tuotantolinjoissa on hyvin monenlaisia kohteita mihin vastaavaa laitteistoa voidaan ajatella käytettäväksi. Seuraavana on esitelty joitakin näistä.

Työpisteiden työkalujen ja siisteyden tarkistus

Oikeiden työkalujen saatavuus työn suorittamiseen on merkittävässä osassa työn mielekkyyttä ja onnistumista. Tuotannon työpisteisiin on tyypillisesti varattu siinä työvaiheessa tarvittavat työkalut sekä materiaali. Työkaluilla on jonkinlainen säilytysratkaisu, tyypillisesti työkaluseinä erilaisilla kiinnikkeillä varustettuna. Jokaisella työkalulla on oma paikkansa. ABB SP:llä on käytössä 5S-menetelmä, jonka periaatteiden mukaisesti työpisteiden työkalut ovat järjestetty ja merkitty. Konenäköä voidaan käyttää tukemaan tätä tarkkailua. Alla olevassa kuvassa Kuva 42 on havainnekuva siitä, miten neuroverkolla voitaisiin tunnistaa työkalutaulusta työkalut. Vasemmalla olevassa kuvassa työkalut ovat paikoillaan, oikealla olevasta puuttuu yksi työkalu paikaltaan. Havainto on korostettu punaisella värillä.

Kuva 42. *Havainnekuva työkalutaulun tarkistuksesta*

Neuroverkko voidaan opettaa kuvilla, joissa työkalut ovat paikoillaan, joten opetuksen jälkeen neuroverkko tunnistaa kuvasta muuttuneet asiat. Ongelmaksi tosin voi muodostua työkalujen muuttuvat piirteet tai värit, jolloin niistä saattaa tulla virheellisesti puuttuvan työkalun havainto.

Nykyaikaisten mobiilirobottien tuoma mahdollisuus rakentaa liikkuva tarkastuslaitteisto voisi toimia kiertävänä työpisteiden tarkastajana, kun ihmiset ovat poissa työpisteiltä. Esimerkiksi tämä voisi tapahtua yöllä. Työpisteen puutteellisuudesta voidaan lähettää raportti työnjohtajalle. Puutteelliseen varustukseen voidaan reagoida ilman että työntekijät ilmoittavat puutteista.

Materiaalin varastojen seuranta

Kokoonpanolinjoilla kuluu jatkuvasti komponentteja, joiden oikea-aikainen saatavuus on kriittinen työskentelyn edellytys. Jos komponentit loppuvat, kokoonpano pysähtyy. Nykyaikaisessa tuotannonohjauksessa komponenttivarastoa täydennetään kulutuksen mukaan, mutta tämä seuranta on usein erilaisten laskureiden tai tapahtumien seurannan varassa. Edullinen kameralaitteisto mahdollistaisi komponenttien kulutuksen seurannan reaaliajassa. Alla olevassa kuvassa Kuva 43 vasemmanpuoleisessa tapauksessa komponentteja havaintaan riittävästi. Oikeanpuoleisessa tapauksessa komponentit ovat vähissä tai loppuneet kokonaan.

Kuva 43. *Havainnekuva komponenttien määrän tarkistuksesta*

Komponenttien vähäisestä määrästä voidaan tehdä ilmoitus logistiikkapuolelle, joka hoitaa paikalle tavaratäydennyksen. Ilmoitus voidaan myös esittää työpisteen työntekijöille heidän käyttämillään työpisteen tietokoneilla.

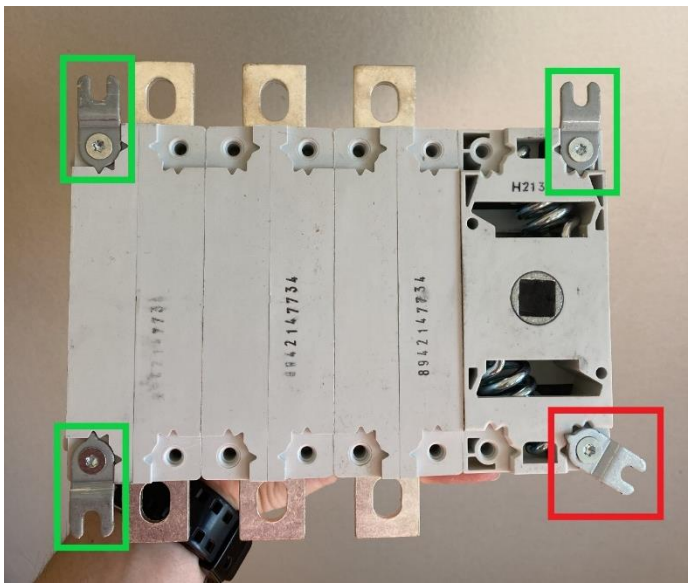
Materiaalivarastoja voidaan tarkkailla myös erilaisista hyllyistä tai lavoista. Tähänkin sovellukseen voidaan ajatella mobiilirobottia, joka suorittaisi liikkumisen varastossa ja kuvaisi kameralla eri varastot. Jatkokehittämällä laitteistoa ja neuroverkkoa voidaan materiaalien seurannasta löytää hyvin monta käyttökelpoista kohdetta.

Kokoonpanon vaiheiden tarkistus

Konenäköjärjestelmällä voidaan myös korvata perinteisiä antureita tai kamerajärjestelmiä automaattisissa kokoonpanolinjastoissa. Konenäköjärjestelmällä on joustava, ja samalla laitteistolla voidaan suorittaa useita erilaisia tarkistuksia.

Kuvassa Kuva 44 on havainnekuva, kuinka laitteistolla voidaan tarkistaa tuotteen kiinnitysjalkojen asennuksen onnistuminen. ABB SP:n kokoonpanosolussa robotti liikuttaa tuotteen eri kokoonpanovaiheiden välillä. Kiinnitysjalkojen automaattisen asennuksen jälkeen asennuksen onnistuminen tarkistetaan induktiivisella anturilla, joka tunnistaa onko jalan kiinnityskohdassa metallia. Tarkistus toimii muutoin hyvin, mutta joissakin tapauksissa kiinnitysruuvi ei kiristä jalkaa paikoilleen. Jalka jää roikkumaan ja saattaa olla oikeassa asennossakin, mutta kiinnitys ei ole hyväksyttävissä. Anturi havaitsee kuitenkin jalan ja laitteisto tulkitsee asennuksen onnistuneeksi. Myöhemmissä kokoonpanon vaiheissa väärin asennettu jalka voi aiheuttaa ongelmia.

Tämä tarkistus voidaan toteuttaa myös konenäön avulla. Neuroverkolla kuvasta etsitään jalat ja niiden asento. Kuvaus voidaan myös suorittaa eri suunnista. Esimerkiksi yksi kuva pohjasta, jolla havaitaan ovatko jalat paikoillaan ja oikeassa kulmassa. Seuraavat kuvat voidaan ottaa sivusta, jolloin nähdään, roikkuuko jalka huonosti onnistuneen ruuvauksen takia.



Kuva 44. *Havainnekuva tuotteen kiinnitysjalkojen tarkistuksesta*

6. YHTEENVETO

Tutkimustyölle asetettu tutkimuskysymys oli, kuinka avoimen lähdekoodin oppivaa konenäköjärjestelmää voidaan hyödyntää teollisuudessa. Työ keskittyi tutkimaan avoimen lähdekoodin neuroverkkoja ja niiden käyttöä konenäkösovelluksissa. Tutkimustyö on konstruktiivinen tutkimus, jonka menetelminä on käytetty havainnointia, dokumentointia, haastatteluita ja kyselyitä. Käytännön osuudessa menetelminä on käytetty soveltuvien neuroverkkojen kartoitusta, analysointia ja testausta.

Ensimmäisessä vaiheessa ABB Smart Powerin tuotannosta suoritettiin nykytila-analyysi, ja kartoitettiin käyttökohteita laitteistolle. Löydettyjä käyttökohteita olivat erilaiset tuotannon laadun valvontaan tai tuotantoprosessin edistämiseen liittyvät tarkastukset. Näistä käyttökohteista valittiin kaksi tarkempaan tarkasteluun ja Proof-of-Concept-testaukseen. Nämä olivat:

1. Tuotepakkauksen sisällön tarkistus
2. Tuotteen laadun tarkastus

Tuotepakkauksen sisällön tarkistuksessa haluttiin tunnistaa pakkaukseen tulevat komponentit. Tuotteen laadun tarkastuksessa vuorostaan etsitään tuotteesta viallisia kohtia tai puuttuvia osia. Molemmille PoC-testissä tutkitulle toiminnalle voidaan löytää useita käyttökohteita ABB SP:n tuotannosta, ja siksi niiden tutkiminen oli perusteltua.

Toisessa vaiheessa tutustuttiin aihetta käsittelevään kirjallisuuteen, teoriaan ja tutkimusjulkaisuihin. Näillä hankittiin pohjatietoa aihealueesta. Erilaisista internetissä julkaistuista artikkeleista ja ohjeista saatiin esimerkkiratkaisuja, joiden avulla soveltuvaa ratkaisua voitiin hakea.

Kolmannessa vaiheessa määritettiin tutkimusta varten soveltuva laitteisto sekä ohjelmistot. Laitteistona haluttiin käyttää edullista kuluttajakäyttöön tarkoitettua tietokonetta ja kameraa. Hankintahinnaltaan edulliselle laitteistolle on helpompi perustella käyttökohteita kuin kalliille. Lisäksi halvempia laitteistoja voidaan hankkia useampia samalla hinnalla. Käytetyksi tietokoneeksi valikoitui NVIDIA:n valmistama Jetson Nano pienoistietokone. Pienoistietokonetta käytettiin tutkimuksessa koska haluttiin tutkia tämänkaltaisten tietokoneiden soveltuvuutta tekoälysovellusten suorittamiseen. Edullinen hinta mahdollistaa laitteiston käytön myös kohteissa, joissa kallista laitteistoa ei ole perusteltua käyttää.

Neljännessä vaiheessa suoritettiin PoC-testit alussa määritettyihin käyttökohteisiin. Tuotepakkauksen sisällön tarkistusta varten pakkauksen komponentit kuvattiin. Kuviin merkittiin neuroverkon opetusta varten kuvissa esiintyvät komponentit. Ensimmäisen kokeillun neuroverkon, Mobilenet-SSD:n opetus suoritettiin Jetson Nanolla. Testauksessa neuroverkko tunnisti kaikki komponentit luotettavasti. Toinen kokeilu suoritettiin YOLO V4-neuroverkolla, jonka opetus suoritettiin Google Colab-pilvipalvelussa. Testaus suoritettiin Jetson Nanolla. Testauksessa kävi ilmi, että Jetson Nanon suorituskyky ei ole riittävä YOLO V4:n suorittamiseen live-tunnistuksessa. Neuroverkon tunnistuskyky oli kuitenkin hyvä. Kevyempi YOLO V4 tiny-neuroverkko toimi paremmin, mutta tunnistusvarmuus oli hieman heikompi. Testien perusteella voidaan todeta, että avoimen lähdekoodin neuroverkkoja voidaan hyödyntää tunnistussovelluksissa teollisuudessa. Jetson Nanon suorituskyky riittää joidenkin neuroverkoilla suoritettavien tunnistusten pyörittämiseen, mutta osa neuroverkoista on liian raskaita.

Laaduntarkastuksen PoC-testissä tutkittiin, miten neuroverkolla voidaan tunnistaa tuotteessa olevia vikoja. Tavoitteena oli saada aikaan tunnistussovellus, joka tunnistaa hyvästä tuotteesta poikkeavat viat ilman erillistä vikojen opetusta. Tuotteesta kerättiin kuvia vakioidulla kuvakulmalla ja valaistuksella. Neuroverkon opetukset suoritettiin Google Colab-pilvipalvelulla. Testissä kokeiltiin ensin YOLO V4-neuroverkon soveltuvuutta, mutta testin perusteella YOLO V4 ei toimi halutulla tavalla. YOLO V4 on optimoitu tunnistamaan opetettu kohteita niiden piirteiden avulla. Tuotteen pääpiirteet pysyvät koko ajan samana, jolloin pienet virheet eivät vaikuta tunnistukseen. Toinen tutkittu neuroverkko, VGG16 vuorostaan toimi halutulla tavalla. Neuroverkko opetettiin hyvillä kuvilla, ja se kykeni tunnistamaan eri virheet kohtuullisen luotettavasti. Tunnistusvarmuus virheille vaihteli 60 % ja 100 % välillä. Testin perusteella VGG16 neuroverkko on toimiva laaduntarkastuskäyttöön ja soveltuu jatkokehityksen pohjaksi.

Kokonaisuudessaan tutkimuksen perusteella voidaan todeta, että työssä tutkitulla laitteistolla ja neuroverkoilla on mahdollista toteuttaa teollisuuden konenäkösovelluksia. Tutkitut käyttökohteet ovat vain yksittäisiä esimerkkejä teollisen tuotannon eri alueista missä laitteistoa ja neuroverkkoja voidaan hyödyntää. Myös työssä käytetyt neuroverkot ovat vain pieni otanta olemassa olevista neuroverkoista. Neuroverkkojen kehitys on ollut viime aikoina nopeaa, erityisesti konenäön osalla. Uusia, parempia neuroverkkoja kehitetään jatkuvasti. Sovelluksen jatkokehitystä tehtäessä on suositeltavaa tarkastella mitä mahdollisuuksia näistä löytyy.

LÄHTEET

[1] J. Beyerer, F. Puente Léon, C. Frese, Machine Vision, Automated Visual Inspection: Theory, Practice and Applications, 2015. Saatavissa: https://www.springer.com/gp/book/9783662477939

[2] Työ- ja elinkeinoministeriö, Tekoäly 4.0 -ohjelma. Ensimmäinen väliraportti: käynnistysvaiheesta toteutusvaiheeseen. 2020. Viitattu 2.3.2021. Saatavissa: https://julkaisut.valtioneuvosto.fi/bitstream/handle/10024/163042/TEM_2021_29.pdf?sequence=4

[3] ABB Oy, Liiketoiminta Suomessa, Smart Power, viitattu 4.10.2021 Saatavissa: https://new.abb.com/fi/abb-lyhyesti/suomessa/liiketoiminnat/smart-power

[4] K. Lukka, Konstruktiivinen tutkimusote, viitattu. 1.12.2021. Saatavissa: https://metodix.fi/2014/05/19/lukka-konstruktiivinen-tutkimusote/

[5] ABB AB, IRB 14000 product specification, viitattu 1.11.2021. Saatavissa: https://search.abb.com/library/Download.aspx?DocumentID=3HAC052982-001&LanguageCode=en&DocumentPartId=&Action=Launch

[6] P. Hänninen, Robotiikka ja tekoäly, Tammertekniikka, 2022.

[7] M. Pietikäinen, O Silvén, Tekoälyn haasteet – koneoppimisesta ja konenäöstä tunnetekoälyyn. Oulun Yliopisto, 2019. Saatavissa: http://jultika.oulu.fi/Record/isbn978-952-62-2482-4

[8] Tesla Inc, Tekoäly ja Autopilot. Viitattu: 10.5.2022. Saatavissa: https://www.tesla.com/fi_FI/AI

[9] B. Staar, M. Freitag, M. Lütjen, Anomaly Detection with Convolutional Neural Networks for Industrial Surface Inspection, 2018. Saatavissa: https://www.researchgate.net/publication/327933317_Anomaly_Detection_with_Convolutional_Neural_Networks_for_Industrial_Surface_Inspection

[10] J. Walker, Machine Learning in Manufacturing – Present and Future Use-Cases. 2018. Viitattu 8.5.2022. Saatavissa: https://emerj.com/ai-sector-overviews/machine-learning-in-manufacturing/

[11] M. Lehto, P. Neittaanmäki, E. Niinimäki, R. Nyrhinen, A. Ojalainen, I. Pölönen, I. Rautiainen, T. Ruohonen, H. Tuominen, P. Vähäkainu, S. Äyrämö, S-M. Äyrämö, Tekoälyn perusteita ja sovelluksia. Jyväskylän Yliopisto, 2019. Viitattu 26.2.2022. Saatavissa: https://tim.jyu.fi/view/kurssit/tie/tiep1000/tekoalyn-sovellukset/kirja

[12] IBM, Neural Networks, viitattu 5.11.2021. Saatavissa: https://www.ibm.com/cloud/learn/neural-networks

[13] M. Nielsen, Neural Networks and Deep Learning, 2015. Saatavissa: http://neuralnetworksanddeeplearning.com/

[14] J. Patterson, A. Gibson: Deep Learning: A Practitioner's Approach, 2017.

[15] W. Pedrycz, S-M. Chen, Deep Learning: Concepts and Architectures. 2020. Saatavissa: https://link.springer.com/book/10.1007/978-3-030-31756-0

[16] S. Pan, Q. Yang, A Survey on Transfer Learning, 2010. Saatavissa: https://www.semanticscholar.org/paper/A-Survey-on-Transfer-Learning-Pan-Yang/a25fbcbbae1e8f79c4360d26aa11a3abf1a11972?p2df

[17] K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, 2015. Saatavissa: https://arxiv.org/pdf/1409.1556.pdf

[18] S. Tammina, Transferring learning usin VGG-16 with Deep Convolutional Neural Network for Classifying Images, 2019. Saatavissa: https://www.ijsrp.org/research-paper-1019/ijsrp-p9420.pdf

[19] M. Iitti, J. Grönman, J. Turunen, T. Lipping, Classification of Masonry Bricks Using Convolutional Neural Networks – a Case Study in a University-Industry Collaboration Project. Tampere University, 2021. Saatavissa: https://ieeexplore.ieee.org/document/9687077

[20] W. Liu, D. Anquelov, D. Erhan, C. Szegedy, S. Reed, C-Y. Fu, A. Berg, SSD: Single Shot MultiBox Detector, 2019. Saatavissa: https://arxiv.org/pdf/1512.02325.pdf

[21] Y. Li, H. Huang, Q. Xie, L. Yao, Research on a Surface Defect Detection Algorithm Based on Mobilenet-SSD, 2018. Saatavissa: https://www.mdpi.com/2076-3417/8/9/1678/htm

[22] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You Only Look Once: Unified, Real-Time Object Detection, 2016. Saatavissa: https://arxiv.org/pdf/1506.02640.pdf

[23] YOLOv4 / Scaled – YOLOv4 / YOLO – Neural Networks for Object Detection. Viitattu. 1.5.2022. Saatavissa: https://github.com/AlexeyAB/darknet

[24] J. Redmon, Darknet: Open Source Neural Networks in C. Viitattu 30.4.2022. Saatavissa: https://pjreddie.com/darknet/

[25] S. Anand, L. Priya, A Guide for Machine Vision in Quality Control, 2020. Saatavissa: https://www.taylorfrancis.com/books/mono/10.1201/9781003002826/guide-machine-vision-quality-control-sheila-anand-priya

[26] N. Sebe, I. Cohen, A. Garg, Machine Learning in Computer Vision, 2005. Saatavissa: https://link.springer.com/book/10.1007/1-4020-3275-7

[27] Cognex Corporation, Customer Stories. Viitattu 2.10.2022. Saatavissa: https://www.cognex.com/applications/customer-stories

[28] A. Tellaeche, I Maurtua, A. Ibarguren, Use of machine vision in collaborative robotics: An industrial case, 2016. Saatavissa: https://ieeexplore.ieee.org/document/7733689

[29] Red Hat, What is open source. Viitattu 28.4.2022. Saatavissa: https://www.redhat.com/en/topics/open-source/what-is-open-source

[30] D. Bretthauer, Open Source Software: A History. Viitattu 27.4.2022. Saatavissa: https://opencommons.uconn.edu/cgi/viewcontent.cgi?article=1009&context=libr_pubs

[31] Open Source Initiative, The Open Source Definition. Viitattu 28.4.2022. Saatavissa: https://opensource.org/docs/osd

[32] Free Software Foundation Inc., A Quick Guide to GPLv3. Viitattu 1.5.2022. Saatavissa: https://www.gnu.org/licenses/quick-guide-gplv3.html

[33] Free Software Foundation Inc., Various Licenses and Comments about Them. Viitattu 1.5.2022. Saatavissa: https://www.gnu.org/licenses/license-list.html

[34] Creative Commons, Licenses. Viitattu 3.10.2022. Saatavissa: https://creativecommons.org/licenses/by/4.0/

[35] RaspBerry Pi Foundation, About Us, Viitattu 12.9.2022 Saatavissa: https://www.raspberrypi.org/about/

[36] NVIDIA Corporation, Jetson Nano Developer Kit. Viitattu 7.10.2022. Saatavissa: https://developer.nvidia.com/embedded/jetson-nano-developer-kit

[37] A. Süzen, B. Duman, B. Sen, Benchmark Analysis of Jetson TX2, Jetson Nano and Raspberry PI using Deep-CNN. 2020. Saatavissa: https://ieeexplore.ieee.org/document/9152915

[38] NVIDIA Jetpack SDK, viitattu 10.10.2022. Saatavissa: https://developer.nvidia.com/embedded/jetpack

[39] NVIDIA Tensor RT, viitattu 24.4.2022 https://developer.nvidia.com/tensorrt

[40] Tzutalin, Labelimg. 2015. Viitattu 18.10.2022 Saatavissa: https://github.com/heartexlabs/labelImg

[41] NVIDIA Corporation, Hello AI World. Viitattu 21.10.2022. Saatavissa: https://github.com/dusty-nv/jetson-inference

[42] Google Inc., Mobilenet lähdekoodi, Saatavissa: https://github.com/tensorflow/models/tree/master/research/slim/nets/mobilenet

[43] J. Li, Z. Su, J. Geng, Y. Yin, Real-time Detection of Steel Strip Surface Defects Based on Improved YOLO Detection Network, 2018. Viitattu 21.10.2022. Saatavissa:https://www.sciencedirect.com/science/article/pii/S2405896318321001

[44] O. Chernytska, Explainable Defect Detection Using Convolutional Neural Networks: Case Study. Viitattu 19.10.2022. Saatavissa: https://towardsdatascience.com/explainable-defect-detection-using-convolutional-neural-networks-case-study-284e57337b59

[45] The MVTec Anomaly Detection Dataset, Viitattu 20.10.2022. Saatavissa: https://www.mvtec.com/company/research/datasets/mvtec-ad

[46] O. Chernytska, Visual inspection, Google Colab -työkirja. Viitattu 20.10.2022. Saatavissa: https://github.com/OlgaChernytska/Visual-Inspection/blob/main/Training.ipynb

[47] A. Rosebrock, Intersection over Union (IoU) for object detection. 2016. Viitattu 16.5.2022. Saatavissa: https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

LIITE A: KYSELYLOMAKE KÄYTTÖKOHTEISTA

Kysely kameralaitteiston käyttökohteista

Tämä kysely on osa diplomityöni taustaselvitystä. Diplomityöni aiheena on konenäkö ja tekoäly. Selvitän kyselyllä mahdollisia käyttökohteita mitä tiimissä työntekijät kokevat tarpeelliseksi.

Kysely toteutetaan anonyymisti, vain tiimin tunnus on tarpeen.

Kiitos vastauksestasi!

-Matti Jaakkola

Tiimi:

Mitä asioita tiimissäsi voitaisiin seurata tai tarkistaa kameralla? Esimerkkeinä kappaleiden laatu, kokoonpanon tarkistus, osien määrä laatikossa, työkalujen kunto.

Kirjoita alle vapaaseen muotoon:

LIITE B: KYSELYLOMAKE TARKASTUSKOHTEISTA

Kysely kameralla tehtävästä laadunvalvonnasta tiimissä

Tämä kysely on osa diplomityöni taustaselvitystä. Diplomityöni aiheena on konenäkö ja tekoäly. Selvitän kyselyllä mitä tuotteista on tarpeen tarkistaa laadunvalvonnan vuoksi.

Kysely toteutetaan anonyymisti, vain tiimin tunnus on tarpeen.

Kiitos vastauksestasi!

-Matti Jaakkola

Tiimi:

Listaa alla asioita mitä pitäisi kappaleesta tarkistaa:

1. Mikä:

   Sijainti kappaleessa:

2. Mikä:

   Sijainti kappaleessa:

3. Mikä:

   Sijainti kappaleessa:

4. Mikä:

   Sijainti kappaleessa:

5. Mikä:

   Sijainti kappaleessa:

Muut kommentit tai kehitysehdotukset: