Tampereen ammattikorkeakoulu

Henkilöauton korivaurion tunnistus YOLO-tekniikkaa käyttäen

Joni Ratavaara

OPINNÄYTETYÖ
Lokakuu 2022

Dataosaaminen ja tekoäly
Insinööri, yamk

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Dataosaaminen ja tekoäly, ylempi tutkinto-ohjelma
(Insinööri ylempi AMK)

RATAVAARA, JONI:
Henkilöauton korivaurion tunnistus YOLO-tekniikkaa käyttäen

Opinnäytetyö 75 sivua
Lokakuu 2022

Nykyisin yritykset keräävät paljon erilaista digitaalista dataa talteen toiminnoistaan ja tuotannostaan. Viime vuosien edistysaskeleet tekoälyn ja koneoppimisen alalla ovat tuoneet tämän datan hyödyntämiseen uudenlaisia työkaluja, joilla voidaan saada irti uudenlaisia hyötyjä. Niiden avulla yritysten prosesseja voidaan edelleen kehittää.

Tämän opinnäytetyön tarkoitus oli kouluttaa kapean tekoälyn kuvantunnistusmenetelmään pohjautuva, yksittäiset auton kolarivauriot kuvasta tunnistava koneoppimismalli, käyttäen apuna opinnäytetyön laatijan työpaikan arkistosta löytyvää kuvadataa. Tämänkaltaista automaattista vauriotunnistinta voisi käyttää apuna vauriokorjaamon vahinkotarkastusprosessissa tulevaisuudessa. Työssä käytettiin mallin opetusaineistona K-Auto Oy:n vauriokorjaamon vahinkotarkastuskuvia sekä AlexeyAB:n Darknet-arkkitehtuuria, joka hyödyntää YOLOv4-kuvantunnistusmenetelmää.

Malleja rakennettiin yhteensä kaksi kappaletta, ja ne koulutettiin tunnistamaan neljää eri vauriotyyppiä. Ensimmäinen malli toteutettiin tavallisella pöytätietokoneella ja toinen Tampereen ammattikorkeakoulun koneoppimiskuormien käsittelyyn rakennetulla tietokoneella. Molemmat mallit saatiin toimimaan, ja uusien vauriokuvien tunnistusta testattiin rohkaisevin tuloksin.

Työn tuloksena todettiin, että tämän kaltaisella vanhalla talteen jääneellä kuvadatalla on mahdollista kouluttaa toimiva, vauriot tunnistava malli. Samaa tekniikka hyödyntämällä onnistuisi tehdä laajempi ja yleisempi vauriotunnistin, joka koulutettaisiin kattavammalla opetusdatalla ja suuremmalla määrällä tunnistettavia luokkia.

Avainsanat: autot, vauriokorjaamo, vahinkotarkastus, koneoppiminen, neuroverkot, kuvantunnistus

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Master's Degree Programme in Data Expertise and Artificial Intelligence

RATAVAARA, JONI:
Vehicle Body Damage Detection Using YOLO Object Detection Method

Master's thesis 75 pages
October 2022

Nowadays many companies collect masses of different kinds of digital data from their processes. Resent progress in the field of artificial intelligence and machine learning has provided new tools to utilize this company data and can provide new kinds of benefits to improve company's processes further.

The objective of this thesis was to train an artificial intelligence -based object detection model that can recognize certain kinds of car body damages from car damage pictures. A larger scale damage detector could be used to help body shop's damage inspections in coming years. In this proof of concept, the training data was gathered from K-Auto Oy's body shop's damage inspection offices and utilized AlexeyAB's Darknet detection architecture with YOLOv4 object detection method.

Thesis includes two different damage detection models both of which were trained to detect four types of damages. The first model was trained with a desktop computer and the second model was trained with Tampere University of Applied Sciences computer built to handle machine learning workloads. Training of both models succeeded and detections were made with promising results.

Conclusion was that it is possible to use this kind of old data to train a viable damage detector. Using the same technique, it should be possible to manufacture a larger scale and more general damage detector, when it is trained with a considerably larger training data and with more damage categories.

Key words: car body shop, damage inspection, machine learning, object detection

SISÄLLYS

LYHENTEET JA TERMIT

ANN	Keinotekoinen neuroverkko (Artificial neural network)
Avg Loss	Koneoppimismallin keskimääräinen virhe
CABAS	Vauriokorjaamoilla käytettävä laskentajärjestelmä, jonka avulla voidaan laskea korjauskustannusarvio
CPU	Tietokoneen suoritin (Central Prosessing Unit)
CNN	Konvoluutioneuroverkko (Convolutional neural network)
Darknet	GitHubista löytyvä arkkitehtuuri kuvantunnistuksen toteuttamiseen YOLO-menetelmän avulla
GPU	Tietokoneen grafiikkaprosessori (Graphical Processing Unit)
IoU	Intersection over Union
mAP	Mean Average Precision
MS COCO	Microsoft Common Objects in Context -datasetti
YOLO	You Only Look Once – koneoppimisen kuvantunnistusmenetelmä

1 JOHDANTO

Nykyisin internetin, automaation ja digitalisaation aikana yritykset tuottavat sekä tahattomasti että tarkoituksellisesti toiminnoistaan ja järjestelmistään paljon eri muotoista digitaalista dataa. Nyt on aika ottaa tuota dataa käyttöön parantamaan liiketoiminnan ja palvelun tehokkuutta.

Maailman johtavat dataa käsittelevät yritykset ovat luoneet uuden standardin palvelujen nopeudelle, helppoudelle ja personoinnille. Tämän vuoksi nykyään kaikki muutkin yritykset pyrkivät yltämään palvelussaan tälle tasolle. (Kananen & Puolitaival 2019, 73) Tähän haastavaan tavoitteeseen tarvitaan avuksi data-osaamista ja tekoälyä. Mitä paremmin osaamme kerätä, käsitellä ja tulkita erilaisissa prosesseissa tuottamaamme dataa, sen tehokkaammin ymmärrämme ohjata ja ennustaa prosessin kulkua ja tuottavuutta. Tavallisten tietokoneiden laskentatehon kasvu sekä koneoppimistekniikoiden yleistyminen on tuonut tekoälysovellukset myös muiden kuin tietotekniikan alan yritysten ulottuville.

Tekoälyn mahdollistaman tehtävien automatisoinnin avulla erilaisten palveluiden vasteaika voidaan saada merkittävästi nopeammaksi lisäämättä työntekijöiden määrää. Monenlaiset toistuvat, arkiset työtehtävät ja usein kysytyt kysymykset täyttävät työpäiväämme, jotka tulevaisuudessa voidaan jättää ainakin osin tekoälyllä toteutetun automaation hoidettaviksi.

Tämän opinnäytetyön tarkoituksena on kouluttaa kokeellinen konvoluutioneuroverkkoon pohjautuva ohjatun oppimisen tekoälymalli havaitsemaan vauriokorjaamon autojen vahinkotarkastuskuvista neljän valitun vaurioluokan vaurioita käyttäen YOLO-tunnistusalgoritmia. Tutkimuskysymyksenä on, voidaanko yrityksen vanhaa kuvadataa käyttää toimivan vauriotunnistimen opettamiseen?

Olen työskennellyt 2011 vuodesta alkaen vauriokorjaamon työnjohdossa ensin Autotalo Laakkonen Oy:n vauriokorjaamolla ja myöhemmin 2019 Keskon yrityskauppojen jälkeen K-Auto Oy:n vauriokorjaamolla. Vauriokorjaamolla tehtävät autojen vahinkotarkastukset ovat kuuluneet toimenkuvaani päivittäin. Tämän työkokemuksen tarjoamasta näkökulmasta lähdin pohtimaan automatisoitua

vauriontunnistusta osaltaan avustamaan ja nopeuttamaan vahinkotarkastusprosessia.

2 VAHINKOTARKASTUS VAURIOKORJAAMOLLA

2.1 Vahinkotarkastus vaurion sattuessa

Kun autolle tapahtuu vahinko, josta syntyy vaurioita, ensimmäisenä korjaamolla tehdään vahinkotarkastus, jossa kattavasti dokumentoidaan valokuvaamalla sen kertaisen vahingon aiheuttamat vauriot autosta. Vahinkotarkastuksen tarkoitus on selvittää vaurion korjauksessa tarvittavat työvaiheet ja korjauksessa käytettävä varaosat sekä laskea mahdollisimman tarkka alustava korjauskustannusarvio. Samalla arvioidaan myös korjauksen ajallinen kesto, yleensä korjauspäivien tarkkuudella. Tämä on tarpeen, koska tällöin pystytään varaamaan asiakkaan autolle sopiva aika korimekaanikolle ja automaalarille kalenteriin vaurion korjausta varten ja aloittamaan korvaushakemusprosessi vakuutusyhtiön kanssa sekä tilaamaan korjauksessa tarvittavat varaosat. Myös tietenkin asiakasta kiinnostaa, kuinka kauan hänen autonsa viettää aikaa korjaamolla.

Vahinkotarkastuksen yhteydessä todennetaan asiakastiedot ja ohjeistetaan asiakasta tekemään vahinkoilmoitus vakuutusyhtiöön. Asiakkaan kanssa kolaritapahtuman kulku on yleensä hyvä käydä läpi, jotta tiedetään, onko kolarissa muita osapuolia ja kuka osapuolista on korvausvelvollinen eli kenen vakuutusyhtiölle korjauskustannusarvio korjaamolta lähetetään ja mikä vahinkolaji saattaa olla kyseessä.

Resurssien, logistiikan ja aikataulun hallitsemiseen tarvitaan huomattava määrä työntekijöiden aikaa ja suunnittelua avustavista tietokoneohjelmista huolimatta. Voidaan todeta, että mahdollisimman tarkka ja onnistunut korjauskustannuslaskelma hyödyttää korjausprosessin kaikkia osapuolia: asiakasta, korjaamoa ja vakuutusyhtiötä (Mätäsjärvi K. 2016).

2.2 Cabas-laskelman tekeminen

Vahinkotarkastuksessa K-Auto Oy:ssä käytetään ruotsalaista CAB Group AB:n tarjoamaa Cabas-ohjelmaa. Ohjelma tuntee kaikki Pohjoismaiden yleisimmät

henkilö- ja pakettiautomerkit ja mallit sekä niiden vauriokorjausohjeet, varaosat ja hinnat.

Korjauskustannusarvion laskeminen aloitetaan mallintamalla asiakkaan auto laskelmalle. Cabasin tietokannasta voidaan valita auton oikea merkki, malli, vuosimalli ja jopa varustelutaso. Autosta ja vauriosta otetut kuvat lisätään Cabas-laskelmalle, jolloin ne toimivat korjauskustannusarvion laskijan muistina vaurioituneista kohteista. Kuvassa 1 on esimerkki tyypillisestä korjaamon vauriokuvasta.

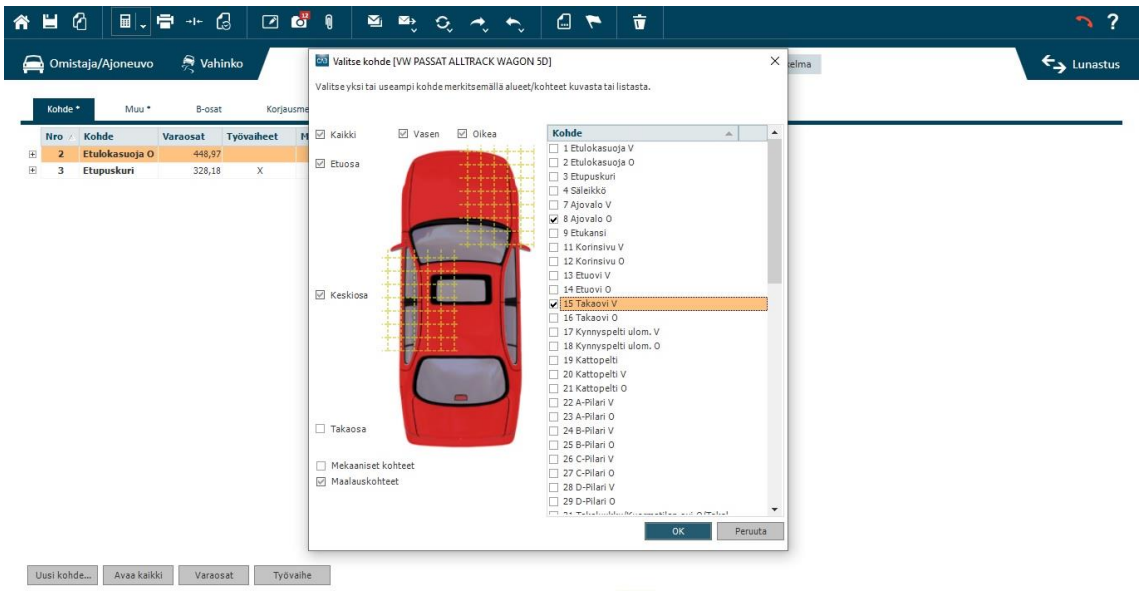
KUVA 1. Esimerkki vauriokuvasta (K-Auto Oy)

Kuvat otetaan niin sanotusti lähestyvästi eli ensin otetaan tunniste- ja yleiskuvat koko autosta, jossa näkyvät auton rekisterinumero ja korimallin muoto. Tämän jälkeen otetaan lähemmät ja yksityiskohtaisemmat kuvat vaurion sijainnista autossa ja sitten vielä tarkemmin itse vaurion laadusta. Samastakin vauriosta voidaan ottaa yksittäisessä tarkastuksessa useampi kuva eri suunnista, jotta saadaan mahdollisimman hyvä käsitys vauriosta pelkkien kuvien perusteella, koska vakuutusyhtiön puolella vauriotapaukset tulkitaan yleensä vain juuri näiden va-

hinkotarkastuskuvien tarjoaman tiedon pohjalta yhdistettynä vakuutuksenottajan tekemään vahinkoilmoitukseen, jossa on tarkka kuvaus sattuneesta vahingosta. Ammattitaitoinen vahinkotarkastaja osaakin ottaa kuvat niin, että tieto vaurioiden laadusta välittyy ymmärrettävästi kuvien avulla myös niille, jotka eivät ole päässeet vaurioita auton viereltä näkemään. Kuvia otetaan yksittäiseen vahinkoon liittyen muutamasta kuvasta jopa useiden kymmenien kuvien verran.

Auton pintojen puhtaus parantaa tarkastuksen laatua huomattavasti. Tämän vuoksi auto joudutaan usein pesemään, ainakin vaurion puolelta ennen kuvausta. Auton pintojen olisi silti hyvä olla pesun jälkeen kuivat kuvia ottaessa, jotta pinta ei ole vedestä kiiltävä ja ettei vesi esimerkiksi peitä pinnan naarmuja.

Cabasilla vaurioituneet kohteet valitaan ohjelmassa olevien osien luettelosta. Luetteloa voi rajata vauriokohdan mukaan ohjelmalla olevasta auton kohdekartasta (kuva 2), jos halutaan esimerkiksi vain auton oikean etukulman varaosat. Apuna on myös hiirellä 3D-näkymässä pyöriteltävät räjäytyskuvat, jotka voi valita osakokonaisuuksista.



KUVA 2. Cabas-näkymä ja kohdekartta autosta (K-Auto Oy)

Pelkästään vaurioituneiden kohteiden valinta ei riitä vaan kohteille pitää valita myös oikeat toimenpiteet vaurion laadun mukaan. Esimerkiksi jos oikean etulokasuojan lommo on korjattavissa, valitaan ohjelmasta tälle osalle sopivan ko-

koinen oikaisutyö, joka merkitään lommon pinta-alan mukaan neliödesimetrin tarkkuudella. Vahinkotarkastuskuvia ottaessa voidaan käyttää apuna auton pintaan kiinnitettäviä magneettimittanauhoja, jolloin lommon koko saadaan valokuvaan dokumentoitua puolueettomasti.

Vaihtoehtona vanhan osan korjaukselle on koko osan vaihto uuteen. Oikea vaihtoehto kussakin tapauksessa punnitaan auton merkkikohtaisten vauriokorjausohjeiden sekä kustannusten mukaan. Jos pellin oikaisutyön hinta ylittää uuden varaosan hinnan päädytään silloin edullisempana vaihtoehtona uuteen varaosaan. Myös maalauksen työvaiheisiin ja hintaan vaikuttaa käytetäänkö korjauksessa vanhaa vai uutta lokasuojaa.

Korjauskustannusarvion tärkein tehtävä on välittää dokumentointi vaurioista ja korjauskustannusarvio asiakkaan vakuutusyhtiölle, mikäli korvausta haetaan sieltä. Kun vakuutusyhtiö vahvistaa korvattavat työt, toimii sama korjauskustannusarvio myös työohjeena korimekaanikolle ja automaalarille. Onnistunut vahinkotarkastus on vauriokorjaamon prosessin tärkein vaihe yksittäisen korjauksen onnistumisen kannalta.

Vahinkotarkastus valokuvaus, kustannusarvion laskeminen ja korjausajan varaus tehtynä vaatii vauriotapauksen kokoluokasta riippuen aikaa omalla kokemuksella noin 20 minuutista jopa tuntiin. Tarkastuksen tekeminen on tarkkaa ja keskittymistä vaativaa työtä, joten laadun kannalta on hyvä, että se saataisiin kerralla keskeytyksittä tehtyä loppuun saakka, jotta korjaussuunnitelmaa ei tarvitsisi muuttaa enää kesken korjauksen, vahinkotarkastuksessa tapahtuneiden ajatusvirheiden takia. Mikäli suurempia muutoksia tulee tarkoittaa se yleensä korjauksen viivästymistä päivillä.

On huomioitava, että kaikkia vaurioita ei pysty autosta päältä päin näkemään ilman auton purkamista ja tämän vuoksi vahinkotarkastuksessa syntyvä alkuperäinen korjauskustannusarvio ei ole lopullinen, vaan lisäyksiä on monesti tehtävä korjauksen edetessä. Osien purkamisia ei tehdä yleensä vahinkotarkastuksessa, koska selvästi suurin osa vaurioituneista autoista tulee vahinkotarkastukseen pienemmän vaurion takia ja näin ollen liikennekelpoisena. Näissä tapauksissa auto menee vahinkotarkastuksen jälkeen takaisin asiakkaalle liiken-

nekäyttöön odottamaan varattua korjausaikaa ja tilattuja varaosia. Purettuja vaurioituneita osia ei myöskään pystyttäisi yleensä kiinnittämään takaisin autoon luotettavasti.

2.3 Tekoälyä vahinkotarkastuksessa?

Jos vahinkotarkastuksessa tarvittavaa työntekijän aikaa saataisiin pienemmäksi, vapauttaisi se aikaa muulle työlle huomattavasti. Vahinkotarkastuksessa valokuvauksesta korjausajan varaukseen asti on mielestäni monia toistuvia työvaiheita, joita voitaisiin hoitaa tekoälyn avulla joko kokonaan tai osaksi. Näkisin, että täysi vahinkotarkastuksen automatisointi vaatisi useita rinnakkain toimivia tekoälysovelluksia ja paljon erityyppistä dataa niiden kouluttamiseen. Tässä voisi olla alan vanhalla toimijalla etu, mikäli yrityksen vanhaa dataa pystytään käyttämään näiden tekoälysovellusten kouluttamiseen. Käyttökelpoinen historiadata tulisi siis todella arvokkaaksi. Sovelluksia voisi myös jatkokouluttaa vahvistetun oppimisen metodilla työn teon rinnalla, jossa se oppisi aina lisää käsitellyistä tapauksista ja tuloksena koko ajan tarkempi koneoppimismalli. Tällaisista malleista tulisi myös arvokasta omaisuutta yritykselle, jota ulkopuoliset tahot voisivat haluta ostaa. Vaikka tekoälymalli ei olisi täydellinen kaikissa vaurioissa, niin se silti nopeuttaisi huomattavasti palveluprosessia. Ihminenkään ei ole ikinä täydellinen, vaikka hän käyttäisi samaan työhön moninkertaisesti aikaa.

Ideaalitilanne voisi tulevaisuudessa olla sellainen, jossa vaurioita tunnistava malli integroidaan älypuhelinsovellukseen, jota asiakas voi käyttää kotipihassaan tai jopa jo kolaripaikalla. Sovellus voisi esimerkiksi puhelimen näytön kameraruudun kautta ohjata asiakasta lisätyn todellisuuden elementein ja opastaa käyttäjää ottamaan vauriokuvat tai jopa videokuvaa oikealla tavalla autosta. Ensimmäinen algoritmi tunnistaisi vauriot ja identifioisi auton rekisterinumeron perusteella. Toinen algoritmi laskisi korjauskustannukset ja korjauksen ajallisen keston. Kolmas algoritmi etsisi käyttäjän sijainnin perusteella lähimmät sopivat korjaamot ja neljäs näyttäisi kyseiselle korjaukselle sopivat vapaat korjausajat, joista käyttäjä voisi valita itselle parhaan vaihtoehdon. Sovellus voisi ottaa myös vastaan asiakkaalta vahinkoilmoituksen vakuutusyhtiölle ja lähettää valmiin korjauskustannuslaskelman kuvineen suoraan vakuutusyhtiölle eli täydellinen kor-

vaushakemus olisi valmis jo vaikka kolaripaikalla. Edelleen vaikeammat vauriot ja hinausta vaativat ajokelvottomat vaurioautot käsiteltäisiin todennäköisesti vasta korjaamoilla, mutta todella suuri osa korjaamon työstä ja asiakkaan vaivasta jäisi tämän automatiikan myötä pois, verrattuna nykyiseen prosessiin.

Tässä opinnäytetyössä kokeilen edellä mainittua ideaa autoliikkeelle kertyneen vanhan kuvadatan hyödyntämisestä vauriotunnistusmallin rakentamisessa. Toimiva vauriotunnistin olisi ensimmäinen osa vahinkotarkastuksen kattavampaa automatisointia varten. Tutkimuskysymyksenä on, voidaanko vanhaa kuvadataa käyttää toimivan vaurioita tunnistavan koneoppimismallin opettamiseen? Opinnäytteessä tehdään niin sanottu soveltuvuusselvitys (proof of consept).

3 DATA

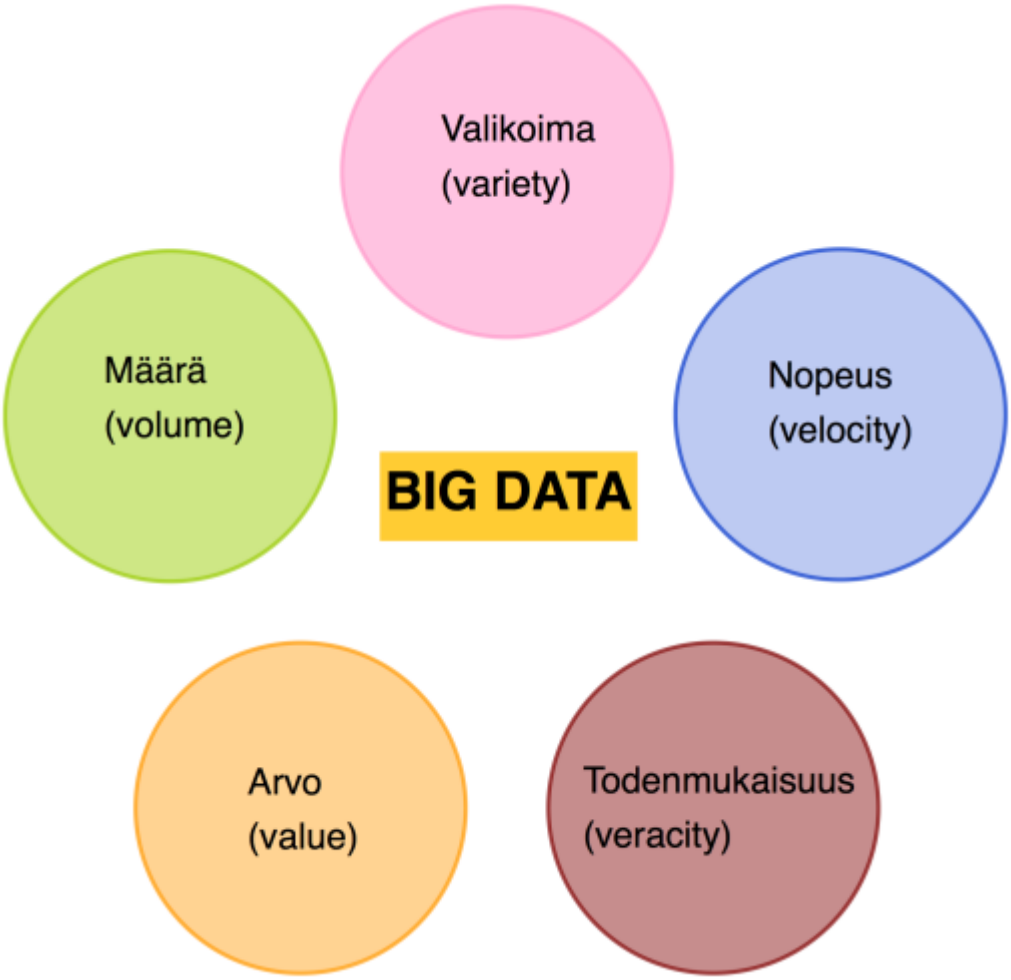
3.1 Mitä on data ja miten siitä saadaan tietoa?

Dataa voi olla monessa muodossa. Dataa ovat numerot, kirjaimet, teksti, kuvat ja jopa videot. Dataa voidaan kerätä useista eri lähteistä. Digitaalinen data voi olla eri formaatteihin tallennettua, kuten esimerkiksi .pdf, .txt, .csv, .wav, .jpg tai .mp4. Dataa pystytään hyödyntämään parhaiten, kun data on hyvälaatuista. Hyvälaatuinen data on yksiselitteistä ja yhdenmukaista. Kun dataan lisätään merkitys, saadaan informaatiota ja kun informaatiota tulkitaan, saadaan alkuperäisestä triviaalista datasta arvokasta ja käyttökelpoista tietoa, jolla lopulta pystytään tekemään päätöksiä. Datana voi olla esimerkiksi numero 100, merkitys voi olla, että numero 100 on kappalemäärä ja tietona, että kuukauden valmistusmäärä on 100 kappaletta. Tätä tietoa voidaan verrata vaikkapa edellisen kuukauden valmistusmäärään ja tehdä johtopäätöksiä, mihin suuntaan ollaan yrityksen valmistusmäärissä menossa. (Kananen & Puolitaival 2019, 71–72)

Datan analysointi on nykyisin bisneksessä tärkeää. Dataa voidaan visualisoida ja eri tietokantoja yhdistellä uusilla työkaluilla, jotta saadaan ymmärrys uusista merkityksistä ja asiayhteyksistä. Analysointi voi paljastaa datasta myös tärkeitä omien oletusten vastaisia faktoja, jotka eivät pelkällä päättelyllä olisi olleet raakadatasta havaittavissa. Mitä tarkemmin tunnetaan yrityksen prosessi tai vaikka asiakkaan tarpeet datan avulla, sitä paremmin pystytään kehittämään tuotantoa tai palvelemaan asiakasta. Kaikki data ei välttämättä tarvitse olla itse tuotettua, vaan omaa dataa voidaan yhdistää esimerkiksi julkiseen avoimeen dataan tai tarvittavaa dataa voidaan ostaa ulkopuolelta. Kun dataa on paljon ja pitkältä ajalta, sekä sitä osataan käsitellä ja tulkita, voidaan esimerkiksi asiakkaan käyttäytymistä tai tarpeita jopa ennakoida. Datan hyödyntäminen voi avata uusia mahdollisuuksia ja liiketoimintaideoita. (Kananen & Puolitaival 2019, 73–74)

3.2 Big data

Big data tai joskus myös massadata tarkoittaa massiivista ja jatkuvasti kasvavaa datamäärää. Itsessään tämä käsite ei määritä datatyyppiä eli kaikenlainen data voi olla big dataa. Määrittävänä asiana kuitenkin on, että datamäärä on niin suuri, että sitä on perinteisin menetelmin joko liian hidasta ja vaikeaa tai jopa kokonaan mahdotonta hallita ja hyödyntää. Big data jaetaan viiteen eri käsitteeseen (kuvio 1), jotka ovat englanniksi 5 V:tä: volume, variety, velocity, value ja veracity. Suomennettuna: määrä, valikoima, nopeus, arvo ja todenmukaisuus. (Tuominen & Neittaanmäki 2019, 5)



KUVIO 1. Big datan tunnusmerkit (Tuominen & Neittaanmäki 2019, 5)

4 TEKOÄLY

4.1 Mitä on tekoäly?

Elämme uusinta tekoälyn hype-aikaa sanovat Pietikäinen ja Silvén (2019, 5). Puhutaan uudesta teollisesta vallankumouksesta, jonka tekoäly ja koneoppiminen luovat. Tekoäly eli tietokonepohjaiset oppivat järjestelmät tulevat muun muassa työelämään ihmisten avuksi ja tueksi, mutta myös paljolti korvaamaan ihmisen työpanosta. Tekoälyllä voi tunnistaa nopeasti kohteita ja ennustaa tarkasti tuloksia tarvittaessa täysin automaattisesti. Sitä on jo nyt ympärillämme päivittäin. Esimerkkejä käytännön tekoälystä ovat kohdennettu mainonta, internetin suosittelevat hakukoneet, autonomista ajoa tavoittelevat autot sekä puheen- ja kasvojen tunnistus. (Pietikäinen & Silvén 2019, 5)

Tekoälyllä (engl. artificial intelligence) ei ole yleistä määritelmää painottavat Tuominen ja Neittaanmäki (2019). Esimerkiksi Euroopan parlamentti (2021) määrittelee tekoälyn olevan koneen kykyä osata perinteisesti ihmisälyyn yhdistettyjä taitoja, kuten suunnittelemista, oppimista, päättelyä tai luomista. Tekoäly on nykyisin myös tieteenalan nimi (Elements of AI 2022).

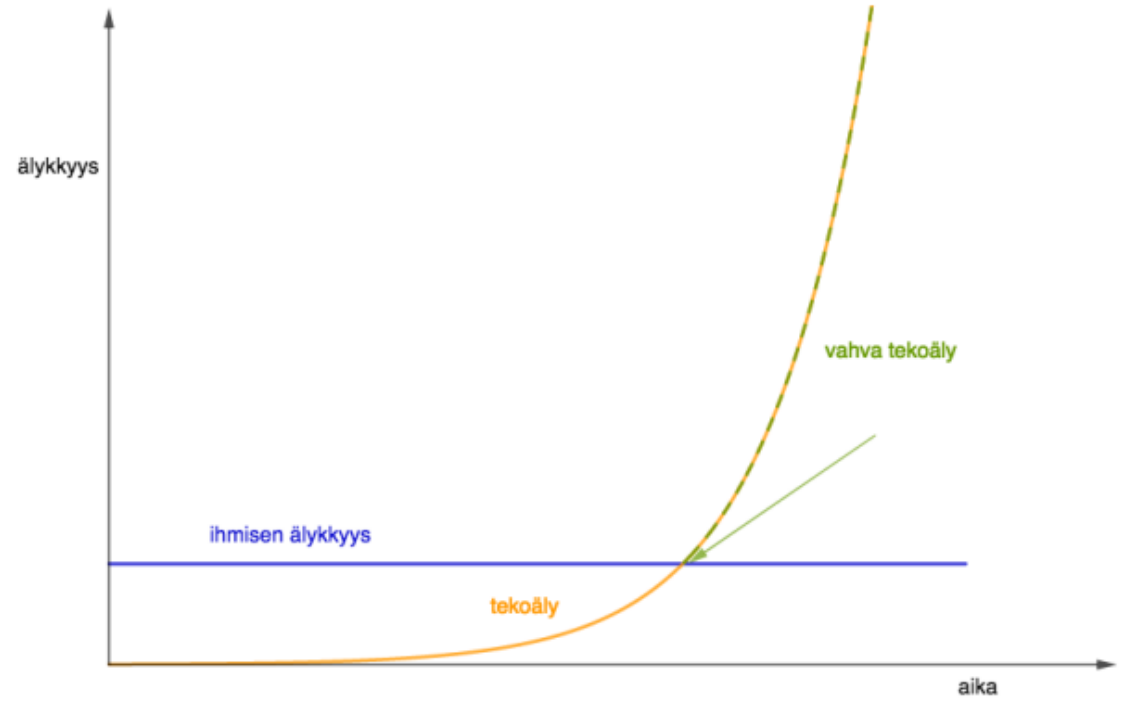
Tekoäly on pelkistettynä tilastotiedettä, matematiikkaa ja ohjelmointia. Se ei ole vain yksi teknologia, vaan rakentuu aina useasta eri menetelmästä ja tekniikasta, jotka valitaan ratkaistavan ongelman ja käyttökohteen mukaan. (Kananen & Puolitaival 2019, 27)

Tekoälyä voi soveltaa käytännössä mille tahansa alalle ja mihin tahansa tarpeeseen. Se mahdollistaa yhteiskuntien tehokkaamman resurssien käytön esimerkiksi energiatehokkuuden saralla sekä tehostaa ihmisten turvallisuutta, terveyttä ja oppimista (Pietikäinen & Silvén 2019, 5). Tekoälystä voi olla suurta apua jatkossa myös yksittäisten ihmisten päivittäisen elämän laadun parantamisessa. Se voi auttaa yksin kotona asuvia heikkokuntoisia ikääntyneitä pärjäämään arjessa, antaa kyvyn yksilölle seurata tarkasti terveyttään reaaliajassa, mahdollistaa virtuaalimatkailua, tehdä ihmisten yhteydenpitoa vapaammaksi kielirajojen ja välimatkojen rajoituksista, helpottaa tiedonhakua sekä ostosten

tekoa, vapauttaa kodin siivouksen taakasta ja auttaa vammautuneita liikkumaan ja saamaan aistejaan takaisin. (Pietikäinen & Silvén 2019, 220)

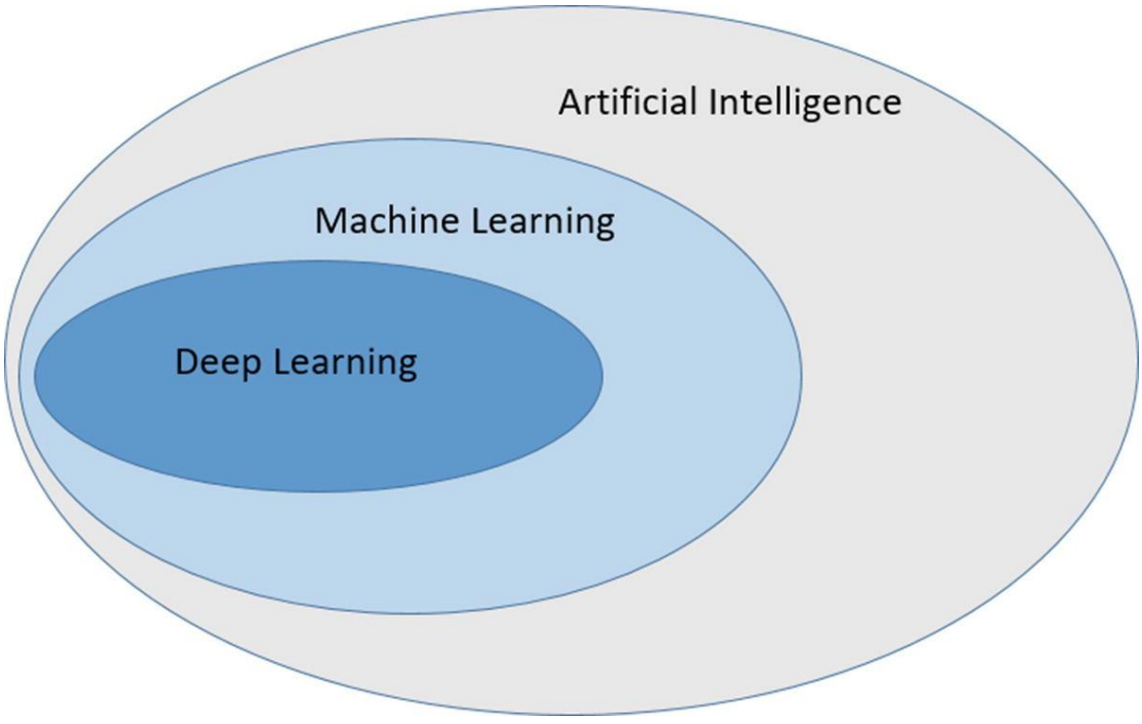
Tekoälyn kehittäminen ei riipu myöskään siitä, mitä teknologiaa käyttää. Yksittäisen tekoälysovelluksen koulutuksen idea voidaan demonstroida yhtälöillä, joita voidaan käsitellä millä tahansa kirjoitusvälineellä, kuten ihan jopa yksinkertaisesti kynällä ja paperilla. Kuitenkin kun tekoälyä aletaan kouluttamaan, vaatii se todella suuren määrän laskentakierroksia, jolloin on käytännöllisempää käyttää tietokonetta. (Kananen & Puolitaival 2019, 29)

Käsitteenä tekoäly voidaan jakaa yleiseen ja kapeaan tekoälyyn (engl. general AI ja narrow AI). Yleinen tekoäly on itsensä tiedostava ihmisälykkyyden tasoinen tai sen jopa reilusti ylittävä keinotekoinen itsestään tietoinen mieli, joka osaa ratkaista minkä tahansa sille annetun ratkaistavissa olevan ongelman aiheesta riippumatta. Kapealla tekoälyllä tarkoitetaan näennäisesti älykkäitä toimintoja toteuttavia järjestelmiä, jotka soveltuvat vain yhden tyyppisen ongelman ratkaisuun työkalun tavoin, eivätkä osaa käsitellä muun tyyppisiä ongelmia, saati tiedosta itseään. Sama jako voidaan tehdä myös termeillä vahva ja heikko tekoäly. Kuviossa 2 suhteutetaan ihmisen älykkyyttä vahvan tekoälyn käsitteeseen eli tulevaisuudessa koneiden vahva tekoäly ohittaisi ihmisen älykkyyden. Käytännössä nykyisin kuitenkin kaikki tekoälysovellukset ovat kapeaa tekoälyä ja yleinen tekoäly kuuluu toistaiseksi tieteiskirjallisuuteen. (Elements of AI 2022.)



KUVIO 2. Vahva tekoäly ylittää ihmisen älykkyyden (Tuominen & Neittaanmäki 2019, 9)

Kuviossa 3 havainnollistetaan tekoälyn suhde koneoppimiseen ja syväoppimiseen. Syväoppiminen on koneoppimisen osa-alue, joka puolestaan kuuluu suurempaan tekoälyn kenttään.



KUVIO 3: Tekoälyn suhde koneoppimiseen ja syväoppimiseen (Nelli F. 2018)

Tekoälyksi nimitettyjä järjestelmiä on ollut aiemminkin, mutta nykyinen tekoälyn aalto perustuu pitkälti neuroverkkoihin ja syväoppiviin algoritmeihin, jotka ovat entistä etevämpiä esimerkiksi tunnistamaan kohteita, luokittelemaan kuvia ja tunnistamaan puhetta. Tekoälystä povataan yhtä suurta tai jopa suurempaa muutosta kuin höyrykoneesta, sähköstä tai internetistä. Suurella muutoksella viitataan muun muassa teollisen tuottavuuden kasvuun ja sitä kautta hallitsevan tekoälysuurvallan taloudellisen ja poliittisen vaikutusvallan kasvuun. Tekoälyn johtavat yritykset ovat toistaiseksi olleet yhdysvaltalaisia, mutta toisena tulevan Kiinan odotetaan kirivän ykköspaikalle lähivuosina. Yhdysvaltalaiset tekoälyjätit, kuten Google, Microsoft, Meta (Facebook) ja Amazon ovat saaneet kiinalaiset haastajat nimeltä Baidu, Alibaba ja Tencent. (Varho 2020)

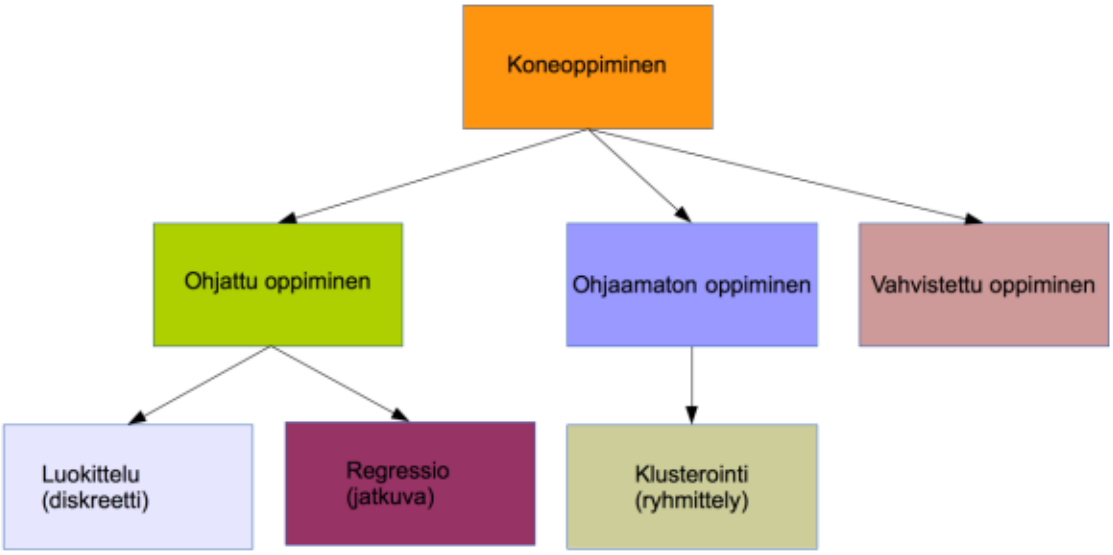
4.2 Koneoppiminen

Koneoppiminen (engl. machine learning) on tekoälyn osa-alue, joka tarkoittaa koneen tai ohjelman kykyä oppia toimimaan tehokkaammin käyttämällä hyväksi esimerkkidataa ja siinä toistuvia tapahtumia. Koneoppimisessa käytetään monimutkaisia algoritmeja käsittelemään dataa tehokkaasti ilman ihmisen jatkuvaa apua. Tämä erottaa sen perinteisestä raja-arvoihin ja sääntöihin pohjautuvasta ohjelmoinnista. Tavoitteena on saada kone havaitsemaan ja tulkitsemaan tietoa automaattisesti ja tehokkaasti. (Tuominen & Neittaanmäki 2019, 6) Tärkeää on myös todeta, että koneoppimisen käsitteellä ei voi korvata tekoälyn käsitettä, koska kokonaisvaltaiseen älyyn sisältyy muutakin kuin vain oppiminen (Pietikäinen & Silvén 2019, 17).

Yksi syy koneoppimisen viime vuosien suosioon on digitalisaation aiheuttama datatulva. Dataa on yrityksillä ja yhteiskunnalla tarjolla enemmän kuin sille on käyttöä (Pietikäinen & Silvén 2019, 68). IBM kertoi jo vuonna 2017 vuosiraportissaan, että 90 prosenttia kaikesta ihmiskunnan koskaan tuottamasta datasta on luotu raporttia edeltäneen kahden vuoden aikana (IMB 2017). Datan määrä on suuressa roolissa eli mitä suuremmalla datamäärällä esimerkiksi neuroverkkoon perustuva koneoppimismalli koulutetaan, sitä tarkempia ja laadukkaampia vastauksia se antaa (Kananen & Puolitaival 2019, 154).

Koska ihminen pystyy visualisoimaan vain kahden tai korkeintaan kolmen muuttujan ongelmia on koneoppiminen apuna ihmiselle vaikeampien ongelmien ymmärtämisessä ja hallitsemisessa. Uudet helpottavat työkalut mahdollistavat myös melkein kenen tahansa ratkaista vaikeita moniulotteisia ongelmia. (Pietikäinen & Silvén 2019, 68)

Koneoppiminen jaetaan kolmeen eri alalajiin (kuvio 4): ohjattuun oppimiseen, ohjaamattomaan oppimiseen sekä vahvistusoppimiseen. Näillä metodeilla kone oppii, joko ennustamaan, tunnistamaan tai ryhmittelemään dataa. (Tuominen & Neittaanmäki 2019, 12)



KUVIO 4: Koneoppimisen osa-alueet (Tuominen & Neittaanmäki 2019)

4.2.1 Ohjattu oppiminen

Ohjatussa koneoppimisessa (engl. supervised learning) ihminen opastaa konetta antamalla koneelle esimerkkeinä toimivan opetusdatan, jossa jokaisen esimerkin kohdalle on merkitty oikea vastaus. Pyrkimyksenä on tuottaa esimerkkidatan ”ohjauksen” avulla malli, joka tunnistaa automaattisesti ja itsenäisesti myös uudesta samankaltaisesta datasta oikeat vastaukset. (Elements of AI 2022)

Ohjattu oppiminen voidaan jakaa edelleen opetusdatan luonteen perusteella kahteen alaluokkaan: regressioon ja luokitteluun. Regressiossa käsitellään jatkuvaa dataa ja luokittelussa niin sanottua diskreettiä eli ryhmiin jaettavaa dataa. (Tuominen & Neittaanmäki 2019, 13)

Regressiolla tehdään ennuste uudelle arvolle. Tyypillinen esimerkki regressiosta on käytetyn auton hinnan määritys. Syötedataksi annetaan auton merkki ja malli, vuosimalli, ajokilometrit. Selittävänä muuttujana on auton hinta. Opetusdatan pohjalta malli ennustaa auton hinnan, kun muu syötedata on annettu. (Pietikäinen & Silvén 2019, 72)

Regression yleinen erikoistapaus on logistinen regressio, jossa ennustetaan esimerkiksi jonkin tapahtuman todennäköisyys. Tuloksena on kaksi vaihtoehtoa eli tapahtuma joko tapahtuu tai ei tapahdu ja todennäköisyydet näille. Logistisessa regressiossa selittävä muuttuja on tapahtuman riskin luonnollinen logaritmi. Riskillä tarkoitetaan todennäköisyyttä sille, että tapahtuma tapahtuu verrattuna siihen, että se ei tapahdu. (Pietikäinen & Silvén 2019, 72)

Ohjatun oppimisen avulla tehtävä luokittelusta on monentyyppisiä esimerkkejä. Kohteiden tunnistus valokuvista on tyypillinen luokitteluongelma. Myös esimerkiksi sairauden diagnosointi on luokittelua, jossa syötteeksi annetaan lääkärin tekemät havainnot, lääkärin tekemien testien tulokset sekä potilaan oireet. (Pietikäinen & Silvén 2019, 72)

4.2.2 Ohjaamaton oppiminen

Ohjaamattomassa oppimisessa (engl. unsupervised learning) koneelle annetaan data, mutta ei oikeita vastauksia, koska niitä ei ole. Tarkoituksena on, että kone löytää valitun algoritmin avulla datasta rakenteita, säännönmukaisuuksia tai poikkeuksia. Ihmiselle jää tulkinta siitä, mitä kone on löytänyt ja onko löydöksestä hyötyä. (Kananen & Puolitaival 2019, 51)

Yksi yleinen menetelmä käyttää ohjaamatonta oppimista on ryvästäminen eli klusterointi, jossa haetaan datasta keskenään samankaltaisia, mutta muista

eroavia yksiköitä ja ryhmitellään ne omiksi ryhmikseen. Ohjaamattomalla oppimisella tuotetun mallin toimivuus on vaikeampi arvioida, koska ei voida tarkastaa tuloksia oikeista vastauksista. (Elements of AI 2022)

4.2.3 Vahvistusoppiminen

Vahvistetussa oppimisessa ei tarvita aluksi isoa datamäärää, kuten ohjatussa ja ohjaamattomassa oppimisessa, vaan sen sijaan siinä opitaan yritys ja erehdys -periaatteella ikään kuin lennosta havainnoimansa prosessin rinnalla. Ideana on, että kone tulkitsee dataa tai toimii ympäristössään ja tekee valintoja, joista se saa joko positiivisen tai negatiivisen palautteen, jonka se ottaa seuraavissa tekemissään valinnoissa huomioon ja ”oppii”. (Elements of AI 2022; Kananen & Puolitaival 2019, 43)

4.2.4 Siirto-oppiminen

Nykyisin kuvasta objekteja tunnistavaa mallia ei välttämättä tarvitse opettaa täysin alusta, vaan voi käyttää niin kutsuttua siirto-oppimismenetelmää (engl. transfer learning), jossa valmiiksi hieman samankaltaiseen tarkoitukseen jo valmiiksi opetettu koneoppimismalli opetetaan edelleen tunnistamaan uusia kohteita. Rakennetaan siis vanhan päälle. Siirto-oppiminen on ohjatun oppimisen kategoriaan kuuluva menetelmä. Tämä tekniikka nopeuttaa oman mukautetun mallin rakentamista ja voi tapauskohtaisesti parantaa tunnistustarkkuutta ja voi vähentää opetusdatan määrän tarvetta. (Coral 2020)

Siirto-oppimista voi tehdä kahdella tavalla:

- Viimeisten kerrosten uudelleenopetus, jossa vain mallin viimeset kerrokset, joissa kohteen luokittelu tapahtuu, opetetaan uudella datalla. Tämä on nopea prosessi ja vaatii vain pienen uuden opetusdatan.

- Koko mallin uudelleenopetus, jossa kaikki kerrokset opetetaan uudelleen uudella opetusdatalla. Näin saatava malli on tarkempi, mutta opetus vie

enemmän aikaa. Koko mallin uudelleenopetus tarvitsee myös runsaan opetusdatan. (Coral 2020)

Siirto-oppimismenetelmä on yleensä tehokkain rakennettuna niin, että alkuperäinen malli on mahdollisimman yleistävä ja opetettu tunnistamaan paljon erilaisia kohteita, eikä rajattu erikoiskohteisiin. Esimerkiksi kotitaloustavaroiden tunnistukseen opetettu malli voi olla hyvä pohja, kun halutaan rakentaa toimistotarvikkeita tunnistava malli. Jos taas samaan tavoitteeseen käytetään koirarotujen tunnistukseen opetettua mallia, ei lopputulos välttämättä ole niin hyvä. (Coral 2020)

4.2.5 Puoliohjattu oppiminen

Puoliohjattu oppiminen on käytännöllinen koneoppimisen erikoistapaus, jossa mallin opetus tehdään yhdistelemällä ohjaamatonta oppimista sekä ohjattua oppimista. Ideana on, että oppiminen alkaa ohjaamattomasti ja, kun tuloksia datan rakenteesta tai tunnistettavista kategorioista syntyy, voi kone kysyä ihmiseltä apua epäselvistä rajatapausesimerkeistä ja oppia oikeista vastauksista taas lisää. Tässä menetelmässä etuna on, että ei tarvitse valmiiksi luokitella massiivista opetusmateriaalia oppimista varten vaan pienempi ”ohjaus” riittää. Menetelmä myös vähentää ihmisen aiheuttamaa virhettä opetusmateriaalin käsittelyssä. (Pietikäinen & Silvén 2019, 70)

4.3 Neuroverkot ja syväoppiminen

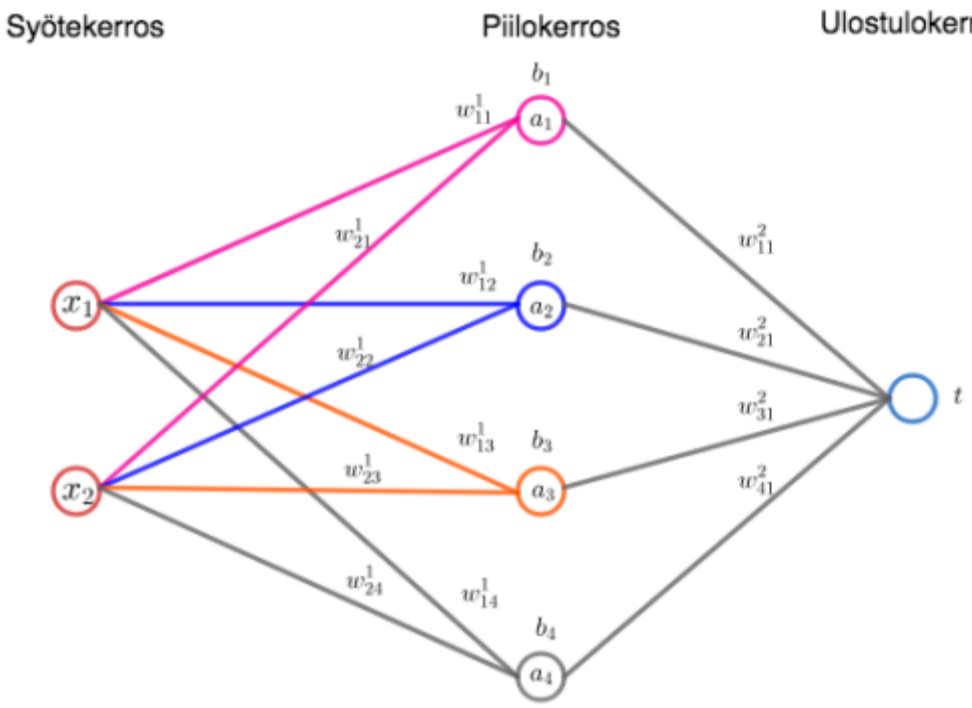
Käsitteet neuroverkko (engl neural network) ja syväoppiminen (engl. deep learning) liittyvät vahvasti toisiinsa tekoälyssä. Syväoppiminen on koneoppimisen osa-alue, joka käytännössä tarkoittaa tekniikkaa, jossa käytetään monikerroksisia neuroverkkoja datan käsittelyyn. Mitä enemmän kerroksia (layer) neuroverkossa on, sitä syvempi verkosta tulee. (Kananen & Puolitaival 2019, 127; Pietikäinen & Silvén 2019, 84)

Syväoppivat neuroverkot ovat lyöneet itsensä läpi 2010-luvulla johtuen siitä, että dataa sekä laskentatehoa on saatavilla paljon aiempaa enemmän. Keinotekoisista neuroverkoista käytetään lyhennettä ANN (artificial neural network). (Tuominen & Neittaanmäki 2019, 6–7). Juuri neuroverkkojen avulla on saavutettu lähivuosina monia teknologioita liittyen kuvan, äänen ja tekstin tunnistukseen, joka on lisännyt kiinnostusta tekoälyä kohtaan (Pietikäinen & Silvén 2019, 13).

4.3.1 Neuroverkon rakenne

Keinotekoinen neuroverkko jäljittelee ihmisaivojen hermoverkkoa, jossa on hermosoluja eli neuroneita. Keinotekoiset neuroverkot on keksitty jo 1940-luvulla. Neuroverkkoon kuuluu aina syöte- ja ulostulokerros (engl. input layer, output layer), joiden välissä on useita, jopa tuhansia piilokerroksia (engl. hidden layer). Kerrokset koostuvat neuroneista, jotka käsittelevät ja välittävät eteenpäin saamaansa syötetietoa. (Tuominen & Neittaanmäki 2019, 6)

Kuviossa 5 havainnollistetaan yksinkertaistettu neuroverkko, jossa *x* on alkuperäinen syötearvo, *w* on painokerroin (engl. weight), *b* on vakiotermi (engl. bias) ja *a* on piilokerroksen antama valmis syöte ulostulokerrokselle. (Tuominen & Neittaanmäki 2019, 26)



KUVIO 5: Yhden piilokerroksen neuroverkko (Tuominen & Neittaanmäki 2019, 26)

Syötekerroksen neuronin *x* syötearvot kerrotaan piilokerroksen neuronin painokertoimella *w* ja lisätään tuloon vakiotermi *b*, jonka summa viedään sitten piilokerroksen aktivointifunktiolle. Aktivointifunktio muuttaa alkuun lineaarisen syötteen epälineaariseksi. Tämän jälkeen piilokerroksen neuronin syöte kerrotaan ulostulokerroksen painokertoimella ja käsitellään sielläkin aktivointifunktiolla, josta saadaan ulostulokerroksen tulos. (Tuominen & Neittaanmäki 2019, 26)

Syötekerroksen neuronien lukumäärä riippuu muun muassa siitä, kuinka monia ominaisuuksia syötedatasta tutkitaan (Tuominen & Neittaanmäki 2019, 24). Neuroverkko on täysin kytketty silloin, kun jokainen neuroni on kytketty seuraavan kerroksen neuroneihin (Pietikäinen & Silvén 2019, 84).

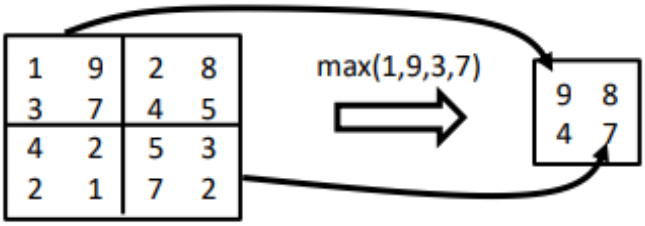
4.3.2 Konvoluutioneuroverkot

Yleisimmät nykyiset syvät neuroverkot hyödyntävät signaalin- ja kuvankäsittelyssä käytettävää konvoluutio-operaatiota. Tällöin on kyse konvoluutioneurover-

koista, joista käytetään lyhennettä CNN (engl. convolutional neural network). (Pietikäinen & Silvén 2019, 89)

Konvoluutio on suhteellisen yksinkertainen matemaattinen operaatio, jolla käsitellään numerotaulukkomuotoista dataa. Digitaaliset kuvat ovatkin tietokoneelle numerotaulukoita eli matriiseja, joissa jokaista pikseliä vastaa numeroarvo. Konvoluutiossa syötteenä olevan kuvan pikseleitä käydään läpi niin sanotulla ominaisuudentunnistajalla (engl. filter tai kernel), joka on syötekuvaa pienempi numerotaulukko. Syötekuvan jokainen kohta käydään läpi vasemmalta oikealle ja ylhäältä alas ominaisuudentunnistaja taulukolla. Voidaan ajatella, että pienempää taulukkoa siirrellään suuremman taulukon päällä. Syötekuvan pikseleiden numeroarvot sekä ominaisuudentunnistajan taulukkoarvot, jotka ovat päällekkäin, kerrotaan keskenään ja nämä tulot lasketaan yhteen ja sijoitetaan uuteen numerotaulukkoon, jota kutsutaan ominaisuuskartaksi (engl. feature map). Näin konvoluutiolla luotu ominaisuuskartta sisältää samat ominaisuudet kuin alkuperäinen syötekuva, mutta on huomattavasti pikselimäärältään sitä pienempi. Konvoluutiokerros muuntaa syötekuvan aina neliömäiseksi eli kuvan korkeuden ja leveyden pikselimäärä on sama, esimerkiksi 416x416 pikseliä. (Kananen & Puolitaival 2019, 149-153)

Konvoluutiota seuraa pooling-niminen vaihe, jossa konvoluution tuloksena saatua ominaisuuskarttaa tai piirrekuvaa pienennetään ja saadaan esiin syötekuvan olennaisimmat piirteet. Kuviossa 6 nähdään niin kutsuttu max-pooling, jossa ominaisuuskartan neljä samassa ruudussa olevaa pikseliä korvataan yhdellä ruudun suurimman vastearvon omaavan pikselin arvolla eli vain suurimman vastearvon saaneet piirteet menevät jatkoon. Numero 9 on suurin luku vasemman yläkulman ruudussa, numero 8 on suurin luku oikean yläkulman ruudussa, numero 4 on suurin luku vasemmassa alakulmassa ja numero 7 on suurin viimeisessä ruudussa. Tulokseksi poolingista saadaan oikeanpuoleinen neljän pikselin ruutu. (Pietikäinen & Silvén 2019, 87)



KUVIO 6. Max-pooling (Pietikäinen & Silvén 2019, 87)

4.3.3 Algoritmit

Algoritmeista puhutaan tekoälyn ja koneoppimisen yhteydessä paljon. Algoritmin yksi määritelmä on tapa tehdä asioita tai ratkoa ongelmia vaiheittain yksinkertaisia sääntöjä noudattaen (Louridas 2021, 5). Arkielämän algoritmi voi olla esimerkiksi ruoan valmistusresepti, mutta algoritmeista puhutaan yleisesti vasta tietokoneiden matemaattisten prosessien yhteydessä. Eri tavoin toimivia algoritmeja on paljon erilaisiin käyttötarkoituksiin ja ne nimetään usein joko käyttötarkoituksen tai keksijänsä mukaan. Esimerkiksi Thomas Bayesin keksimä Bayes-algoritmi on näin nimetty. (Kananen & Puolitaival 2019, 112)

Yleisimpiä koneoppimiseen liittyviä algoritmeja ovat: K lähimmän naapurin luokitin (engl. K-Nearest Neighbour), tukivektorikone (engl. Support Vector Machine), Naive Bayes, päätöspuu (engl. Decition Tree) ja satunnaismetsä (engl. Random Forrest). (Kananen & Puolitaival 2019, 118-126)

Koneoppimisen algoritmit voidaan jakaa viiteen eri kategoriaan:

1. Regressioalgoritmit: Jos A muuttuu mitä tapahtuu B:lle?
2. Luokittelualgoritmit: Kuuluuko havaittu kohde luokkaan A vai B?
3. Ryhmittelyalgoritmit: Kuuluuko havaittu kohde ryhmään A, B vai C?
4. Sijoitusalgoritmit: Mikä on asioiden järjestys?
5. Generaatioalgoritmit: Kuinka asioita voidaan tuottaa koneellisesti?

Koneoppimisalgoritmien vahvuus on, että niillä voi ratkaista moniulotteisia ongelmia sekä niiden toimintaa voidaan kokeilla verrattain nopeasti ja, että ne ovat

toimintaperiaatteeltaan suhteellisen helppoja ymmärtää. (Kananen & Puolitaival 2019, 113)

5 KONENÄKÖ

5.1 Mitä on konenäkö?

Konenäkö (engl. machine vision) jäljittelee ihmisnäköä ja myös ylittää sen kyvyt. Sitä käytetään teollisuudessa toistuvissa liukuhihnalla tapahtuvissa tuotetarkastuksissa, kuten esimerkiksi pullonpalautusautomaateissa, koska se on ihmiseen nähden tarkempi, nopeampi ja väsymätön. Konenäköjärjestelmässä on yleensä seuraavat osat: kohde, kamera, valonlähde, tietokone ja ohjelma, joka tulkitsee kameran lähettämän kuvasignaalin käyttötarkoitukseen sopivalla kuvantunnistusmenetelmällä. (Tuominen & Neittaanmäki 2019, 9)

Konenäön tavoitteena yleisesti on kehittää koneelle näköaisti sekä ymmärrys näkemästään. Jotta konenäkö olisi yleispätevä erilaisiin sovelluksiin, sen olisi osattava tunnistaa eri asennoissa ja eri etäisyyksillä olevia, jopa liikkuvia kohteita. Konenäön kehittäminen on monialainen haaste, jonka parissa työskentelee muun muassa insinöörejä, matemaatikkoja, tietokonealan asiantuntijoita, ihmisnäön psykologian tutkijoita sekä fyysikkoja. (Pietikäinen & Silvén 2019, 127–128)

Läheinen termi konenäölle on tietokonenäkö (engl. computer vision), jolla tarkoitetaan korkeatasoisempaa digitaalisten kuvien ja videoiden käsittelyä ja niiden ymmärrystä. Tietokonenäkö onkin osa tekoälyä ja koneoppimista, vaikka termiä käytetäänkin monesti konenäön kanssa synonyyminä. (Tuominen & Neittaanmäki 2019, 9)

5.2 Kuvantunnistus

Tässä työssä käsitteestä object detection käytetään suomenkielistä termiä kuvantunnistus, joka on konenäön osa-alue. Monissa eri lähteissä tämän käsitteen suomennos vaihtelee paljon, koska yleistä alan termistöä suomeksi ei ole.

Kuvantunnistus tarkoittaa kuvassa tai videolla olevien haluttujen ja opetettujen kohteiden, eli esimerkiksi esineiden tai henkilöiden tunnistusta (Karimi 2021). Kuvantunnistus on osa tietokonenäön tutkimusalaa. Moderni tapa tehdä kuvantunnistus on käyttää syväoppivia konvoluutioneuroverkkoja. Kuvantunnistus yhdistetään usein esimerkiksi autonomisesti ajaviin autoihin ja automaattiseen videovalvontaan esimerkiksi tapahtumissa ja julkisilla paikoilla turvallisuuteen liittyen. (Świeżewski 2020)

Kuvantunnistus koostuu kahdesta käsitteestä, jotka ovat vapaasti suomentaen kuvan luokittelu (engl. image classification) ja kohteen paikannus (engl. object localization). Näillä termeillä syötekuvasta tunnistetaan mitä kuva esittää tai sisältää, ja missä kohdassa kuvaa sijaitsee kuvan luokan määrittävä kohde. Kuvantunnistuksessa kuvassa voi olla myös useita tunnistettavia kohteita. (Świeżewski 2020)

Kuvantunnistuksella siis pyritään vastaamaan kahteen kysymykseen:

1. *Mikä kohde*? Pyritään tunnistamaan haluttu kohde kuvasta.

2. *Missä kohde on*? Pyritään tunnistamaan kohteen sijainti kuvassa.

Tapoja ja algoritmeja tunnistaa kohteita kuvasta on useita kuten Fast R-CNN, Retina-Net, and Single-Shot MultiBox Detector (SSD). Silti suosituimmaksi algoritmiksi kuvantunnistukseen on noussut nykyisin YOLO, johtuen sen hyvästä suorituskyvystä (tarkkuus suhteessa nopeuteen) verrattuna edellä mainittuihin. (Karimi 2021). Kappaleessa 7.7. käsitellään lisää kuvantunnistuksen tarkkuuden määritelmää.

Erilaiset tunnistusalgoritmit jaetaan kahteen ryhmään: luokitteluun perustuvat algoritmit ja regressioon perustuvat algoritmit. Ensimmäiseen luokkaan kuuluu esimerkiksi R-CNN ja jälkimmäiseen esimerkiksi YOLO. (Świeżewski 2020)

6 VAURIOTUNNISTIMEN TOTEUTUS

6.1 Tutkimuksen tavoite

Tässä opinnäytetyössä tavoitteena oli rakentaa kokeellinen koneoppimismalli auton kolarivaurion tunnistukseen käyttäen apuna työpaikkani K-Auto Oy:n vauriokorjaamon vahinkotarkastuskuvia sekä YOLO-tunnistusmenetelmää. Tutkimuskysymyksenä on, voidaanko vanhaa kuvadataa hyödyntämällä saada opetettua vaurioita tunnistava malli?

Työssä kerättiin opetusdataksi autojen vauriokuvia, jotka annotoitiin, ja joilla koneoppimismalli ohjatusti opetettiin tunnistamaan neljän eri kategorian vaurioita. Vaurion tunnistukseen käytetään järjestyksessä neljännen version YOLOv4-tunnistusalgoritmia Darknet -nimisellä arkkitehtuurilla. Ensimmäinen koneoppimismalli toteutettiin kotitietokoneella käyttäen apuna yksittäisen Nvidia-näytönohjaimen laskentatehoa, jonka jälkeen samalla datalla luotiin myös toinen malli käyttäen TAMK:in tehokkaampaa, koneoppimiskuormiin tarkoitettua tietokonetta sekä samaa tunnistusarkkitehtuuria. Valmiiden mallien tunnistuskykyä testattiin niille ennestään tuntemattomilla vauriokuvilla ja tunnistuskykyä analysoitiin tuloksien pohjalta.

6.2 Käytetty data

Aineistona tässä työssä käytetään työpaikkani K-Auto Oy:n vauriokorjaamoiden ottamia valokuvia autojen kolarivaurioista. Olen saanut esimieheltäni luvan kuvien käyttöön sillä ehdolla, että autojen tunnistetietoja tai tunnistettavia henkilöitä ei tässä työssä julkaistavissa kuvissa näy. Koneoppimismallin opetusdatassa näitä tietoja saa olla, koska opetusdataa tai mallia ei julkaista kokonaan tässä raportissa.

Kuvat ovat oikeita työelämässä käytettyjä kuvia K-Auto Oy:n eri paikkakuntien toimipisteiltä, joilla tosielämän asiakastapausten vauriot on dokumentoitu ja korjauskustannusarviot laskettu. Erilaisia vauriotapauksia ei ole erikseen valikoitu

millään tavalla, vaan mukana on satunnainen otos erilaisten vauriotapausten kuvia. Tähän työhön opetusaineistoksi päätyneitä kuvia otetaan eri paikkakuntien korjaamoilla hieman erilaisilla kameroilla hieman eri asetuksilla ja tämän vuoksi muun muassa kuvatiedostojen koko ja laatu vaihtelee. Mukana on myös hieman jopa asiakkaiden ottamia vauriokuvia, koska vahinkotarkastuksia voidaan tehdä myös asiakkaan lähettämillä kuvilla.

Opetusaineiston kuvien rajausta pohdittiin aluksi esimerkiksi taustaympäristön, auton värin, korimallin, automallin, sisä- tai ulkovalon tai vaurion kohdan mukaan. Enemmän valikoitu aineisto voisi olla rajatumpaan tarkoitukseen tarkempi, varsinkin jos opetusaineisto on suppea. Nyt kuitenkin ajatuksena oli rakentaa yleisempi vauriotunnistin, jota voi käyttää kaikenlaisiin vauriokuviin, joten tämän vuoksi aineiston kuvia ei valikoitu minkään edellä mainitun kriteerin mukaan. Ainoa rajaus kuvien valinnassa on tehty auton ulkopuolisista osista otettuihin kuviin eli opetusdatassa ei ole mukana purettujen autojen kuvia tai auton sisäosien kuvia, koska tässä työssä tarkoituksena on tunnistaa vauriot auton pinnasta sellaisena kuin ne ovat kolaripaikalla tai asiakkaan kotipihassa. Suurimmaksi osaksi kuvat ovat henkilöautoista ja mukana on myös hieman pakettiautoja. Alkuperänsä takia aineiston kuvat ovat suurimmaksi osaksi Volkswagen -konsernin eri ikäisiä automalleja.

Itse valokuvia ei ole käsitelty erikseen millään tavalla ennen annotointia. Vain kuvien tiedostonimet on muutettu, koska alkuperäiset tiedostonimet olivat muotoa rekisterinnumero-järjestysnumero-kuvateksti-päivämäärä-kellonaika. Tiedostonimet sisälsivät välejä ja joissain nimissä oli myös erikoismerkkejä, jotka häiritsevät kuvia lukevaa algoritmia, joten opetusajoa varten tiedostonimet oli muutettava tasalaatuisiksi, jotta ne päästään käymään läpi.

Opetusaineiston laadulla on suuri merkitys. Mikäli data on riittävän korkealaatuista, pystyy tekoäly jopa ylittämään ihmisen suorituskyvyn rajatuissa sovelluksissa (Pietikäinen & Silvén 2019, 116). Darknet YOLOv4 -ohjeistuksen mukaan laatu olisi hyvä olla opetusdatassa vastaava kuin tunnistettavien kuvien laatu, riittävä kuvan resoluutio ja kohdistuksen tarkkuus.

Vauriokuvia kerättiin yhteensä 2259 kappaletta tätä työtä varten. Jokaista tunnistettavaa luokkaa varten on siis satoja esimerkkikuvia ja jopa tuhansia annotoituja esimerkkikohteita. Vaurioita tunnistavia malleja opetettiin työssä kaksi kappaletta. oiseen malliin jätettiin pois autot tunnistava viides luokka, jolloin annotoitujen kuvien kokonaismäärä väheni 1965 kappaleeseen. Kuvien resoluutio vaihtelee eri kameroista ja kuvausmoodeista johtuen 1280 x 960 ja 1800 x 1350 välillä. Laatu kuvissa on sellainen, jolla ihminen pystyy tehdä tunnistuksen, joten odotus on, että silloin sen pitää riittää tietokonenäöllekin.

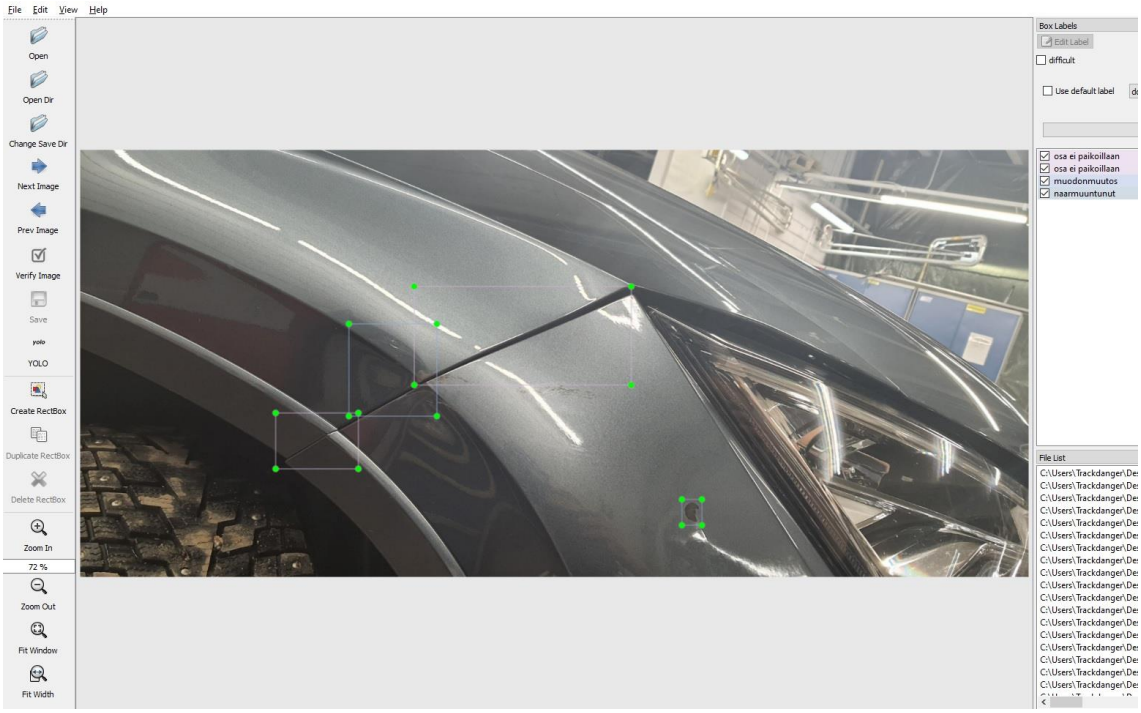
6.3 Kuvadatan annotointi

Koneoppimismallin opetus edellyttää, että pohjana on riittävä määrä esimerkkidataa, josta malli voi oppia. Kun kyseessä on ohjatun oppimisen metodi pitää raakadata annotoida eli kertoa mitä tärkeitä tunnistettavia piirteitä missäkin datayksikössä on. Toinen samaa tarkoittava termi on labelointi (engl. labeling). (Bochkocskiy 2020)

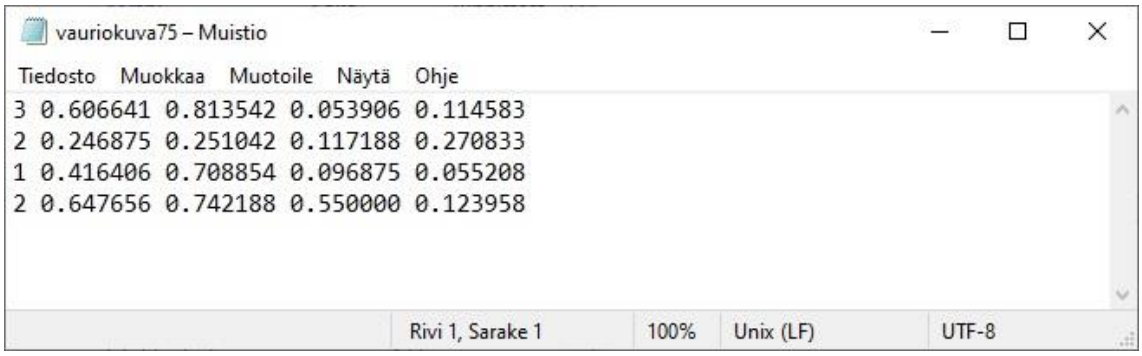
Tässä työssä on käytetty ohjelmaa nimeltä LabelImg, joka on kirjoitettu Pythonilla ja käyttää Qt ohjelmistoympäristöä (Lin 2018). LabelImg valittiin, koska se tukee YOLO-formaattia ja oli arvioiden mukaan helppo asentaa ja oppia käyttämään. Annotointivaiheessa on tiedettävä mitä tunnistusmenetelmää jatkossa käytetään, jotta annotointiedoston muoto on sitä varten oikea.

Käytännössä asennuksen jälkeen LabelImg -ohjelmaan avataan mallin opetusmateriaalin hakemistosijainti ja yksi kuva kerrallaan hiiren osoitinta käyttäen merkitään opetettavat kohteet ympäröimällä ne suorakaiteenmuotoisilla laatikoilla, joita kutsutaan englanniksi nimellä bounding box. Kuvassa 3 on näkymä LabelImg ohjelmasta sekä tämän työn kohteita merkittynä. Kohteita voi olla useita samassa kuvassa ja niille annetaan jokaiselle luokan mukainen nimike (label). Kohteita on siis huomattavasti suurempi määrä kuin itse kuvia. Tähän työhön itse tehtyjä annotointeja opetusdatan kohteille oli yhteensä 8356 kappaletta. Nimikkeet tallentuvat omina tekstitiedostoinaan kuvakansioon, samalla nimellä kuin itse kuvan tiedostonimi. Jokaista .jpg -tiedostoa kohti ohjelma luo samannimisen .txt -tiedoston. Kuvassa 4 näkyy YOLO-formaatin muotoinen an-

notointitiedosto avattuna. Yksi tekstirivi vastaa yhtä kuvaan merkittyä kohdetta. Ensimmäisenä on kohteen luokka, seuraavana kohteen keskipisteen x- ja y-koordinaatit sekä kohteen leveys ja korkeus (<object-class> <x_center> <y_center> <width> <height>). (Bochkocskiy 2020)



KUVA 3. Näkymä LabelImg -ohjelmasta sekä merkittyjä kohteita



KUVA 4. YOLO-annotointitiedoston sisältö

6.4 Tunnistettavat vaurioluokat ja työn rajaus

Työpaikallani on sanonta, että jokainen vaurio on ainutlaatuinen - ei ole olemassa kahta täysin identtistä kolarivauriota. Kun automaattista vaurioiden tunnistusta rakennetaan pitää vaurioita kuitenkin luokitella.

Pohdin pitkään, että pyrinkö saamaan tunnistuksen vaurioiden lisäksi myös kohteen sijainnille niin, että tunnistin tietäisi missä osassa autoa vaurio sijaitsee. Tämä olisi tärkeä kyky pidemmälle viedyssä vauriotunnistimessa. Tulin kuitenkin siihen tulokseen, että myös auton eri osat pitäisi opettaa tunnistimelle ja silloin luokkia tulisi todella suuri määrä. Mitä enemmän luokkia on, sen enemmän opetusmateriaalia ne tarvitsisivat ja sitä enemmän aikaa mallin rakennus tarvitsisi. Tämä voisi vaatia myös erilaista pohjadataa tai eri annotointivälineen.

Toinen pohdinnassa ollut aspekti oli vauriohavainnon jälkeinen korjausmenetelmän valinta esimerkiksi päätöspuulogiikalla eli kone päättelisi vaurion tyypistä minkälainen korjaustoimenpide olisi sopiva kullekin vauriolle. Tässä vaikuttaa myös se missä osassa autoa vaurio sijaitsee ja mitä materiaalia vaurion kärsinyt osa on. Nämä molemmat ideat olisivat laajuudessaan jo ihan omia projektejaan, joten rajasin ne ulos tästä työstä.

Rajaan tunnistuksen tässä kokeilussa pelkästään valitsemiini neljään eri vaurioluokkaan. Ensimmäisessä mallissa on mukana viideskin luokka, joka tunnistaa kuvasta autot, mutta se karsittiin pois seuraavan mallin rakennuksessa, koska tuo luokka ei ollut tärkeä työn tavoitteen kannalta ja oli myös tärkeää saada mallia kevennettyä.

Seuraavaksi esitellään tunnistimeen valitut vaurioluokat. Esimerkkikuvat on kaapattu LabelImg-ohjelmasta kuvien annotointivaiheessa ja näissä mallin opetukseen tarkoitetut kohteet on rajattu kuvaan bounding boxeilla.

Naarmuuntunut

Tähän vaurioluokkaan on merkitty yksittäiset naarmut sekä naarmuuntuneet alueet, hankaumat ja pienet pintavauriot kuten maalipintarikot (kuva 5).

KUVA 5. Esimerkkikuva vaurioluokasta naarmuuntunut (K-Auto Oy)

Muodonmuutos

Tähän vaurioluokkaan kuuluvat lommot, painaumat ja taittumat kaikissa materiaaleissa. Myös muovipuskurin murtumat on merkitty tähän vaurioluokkaan (kuva 6). Tämä luokka on todennäköisesti haastavin ainakin rajata oikein, koska muodonmuutos ei ole kohteena selkeästi näkyvä ääriviivoiltaan. Tietynlaiset muodonmuutokset voivat näkyä auton pinnasta vain tietyssä kulmassa eli valon tulosuunta vaikuttaa kohteen näkyvyyteen. Peltipinnassa olevan tietyn tyyppisen lommon äärirajojen havaitseminen ihmissilmälläkin on monesti vaikeaa.

KUVA 6. Esimerkkikuva vaurioluokasta muodonmuutos (K-Auto Oy)

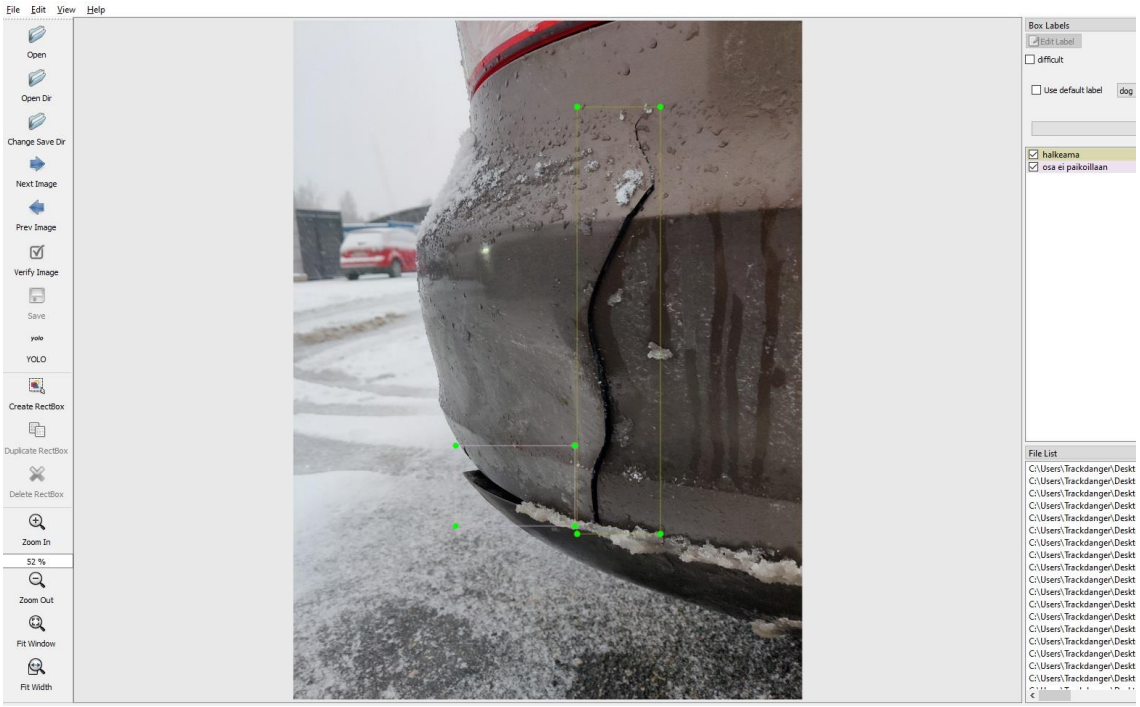
Osa pois paikoiltaan

Osien välisten rakojen laajenemat, osittain kiinniolevat repsottavat osat, kokonaan tai osaksi puuttuvat osat on merkitty tähän vaurioluokkaan. Kuvassa 7 nähdään esimerkki tästä luokasta. Kokonaan rikkinäiset valoumpiot on myös merkitty tähän vaurioluokkaan.

KUVA 7. Esimerkkikuva vaurioluokasta osa pois paikoiltaan (K-Auto Oy)

Halkeama

Selvät halkeamat ja repeämät materiaalista riippumatta kuuluvat tähän vaurioluokkaan (kuva 8).



KUVA 8. Esimerkkikuva vaurioluokasta halkeama (K-Auto Oy)

6.5 Ensimmäinen malli

Ensimmäisessä mallissa oli mukana viisi luokkaa: ”naarmuuntunut”, ”muodonmuutos”, ”osa ei paikoillaan”, ”halkeama” ja ”car”. Alkuun ideana oli luoda sääntö, että vauriotunnistukset on oltava siinä kohdassa kuvaa, jossa auto sijaitsee eli vauriot eivät voi sijaita auton ulkopuolella kuvassa. Tätä sääntöä ei kuitenkaan tarvinnut tehdä, koska testatuissa kuvissa vauriot tunnistettiin aina autosta eikä taustasta ja myös seuraavaa mallia haluttiin keventää opetuksen nopeuttamiseksi, joten tämä ”car”-luokka jätettiin kokonaan pois toisesta mallista.

Myöhemmin ensimmäisen opetusajon kohdalla huomasin, että Darknetin ohjeistus suosittelee vähintään 4GB:n (gigatavun) RAM-muistikapasiteetilla varustettuja Nvidia näytönohjaimia, mutta oman Nvidia-kortin muisti olikin vain 2GB.

Tämä johti siihen, että jotta sain opetusajon alkamaan, piti kuvan lukuasetukset configure-tiedostoon säätää todella pieneksi.

Ensimmäisen mallin opetusajo sekä testikuvien tunnistus toteutettiin Windows CMD komentokehote käyttöliittymällä. Esimerkiksi ensimmäinen opetusajo käynnistettiin antamalla käsky:

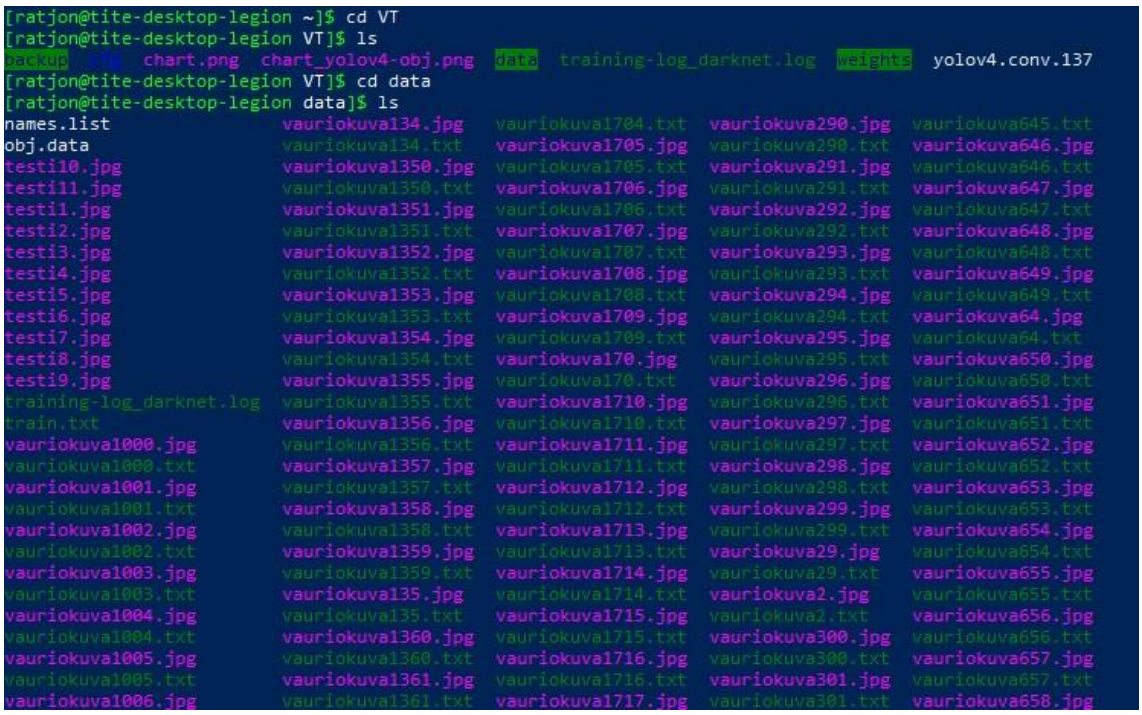
darknet.exe detector train data/obj.data cfg/yolov4-obj.cfg yolov4.conv.137

Tällä käskyllä Darknet aloittaa opetusajon, jossa se käyttää data/obj.data polusta löytyvää opetusdataa, yolov4-obj.cfg konfigurointitiedostoa sekä yolov4.conv.137 alkupainoja. Kuten aiemmin totesin, jotta opetusajo saatiin kotikoneen pienemmällä muistikapasiteetilla edes alkamaan, oli kuvan resoluutio, jolla neuroverkko käsitteli annotoituja kuvia mallin opetuksessa, säädettävä kooltaan vain 192x192 pikseliin. Darknetin ohjeen mukaan tämä olisi hyvä olla minimissään 416x416 pikseliä. Liian pieni kuvakoko voi vähentää ratkaisevasti tunnistuksen tarkkuutta ja tehdä tuloksista epäluotettavia. Ensimmäisessä mallissa on myös kuvissa mukana confidence score eli varmuusprosentti mallin tekemästä tunnistuksesta.

Excalibur-kotitietokoneen kokoonpano:
- Intel Core i7-4790K, 4.00 GHz, 4-ydinprosessori
- RAM-muistia 8,00 Gt
- Nvidia GeForce GTX 960 näytönohjain

6.6 Toinen malli

Kun ensimmäinen malli oli saatu onnistuneesti kotikoneella toimimaan rajoituksineen, oli aika kokeilla samaa leveämmin hartein. Toinen malli toteutettiin Tampereen ammattikorkeakoulun koneoppimiskuormiin rakennetulla Legion -nimisellä tietokoneella, jota pystyi käyttämään kotikoneelle asennetulla suojatulla etäyhteydellä Windows Powershell -sovelluksen kautta kätevästi (kuva 9).



KUVA 9. Powershell-näkymä Legionin vauriotunnistin -kansioon

Tampereen yliopiston ja ammattikorkeakoulun yhteinen SURE-projekti käyttää samaa AlexeyAB:n Darknet -arkkitehtuuria, joten pääsin nopeasti kouluttamaan mallia siirtämällä oman opetusdatan sekä CFG-tiedostot Legionille, koska itse kokoonpano oli siellä valmiina. Koska laskentatehoa ja muistikapasiteettia Legionissa on moninkertaisesti kotitietokoneeseeni verrattuna, tein CFG-tiedostoon muutokset suuremmasta 992x992 resoluutiosta kuvan lukemiseen siinä toivossa, että mallista tulee huomattavasti tarkempi kuin ensimmäisestä mallista ja varsinkin pienempien ja etäämmällä kuvassa olevien vaurioiden tunnistukseen, joka oli ensimmäisen mallin selkein puute. Toisen mallin testikuviin ei tullut mukaan confidence score, johtuen kuvien lukemiseen käytetystä Python koodista, jolla saatiin käytyä läpi kerralla suurempi otos testikuvia.

Legion-tietokoneen kokoonpano:
- AMD Ryzen 9 3900X, 3,8 GHz 12 ytimen prosessori
- Samsung 970 Pro 1TB SSD massamuisti
- Nvidia Quadro RTX 5000 16 Gt näytönohjain

7 TUNNISTUKSEN ARKKITEHTUURI

7.1 AlexeyAB:n Darknet

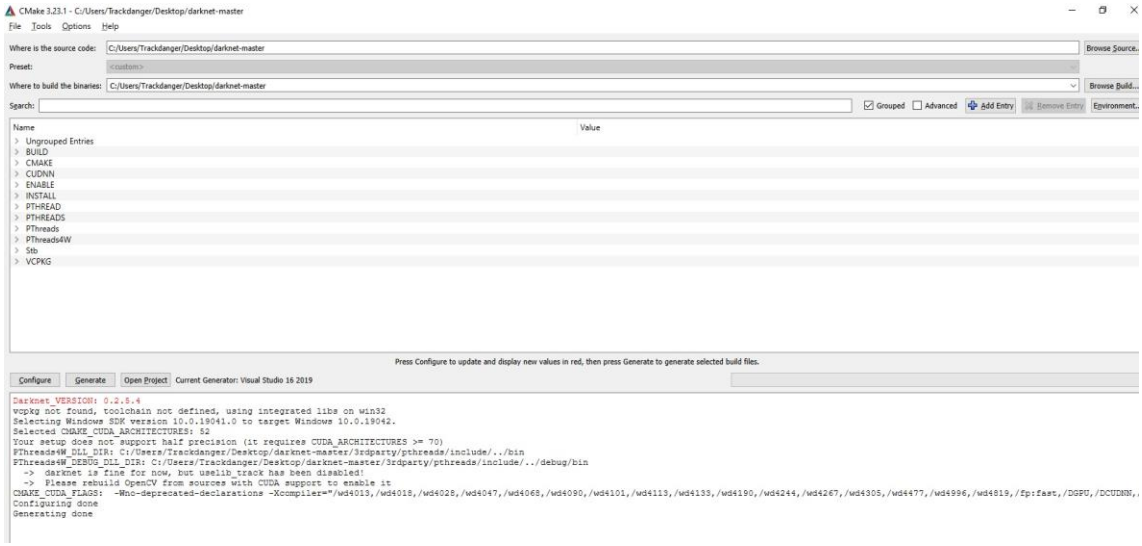
Tässä opinnäytetyössä käytettiin runkona GitHubista löytyvää AlexeyAB:n Darknet -arkkitehtuuria, joka on monikäyttöinen ja suorituskykyinen avoimen lähdekoodin neuroverkkorakenne, jossa ohjelmointikielinä on C ja CUDA (Bochkocskiy, A. 2020). Tässä työssä tähän arkkitehtuuriin viitataan lyhyesti Darknetina. Valitsin Darknet -arkkitehtuurin, koska se mahdollistaa suorituskykyisen YOLO-tunnistusalgoritmin käytön ja myös, koska se tukee Nvidia -merkkisten näytönohjainten GPU-laskentatehon hyödyntämistä. Se on myös suunniteltu varta vasten yhden GPU:n avulla toimivan tunnistimen kouluttamiseen. Mallin olisi voinut luoda myös CPU-laskentateholla, mutta se olisi huomattavasti hitaampi ja kestäisi moninkertaisesti kauemmin (Bochkocskiy, A. 2020). Darknet ei myöskään vaadi kattavaa koodausosaamista, joka helpotti käyttöönottoa omalta kohdaltani.

7.2 Koonpano

Darknetin ohjeen mukaan seuraavia työkaluja tarvitaan, jotta saadaan tunnistin toimimaan. Kyseiset ohjelmistoversiot on valittu yhteensopivuuden vuoksi tähän työhön. Yhteensopivien versioiden löytäminen oli etsinnän ja kokeilun tulos, mutta lopulta oikea kokoonpano löytyi. Muitakin toimivia yhdistelmiä varmasti löytyy, mutta ihan kaikki versiot eivät kaikkien ohjelmien kesken olleet yhteensopivia, kuten huomasin kokoonpanoa rakentaessa.

CMake 3.23

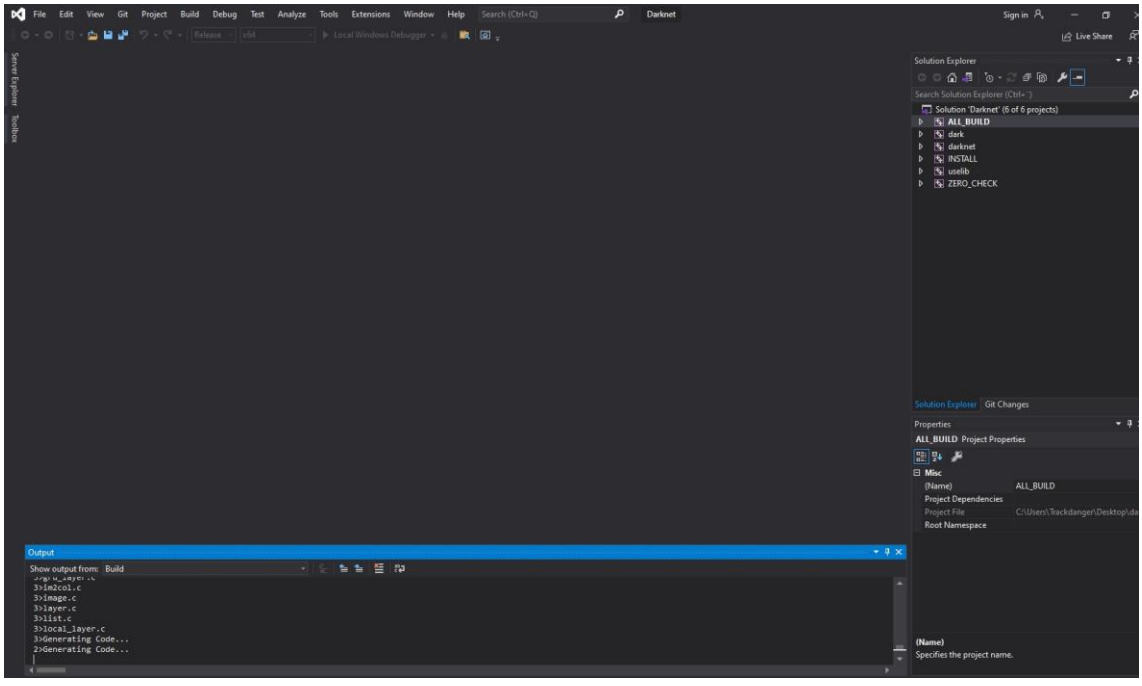
Avoimen lähdekoodin, monialustainen työkalukirjasto ohjelmistojen rakennukseen, testaukseen sekä pakkaukseen (Cmake 2022). Näkymä Cmakesta kuvassa 10.



KUVA 10. Cmake

Visual Studio 2019

Kattava integroitu ohjelmistokehitysympäristö Windowsille (Microsoft 2022). Kuvassa 11 on näkymä Visual Studio 2019 ohjelmasta.



KUVA 11. Visual Studio 2019

CUDA 11.2

NVIDIA® CUDA® Toolkit on Nvidian tarjoama kehitysympäristö GPU-kiihdytettyihin sovelluksiin (Nvidia 2022). Asennetaan Nvidia-näytönohjaimen lisäosaksi.

cuDNN 8.0

NVIDIA CUDA® Deep Neural Network library (cuDNN) on syvien neuroverkkojen funktioihin keskittynyt kirjasto. Se tarjoaa asiantuntijatason valikoiman standardoiduista neuroverkkojen eri kerroksista, kuten konvoluutiot, poolingin, normalisoinnit ja aktivointikerrokset. (Nvidia 2022)

OpenCV 4.5.4

Open Source Computer Vision Library. OpenCV on avoimen lähdekoodin tietokonenäköön ja koneoppimiseen kehitetty ohjelmistokirjasto. (OpenCV 2022)

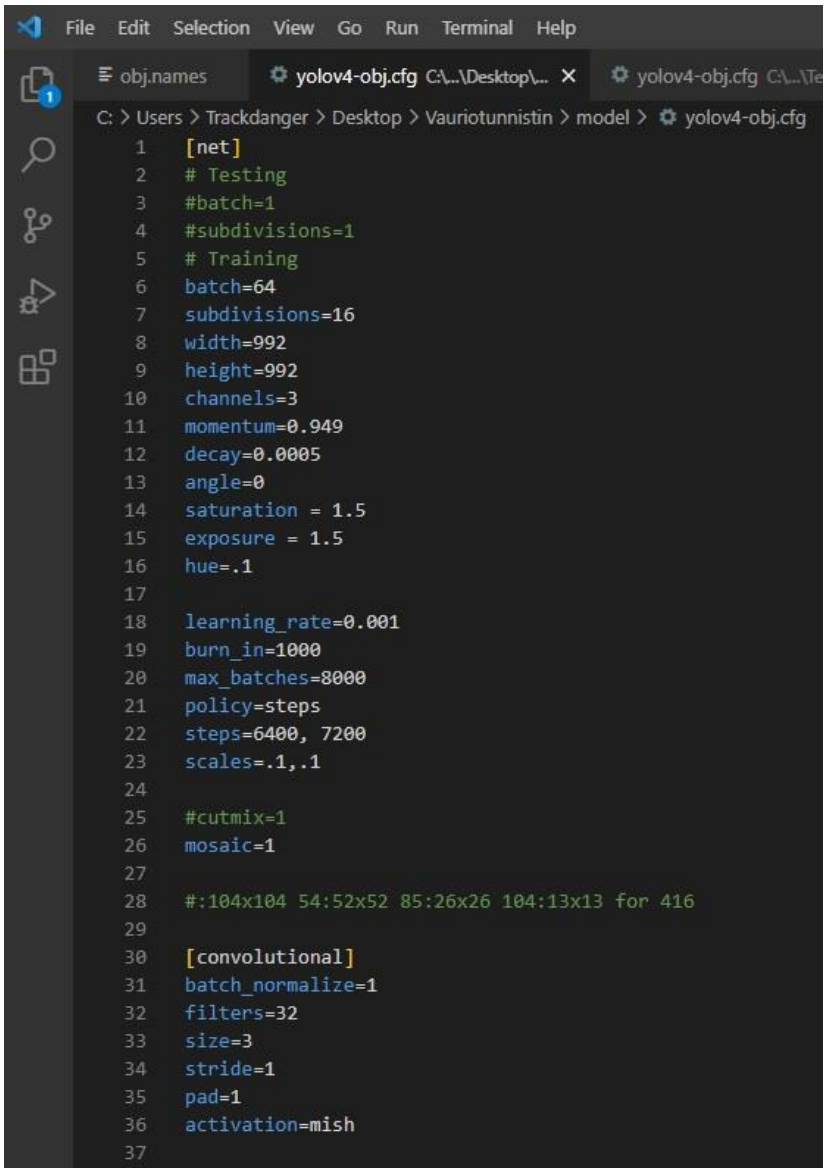
Ohjelmat asennettiin Windows-käyttöjärjestelmälle tarkoitetuilla ohjeilla ja ohjelmistoversioilla. Ohjeistukset löytyivät myös Linux ja macOS käyttöjärjestelmille. Ohjelmien asentaminen piti tehdä oikeassa järjestyksessä ja myös Windowsin ohjauspaneelista löytyvään ympäristömuuttujavalikkoon oli tehtävä polut Nvidia CUDA Toolkitille sekä OpenCV:lle, jotta Windows-järjestelmässä toimiva Darknet löytää ne käyttöönsä. Tekstieditorina työssä on käytetty Visual Studio Codea.

7.3 Mallin luontiin vaadittavat tiedostot

Käytännössä mallin opetuksen haluttu konfigurointi toteutettiin neljän eri tiedoston avulla: obj.data, obj.names, yolov4-obj.cfg ja yolov4-obj.conv.137.weights. Obj.data pitää sisällään lyhyet tekstirivit tunnistettavien luokkien määrästä sekä arvot ”train”, ”names” ja ”backup” -muuttujille. Obj.names sisältää vain listan luokkien nimistä, eli tässä työssä luokat ”naarmuuntunut”, ”muodonmuutos”, ”osa pois paikoiltaan” ja ”halkeama”. Yolov4-obj.cfg on näistä tärkein tiedosto eli

itse koodi tunnistimen toiminnalle. 1161-rivistä kooditiedostoa muuntelemalla voidaan vaikuttaa mallin opetuksen parametreihin antamalla arvot, joita algoritmit hyödyntävät kuvien käsittelyssä (kuva 12). Darknet antaa ohjeet konfigurointitiedoston säätämiseen oman mallin rakentamisessa. Koodia muutetaan sopivaksi opetusdatan määrän, tunnistettavien luokkien määrän sekä tietokoneen laskentatehon osalta.

Yolov4-obj.conv.137.weights-tiedosto sisältää alkuperäiset painot, joilla opetus lähtee aluksi liikkeelle. Mallin opetusajo luo useamman weights-tiedoston opetusajon edetessä, joissa painot säätyvät opetusajon edetessä ja hakevat oikeaa arvoaan suhteessa opetusdataan eli malli ”oppii”. Valmis malli sekä weights-tiedostot tallentuvat backup -kansioon.



KUVA 12. Visual Studio Codessa avattu yolov4-obj konfigurointitiedosto

7.4 YOLO

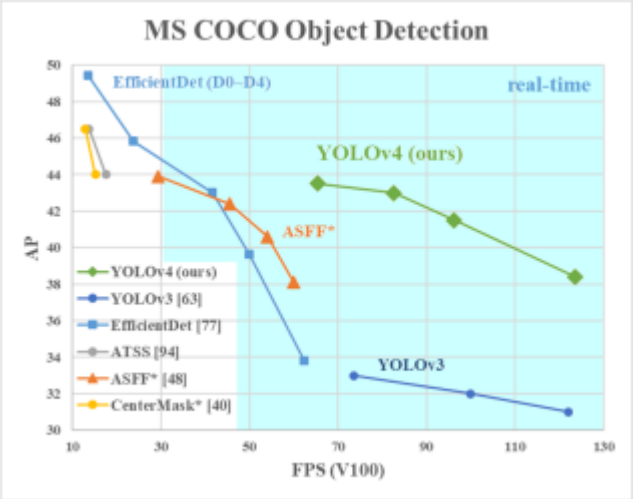
YOLO on vuonna 2015 kehitetty kohteen tunnistusalgoritmi, joka käyttää neuroverkkoja kuvantunnistukseen. YOLO on lyhenne sanoista You Only Look Once eli suomeksi käännettynä "katsot vain kerran”. Tämän työn aikana on julkaistu ensimmäiset versiot jo seitsemännen sukupolven YOLO-algoritmista (YOLOv7). Jokainen sukupolvi on paranneltu versio edellisestä. YOLO käsittelee tunnistusta regressio-ongelmana ja tarjoaa myös tunnistuksen todennäköisyyden. YOLO käyttää konvoluutioneuroverkkoja rakenteessaan ja toisin kuin vanhemmat algoritmit se ei tarvitse vastavirta-algoritmia (engl. backpropagation), vaan kertaluontoisen neuroverkon ajon tehdäkseen kohteen luokasta ennusteen. YOLO on siis yksitasoinen tunnistusalgoritmi. (Karimi 2021; Maindola 2021)

Mitä YOLO sitten tekee? YOLO muodostaa ennusteen kohteen luokasta ja sijainnista kuvassa, joka rajataan bounding boxilla eli kohteen ulkomitat rajaavalla suorakaiteen muotoisella laatikolla. Jokainen bounding box voidaan kertoa seuraavilla kriteereillä:

- keskipiste **(bxby)**
- leveys **(bw)**
- korkeus **(bh)**
- arvo **cis**, joka viittaa kohteen luokkaan (auto, koira, hedelmä…)

YOLO ei etsi syötekuvasta suoraan kiinnostavia alueita, jotka voisivat potentiaalisesti sisältää kohteen, vaan se jakaa syötekuvan tyypillisesti 19x19 ruudun ruudukoksi, joista jokaisesta ruudusta se pyrkii löytämään 5 kappaletta mahdollisia kohteita. Kohteitahan voi olla yhdessä kuvassa enemmän kuin yksi. Tällöin päädymme 1805 kappaleeseen bounding boxeja yhtä kuvaa kohden. YOLO laskee seuraavaksi näiden löydettyjen mahdollisten kohteiden todennäköisyyden ja pudottaa seuraavaksi pois ruudut, joiden kohteen sisältämisen todennäköisyys on matala eli ne ruudut, jotka eivät sisällä kohteita. YOLO keskittyy tällöin alueisiin, joiden ruudut antavat korkean todennäköisyyden mahdollisesta kohteesta ja myös ne ruudut, joilla on paljon yhteistä alaa, koska sama kohde voi olla merkitty usealla eri bounding boxilla. Tätä kutsutaan non-max suppressioniksi. (Świeżewski 2020)

Kuviossa 7 on vertailtu kuvantunnistusalgoritmien suorituskykyä. YOLOv4 tunnistusalgoritmi pärjäsi hyvin, kun verrattiin tarkkuutta AP suhteessa frame rate (FPS) nopeuteen. Tarkkuus laskee aina nopeuden noustessa. YOLOv4 oli vuonna 2020 kaksi kertaa nopeampi kuin tarkimmin tunnistava kilpailija EfficentDet miltei vastaavalla tunnistustarkkuudella, MS COCO aineistolla testattaessa. (Bochkovskiy, Chien-Yao, Hong-Yuan 2020)



KUVIO 7. YOLOv4 suorituskyky verrattuna kilpailijoihin vuonna 2020 (Bochkovskiy ym. 2020)

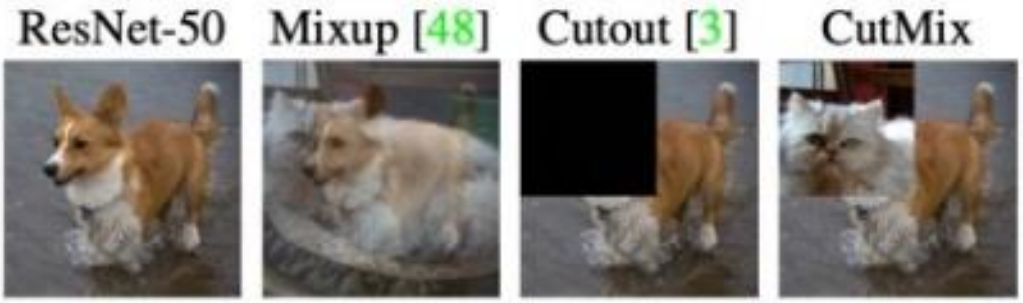
7.5 Kuvantunnistusalgoritmin rakenne

Moderni tyypillinen kuvantunnistusmalli on rakennettu useasta osasta, joita kutsutaan nimillä input, backbone, neck ja head. Ensimmäinen osa, input on tämän työn tapauksessa annotoitu opetusmateriaali. Seuraava osa on niin kutsuttu backbone, johon kuuluu, esimerkiksi ImageNet datasetillä valmiiksi koulutettu luokitin. YOLOv4:n backbonena toimii CSP Darknet53 niminen algoritmi. Tämän jälkeen on osa nimeltä neck, jossa on tyypillisesti kerroksia, jotka muun muassa keräävät ominaisuuskarttoja tunnistuksen eri vaiheista. Viimeisenä osana on head, jossa tapahtuu kohteen luokan ennustus sekä kohteen rajaus bounding boxilla. (Bochkovskiy ym. 2020)

7.6 Kuvan augmentointi

YOLO-algoritmiin ja muihin kuvapohjaisiin kuvantunnistusmenetelmiin kuuluu tunnistusta parantavia työkaluja, joita kutsutaan kuvien augmentaatioksi. Tarkoituksena on saada kasvatettua mallin ymmärrystä erilaisten syötekuvien ominaisuuksien vaihteluista, jotta mallista saadaan yleisesti parempi tunnistamaan eri lähteistä peräisin olevia ja vaihtelevan laatuisia kuvia. Esimerkiksi fotometrinen - ja geometrinen vääristymä ovat yleisesti käytettyjä kuvan augmentaatiomenetelmiä, jotka vahvistavat mallin häiriöiden sietokykyä tunnistuksia tehdessä. Fotometrinen vääristymä säätää kuvien kontrastia, kirkkautta, kylläisyyttä ja kohinaa. Geometrinen vääristymä tekee kuville satunnaista skaalausta, rajausta ja kuvan kääntelyä eri asentoihin. (Bochkovskiy ym. 2020)

Edellä mainitut metodit ovat vain pikselikohtaista muuntelua, joissa kaikki pikselidata kuitenkin pysyy mukana augmentoinnin jälkeen. Näiden lisäksi on myös muita augmentointimenetelmiä, jotka pilkkovat, peittävät tai yhdistelevät opetusmateriaalin kuvia toisiinsa. Esimerkiksi CutOut metodi tekee kuviin tyhjiä nolla-alueita, joka imitoi sitä, että kohde on jonkin toisen esineen takana sekä CutMix, joka tuo alkuperäisen kuvan päälle muita pienempiä kuvia. Kuvassa 13 on visualisoitu edellä mainittuja augmentointimenetelmiä. Mainitsemisen arvoinen on myös mosaic augmentation -metodi, jossa yhdistetään neljä eri opetusdatan kuvaa, joka mahdollistaa CutMixin kanssa kohteen tunnistuksen myös normaalin taustan ulkopuolella. (Bochkovskiy ym. 2020)



KUVA 13. Kuvien erilaisten augmentointien visualisointi (Hui 2020)

7.7 Koneoppimismallit ja niiden vertailu

Lopputuloksena datan keräämisestä ja valmistelusta ja järjestelystä, algoritmien valinnasta, koulutuksen suorittavan ohjelmistokokoonpanon rakennuksesta ja parametrien säädöstä sekä opetusajosta syntyy valmis koneoppimismalli. Malli on oma kokonaisuutensa ja se toimii opetuksen jälkeen ilman yhteyttä opetusaineistoon. Tässä työssä käyttövalmis malli koostuu luvussa 7.3 mainituista konfigurointitiedostoista sekä opetusajolla luodusta weights-tiedostosta. Malli ”osaa” toimia vain siinä käyttökohteessa, johon se on koulutettu. Koulutetun mallin hyvyyttä pitää pystyä myös arvioimaan ja vertaamaan. Miten tiedetään, kuinka tarkka malli on tunnistamaan opetettuja kohteita? Miten malleja voidaan verrata? On hyvä muistaa, että mallin todellinen tarkkuus, suorituskyky ja käyttökelpoisuus on arvioitava lopulta aina käyttötarkoituksensa mukaan.

7.7.1 MS COCO

Tietokonenäön ja kuvantunnistuksen alalla on useita käsitteeksi vakiintuneita, valmiiksi annotoituja datasettejä, joilla koulutettuja erilaisia tunnistusmenetelmien ja arkkitehtuurien tarkkuuksia ja nopeuksia voidaan vertailla. Tällaisia ovat muun muassa ImageNet, PASCAL ja MS COCO. Tarkastellaan esimerkkinä viimeiseksi mainittua. MS COCO lyhenne tulee sanoista Microsoft Common Objects in Context, jossa asiantuntijaryhmä on koonnut ja annotoinut valmiiksi kattavan sarjan kuvia, nimensä mukaisesti tavallisista, jokapäiväisistä kohteista. MS COCO sisältää 328 tuhatta kuvaa, 2,5 miljoonaa merkittyä kohdetta ja 91 eri luokkaa. (Belongie, S., Bourdev, L., Dollar, P., Girshick, R., Hays, J., Lin, T., Maire, M., Perona, P., Ramanan, D., Zitnick, C.L. 2015). Kyseinen MS COCO -datasetti on tämän työn Darknet-arkkitehtuurin yolov4-conv.137.weights-tiedoston aloituspainojen taustalla eli tässä työssä käytetäänkin tekoälyn koulutukseen siirto-oppimismenetelmää, koska taustalla on vanhemman tunnistusmallin painot. (Bochkocskiy, A. 2020)

7.7.2 Tarkkuuden mittarit: IoU, Precision, Recall ja mAP

Tekoälyn suorituskyvyn kuvaamisessa on kyse siitä kuinka paljon malli ennustaa oikein ja kuinka paljon väärin, verrattuna todelliseen tilanteeseen. Koskaan malli ei ole täysin sataprosenttinen ennustaja. Suorituskyvyn kuvaaminen aloitetaan luokittelemalla tulokset neljän eri kategorian alle. Tässä yhteydessä positive tarkoittaa tunnistettua ja negative hylättyä. (Kananen & Puolitaival 2019, 174-175)

TP / True Positive = oikein tunnistettu
FP / False Positive = väärin tunnistettu
TN / True Negative = oikein hylätty
FP / False Positive = väärin hylätty

Kuviossa 8 on havainnollistettu kaikki mahdolliset vaihtoehdot. Kuviossa on todellinen (actual) positive ja negative sekä ennustettu (predicted) positive ja negative. Malli jakaa tulokset tässä esimerkissä kahteen kategoriaan eli tunnistetuiksi ja hylätyiksi, jolloin mallia arvioidaan vertaamalla tuloksia oikeaan totuuteen. Kaikki mallin tekemät tulokset eivät ole aina oikein ja halutaan tietää, kuinka moni mallin antamista positiivisista ennusteista on oikeasti oikein ja kääntäen, kuinka moni mallin antamista negatiivisista ennusteista on oikeasti väärin.

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

KUVIO 8. Luokittelu TP, FP, FN, TN (Liu 2018)

Mallin sisäinen tarkkuus ilmaistaan kahdella tarkkuusarvolla: precision eli spesifisyys ja recall eli sensitiivisyys. Käsitteet true positive, false positive, true nega-

tive ja false negative ovat tärkeitä näiden tarkkuuksien laskemisessa. Precision mittaa kuinka suuri osuus mallin tekemistä positiivisista ennustuksista on todellisuudessa oikein. Laskukaava:

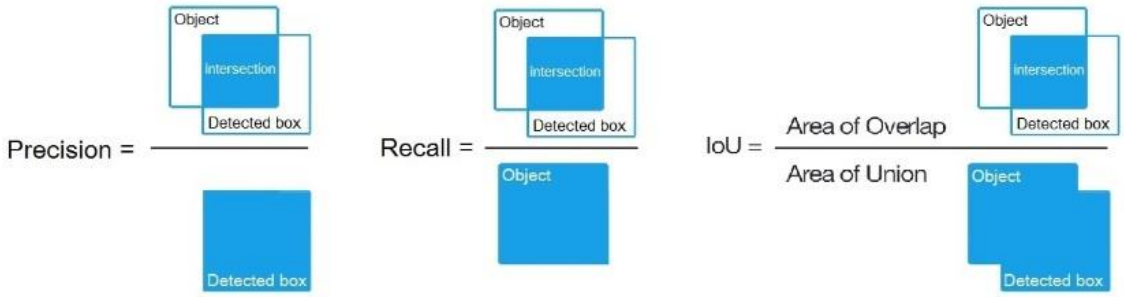
precision = TP / (TP + FP)

Recall puolestaan mittaa kuinka suuri osuus mallin tekemät positiiviset ennusteet ovat kaikista todellisista positiivisista kohteista. Laskukaava:

recall = TP / (TP + FN)

Jos pelkästään mallin precision on korkea, mutta recall matala, eivät mallin tekemät ennusteet ole tarkkoja. Myös korkea recall ja matala precision ei anna tarkkaa tulosta. Sekä precision että recall kuuluisi olla parhaassa tapauksessa korkeita ja tasapainossa toisiinsa nähden. Mittaria, joka ottaa nämä molemmat huomioon kutsuaan F1-scoreksi. (Kananen & Puolitaival 2019, 176-177)

Tärkeä käsite on myös IoU, joka on lyhenne sanoista Intersection over Union. Tällä tarkoitetaan todellisen kohteen ja ennustetun kohteen päällekkäistä alaa suhteessa todellisen kohteen ja ennustetun kohteen yhteiseen alaan (Bochkocskiy, A. 2020). Kuviossa 9 on havainnollistettu nämä edelliset käsitteet.



KUVIO 9. Precision, Recall ja IoU (Bochkocskiy, A. 2020)

IoU:n tulkinnalle voidaan antaa kynnysarvo, kuten esimerkiksi 0,5. Jos ennusteen IoU tulos on 0,7, ylittää se annetun kynnysarvon, jolloin tunnistus tulkitaan True Positiveksi. Mikäli ennusteen IoU tulos on 0,3 eli bounding box osuu väljästi oikean kohteen päälle ja tulos alittaa annetun kynnysarvon, tulkitaan tun-

nistus tällöin False Positiveksi. Jos sovelluskohteeseen riittää väljempi tarkkuus voidaan kynnysarvoa pienentää. Precision ja Recall tuloksia tulkittaessa onkin tärkeää tietää kyseiseen tapaukseen määritetty IoU:n kynnysarvo. (Yohanandan 2020)

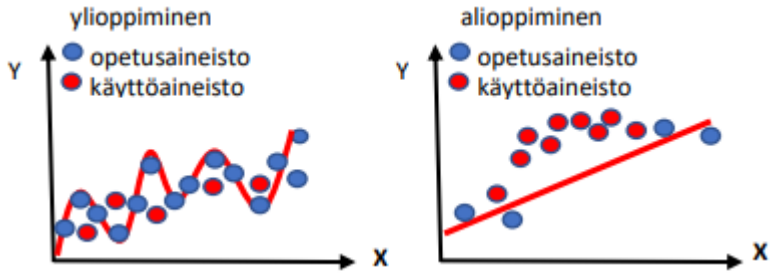
Yksi nykyisin määrittävimmistä ja suosituimmista tarkkuuden yksiköistä kuvantunnistuksessa on AP eli average precision ja siitä johdettuna mAP eli mean average precision. Monet yleiset kuvantunnistusalgoritmit, kuten Faster R-CNN, YOLO ja MobileNet SSD, käyttävät mAP vertailuarvoa, kun tutkimustuloksia julkaistaan. Usein näitä kahta AP ja mAP käytetään synonyymeinä, mutta riippuen onko kyseessä PASCAL- vai COCO-kuvantunnistuskilpailu, voi mAP-arvon laskennassa olla eri kaava. Määritelmänä on kuitenkin, että average precision (AP) lasketaan jokaiselle tunnistettavalle luokalle erikseen ja mAP on näiden luokkakohtaisten keskiarvojen keskiarvo, joka ottaa huomioon myös kyseisessä tapauksessa määritetyn IoU-kynnysarvon. Eli mAP ei ole vain yksinkertaisesti tarkkuuden precision keskiarvo. (Yohanandan 2020)

Darknet käyttää myös määrettä average loss eli keskimääräinen virhe, joka kuvaa mallin tunnistuksen suorituskykyä. Kun mallia koulutetaan ohjatun oppimisen menetelmällä, algoritmit pyrkivät löytämään matemaattisen mallin, joka minimoi tunnistuksen virheen. Mitä suurempi average loss on, sen epätarkempi malli on. (Google Machine Learning Education. n.d)

7.7.3 Yli- ja alioppiminen

Molempiin sekä luokitteluun että regressioon liittyy koneoppimisen riski mallin ylioppimiselle tai alioppimiselle (engl. overfitting ja underfitting). Yksinkertaisesti ylioppiessaan malli sovittautuu liian orjallisesti opetusmateriaaliin ja epäonnistuu tällöin muun materiaalin tunnistamisessa tai ennustamisessa. Alioppiminen taas tarkoittaa sitä, että opetusmateriaali ei kuvaa tarpeeksi hyvin mallin kaikkia mahdollisia käyttötilanteita. Kuviossa 10 esitettynä edellä mainitut käsitteet. (Pietikäinen & Silvén 2019, 75)

.



KUVIO 10. Yli- ja alioppiminen (Pietikäinen & Silvén 2019, 87)

7.7.4 Mallin kehittäminen paremmaksi

Koneoppimismalliin voi joutua ensimmäisen mallin valmistumisen jälkeen tekemään parannuksia. Mallin kehittämiseen on useita tyypillisiä toimia, kertoo Kananen ja Puolitaival (2019, 182–183). Ensimmäinen keino tarkentaa ohjattua oppimista on lisätä laadukasta opetusmateriaalia mallin rakennukseen. Mitä enemmän esimerkkitapauksia mallille opetetaan, sen paremmin se suoriutuu tunnistustehtävistä. Vaihtoehtoisesti voi kokeilla jotain toista koneoppimisalgoritmia, joka myös soveltuu kyseisen sovelluksen käyttötarkoitukseen. Neuroverkkotyyppistä mallia voi kehittää luomalla entistä suuremman ja syvemmän neuroverkon sekä käyttämällä dropoutia tai regularisointitekniikoita. Nämä ovat teknisia toimenpiteitä, joilla esimerkiksi data engineer muuttaa mallin toimintaa, kun haetaan parempaa tunnistustarkkuutta.

Kuvantunnistustehtävissä tunnistustarkkuuteen auttaa myös mallin sisäisen konfiguroinnin kautta kuvien lukemisen resoluution muuttaminen suuremmaksi, joka vaatii enemmän suorituskykyä tietokoneelta tai enemmän aikaa mallin opetusajoon. Yksi keino on myös lisätä opetusmateriaaliin samanlaisia kuvia, joissa ei ole kohteita. Tällöin malli oppii paremmin, mitkä alueet kuvissa eivät ole haluttuja kohteita. Kuvilla opetetuissa malleissa tärkeää tarkkuuden kannalta on myös oikeanlainen, tyyliltään täsmällinen ja tarkka opetuskuvien kohteiden labelointi. (Bochkocskiy, A. 2020).

8 TULOKSET

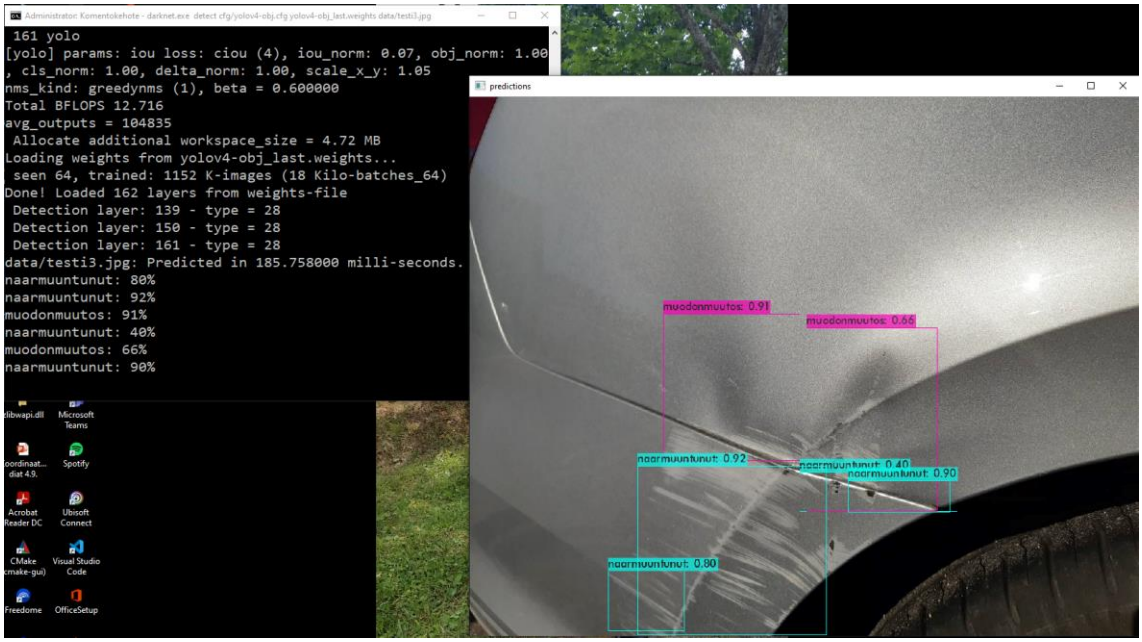
8.1 Ensimmäinen malli

Työssä toteutettiin kaksi eri mallia samalla datalla kahdella eri tietokoneella. Ensimmäinen malli oli harjoitus saada Darknet-arkkitehtuuri toimimaan omalla koti-PC:llä ja saada ymmärrys tarvittavista ohjelmista ja kokoonpanosta. Toinen malli toteutettiin Tampereen ammattikorkeakoulun koneoppimiskuormiin tarkoitetulla Legion -tietokoneella samalla arkkitehtuurilla. Molempien mallien opetusajo kesti noin 35-40 tuntia.

Ensimmäinen malli osoittautui hyvin rohkaisevaksi puutteineenkin ja toimi paljon odotettua paremmin. Ensimmäisellä mallilla kokeiltiin tunnistaa vain kourallinen testikuvia, koska tarkoitus oli siirtyä tekemään varsinainen toinen malli tehokkaammalla tietokoneella heti kun ensimmäinen malli saadaan välttävästi toimimaan. Tärkeintä oli, että mallin opetus onnistui ja tunnistus toimi niin kuin oli tarkoitettu ja, että tunnistusarkkitehtuurin kokoonpano oli saatu todistettavasti toimimaan. Malli siis oppi, mutta ei ylioppinut, joka on pienellä opetusmateriaalilla vaarana. Ensimmäisen mallin tunnistuksen puutteet resoluution pienuuden takia kävivät kuitenkin selviksi jo ensimmäisissä kokeiluissa, joten nopeasti siirryttiin seuraavan mallin rakennukseen. Vanhat vauriokuvat kuitenkin selvästi toimivat opetusmateriaalina tähän tekniikkaan hyvin.

Seuraavaksi esitetään esimerkkejä ensimmäisen mallin tekemistä tunnistuksista, kun sille annettiin käsiteltäväksi uusia, ennestään näkemättömiä vauriokuvia. Ensimmäinen malli antoi hyviä tunnistustuloksia varsinkin lähikuvissa oleviin vaurioihin, kuten nähdään kuvasta 14 ja 15.

Kuvassa 14 on auton oikean takalokasuojan ja takapuskurin päädyn vaurio, josta tunnistin havaitsee lommot muodonmuutoksina sekä naarmuiset alueet naarmuina aivan oikein. Tunnistin havaitsee siis vaurion sijainnit sekä myös vauriotyypit oikein. Lommojen koko ei aivan rajaudu silti oikein eli varsinkin vasemmanpuoleinen muodonmuutoskohde on oikeasti suurempi kuin bounding box kuvasta osoittaa.

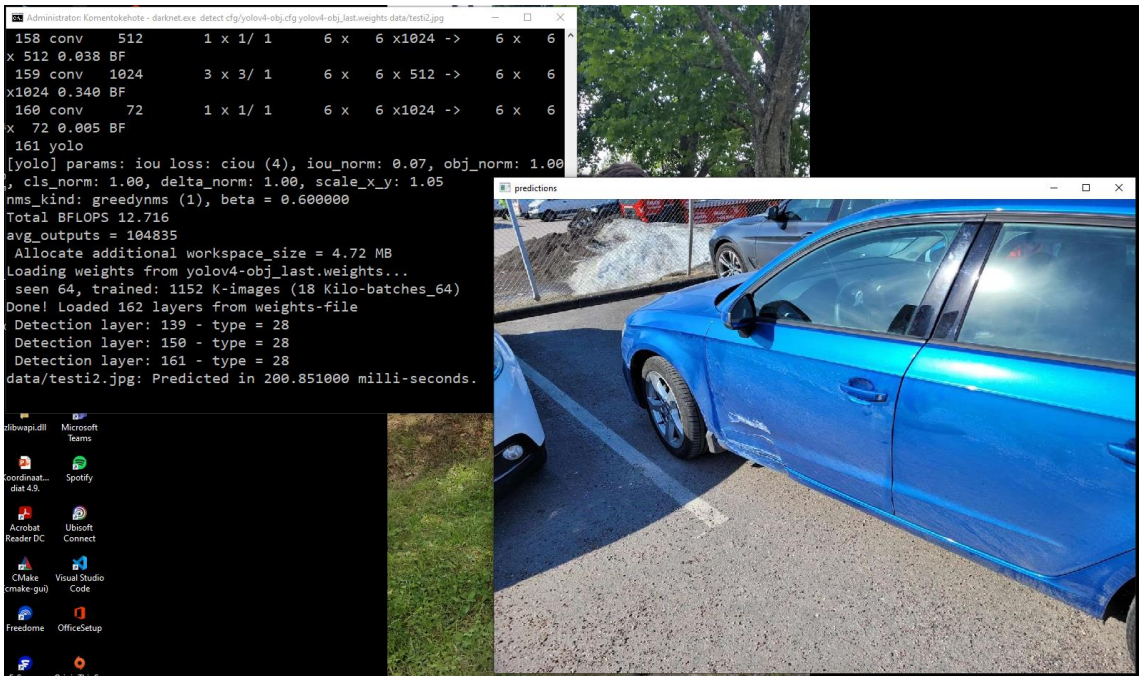


KUVA 14. Oikean takakyljen vaurion tunnistus

Kuvassa 15 näkyy auton tunnistus kuvasta eli ”car”-luokka. Tunnistin löytää myös oikein naarmuja sekä muodonmuutoksen, joka on liian suuri oikeaan lokasuojan lommoon nähden, mutta silti oikealla kohdalla. Joskus työpaikan kuvissa on mukana myös niihin lisättyjä merkintöjä niin kuin tässä kuvassa näkyvä musta suorakaide, mutta se ei ole häirinnyt tunnistusta.

KUVA 15. Oikean etukulman vaurioiden tunnistus

Kuvassa 16 huomataan ensimmäisen mallin heikkous eli pieni käsittelyresoluutio. Kun vaurio on kaukana ja pienenä kohteena kuvassa ei resoluutio enää riitä tunnistamiseen ja vaurio jää löytymättä auton etuoven alakulmasta.

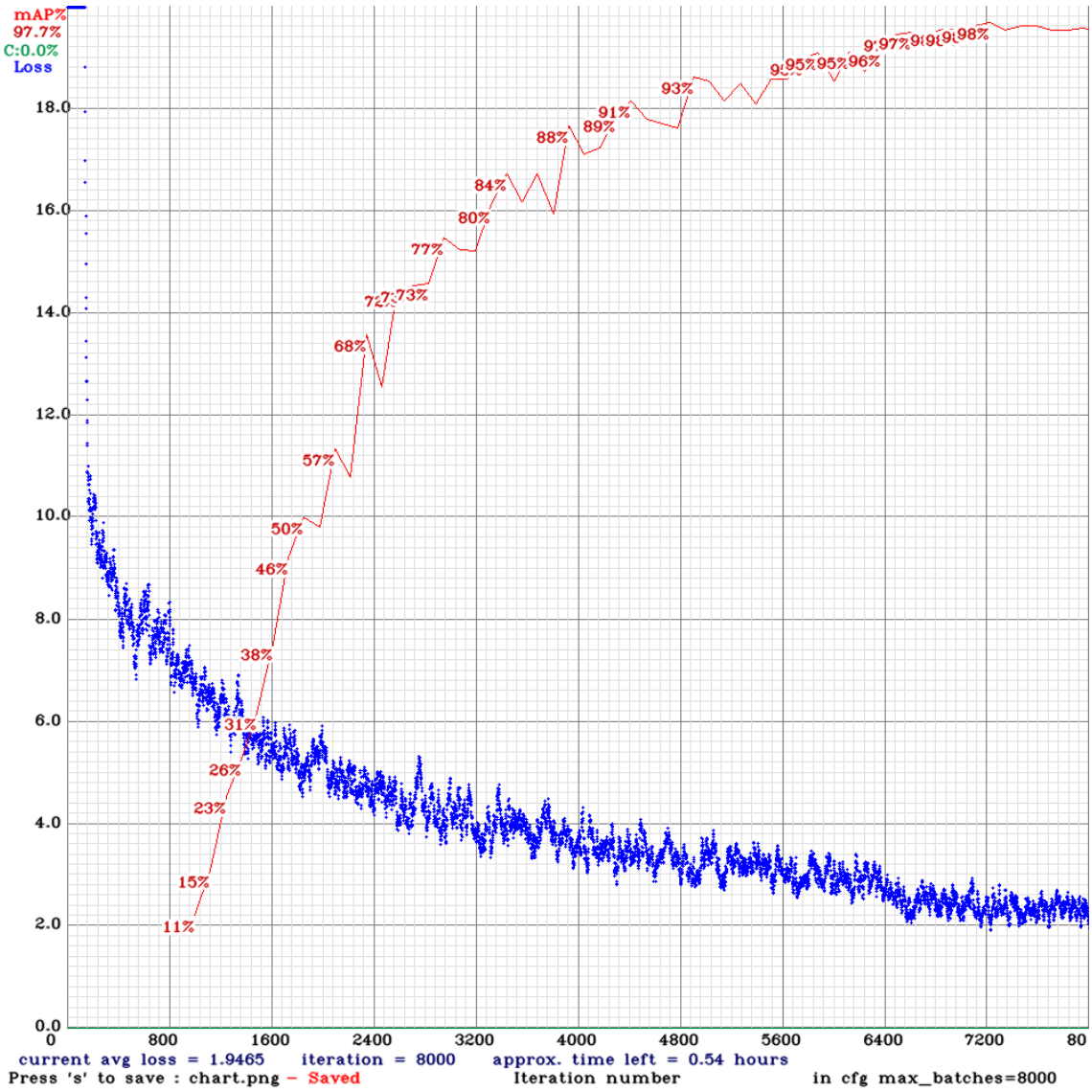


KUVA 16. Vaurio ei löytynyt kuvasta

8.2 Toinen malli

Toisen mallin testeistä havaittiin, että tunnistukset olivat huomattavasti parempia verrattuna ensimmäiseen malliin. Odotetusti juuri pienemmät ja kuvissa kauempana olevat vauriokohteet tunnistettiin varmemmin. Vahvuuksina mallilla on, että miltei aina kuvassa oleva vauriokohta löytyi ja suurimmaksi osaksi myös tehdyn tunnistuksen luokka oli oikein. Toisen mallin tunnistuksissa oli myös virheitä. Yleisimmät virheet olivat, että vauriota ei havaittu tai bounding box on kohdistettu epätarkasti. Myös tietyt auton pinnasta heijastavat taustalla olevat kohteet voivat muistuttaa opetettuja vaurioita ja voivat saada aikaan virheellisiä tunnistuksia. Silti malli oli selvästi oppinut ja käytetty tunnistustekniikka toimi jo näinkin pienellä opetusmateriaalin määrällä suhteellisen hyvin.

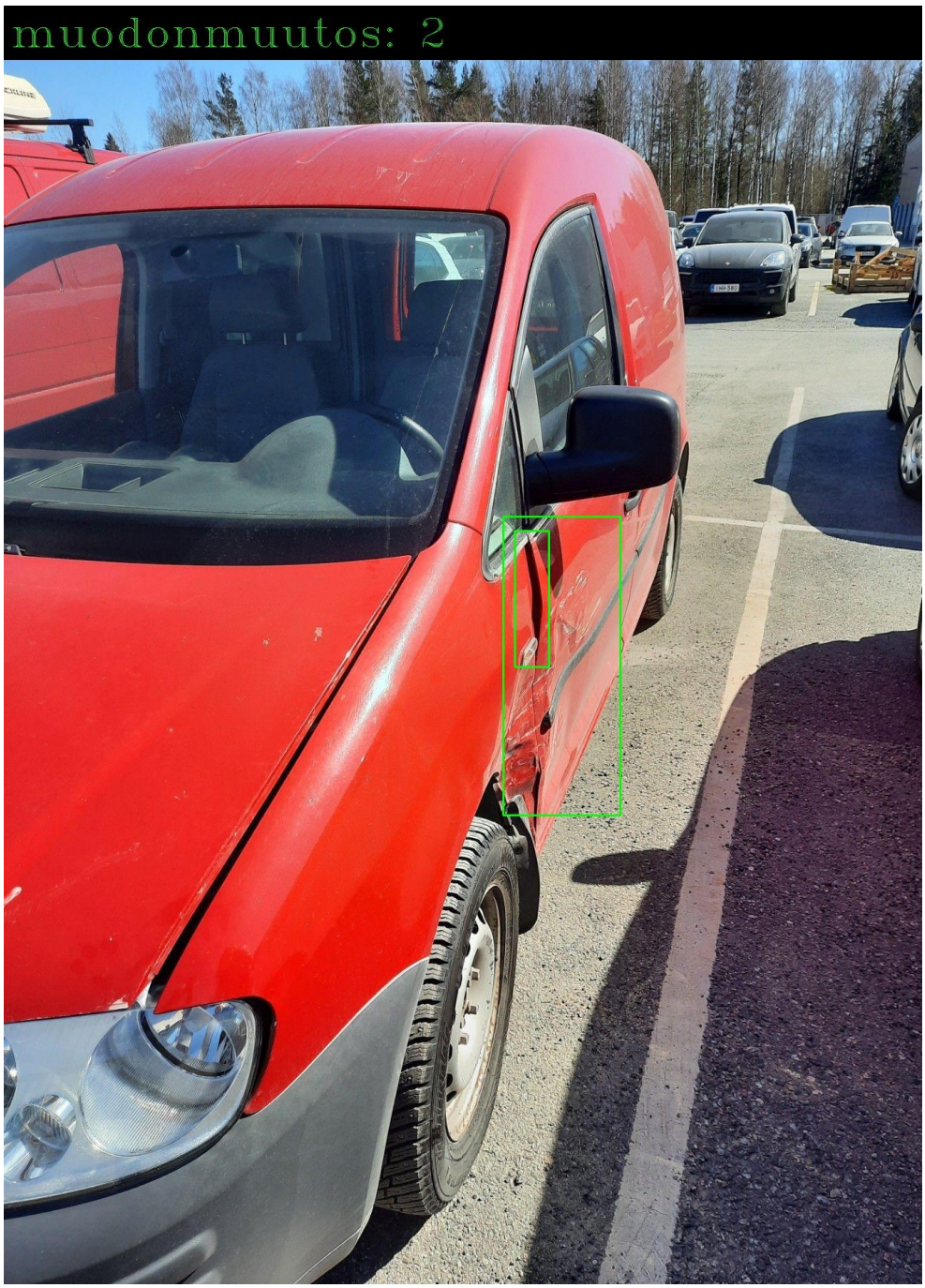
Kuviossa 11 on tämän työn toisen mallin koulutuksen average loss- ja mAP-kuvaajat. Kuvaajan x-akselilla on opetuksen laskentakierrosten määrä ja y-akselilla prosentuaalinen virhe. Sininen kuvaaja visualisoi virheen pienenemistä ja punainen kuvaaja esittää kuinka mAP-arvo nousee opetuksen edetessä. Molemmat arvot kehittyivät oikealla tavalla opetuksen aikana. Kyseisellä hetkellä 8000:nen laskentakierroksen kohdalla average loss oli painunut hieman alle 2%:n. Korkein mAP-arvo 98% saavutettiin 7200 laskentakierroksen kohdalla.



KUVIO 11. Average-Loss- ja mAP- kuvaajat toisen mallin opetuksen edetessä

Seuraavana esitetään tunnistuksia, joita toisella mallilla on tehty. Toisella mallilla ajettiin läpi yhteensä 600:n uuden testikuvan erä, joista poimin erilaisia esimerkkejä tähän raporttiin. Toinen malli toimi paremmin kuin ensimmäinen ja tulosesimerkkejäkin käydään läpi siksi enemmän.

Kuvassa 17 malli löytää vaurion, vaikka kuvassa vaurio näkyy vain pienestä kulmasta auton kyljessä. Vauriotyyppi on tunnistettu oikein ja miltei rajattu myös oikein. Kuvasta olisi voinut lisäksi havaita konepellin huonon istuvuuden lokasuojaan nähden, jonka esimerkkejä on opetettu osa ei paikoillaan -luokkaan.

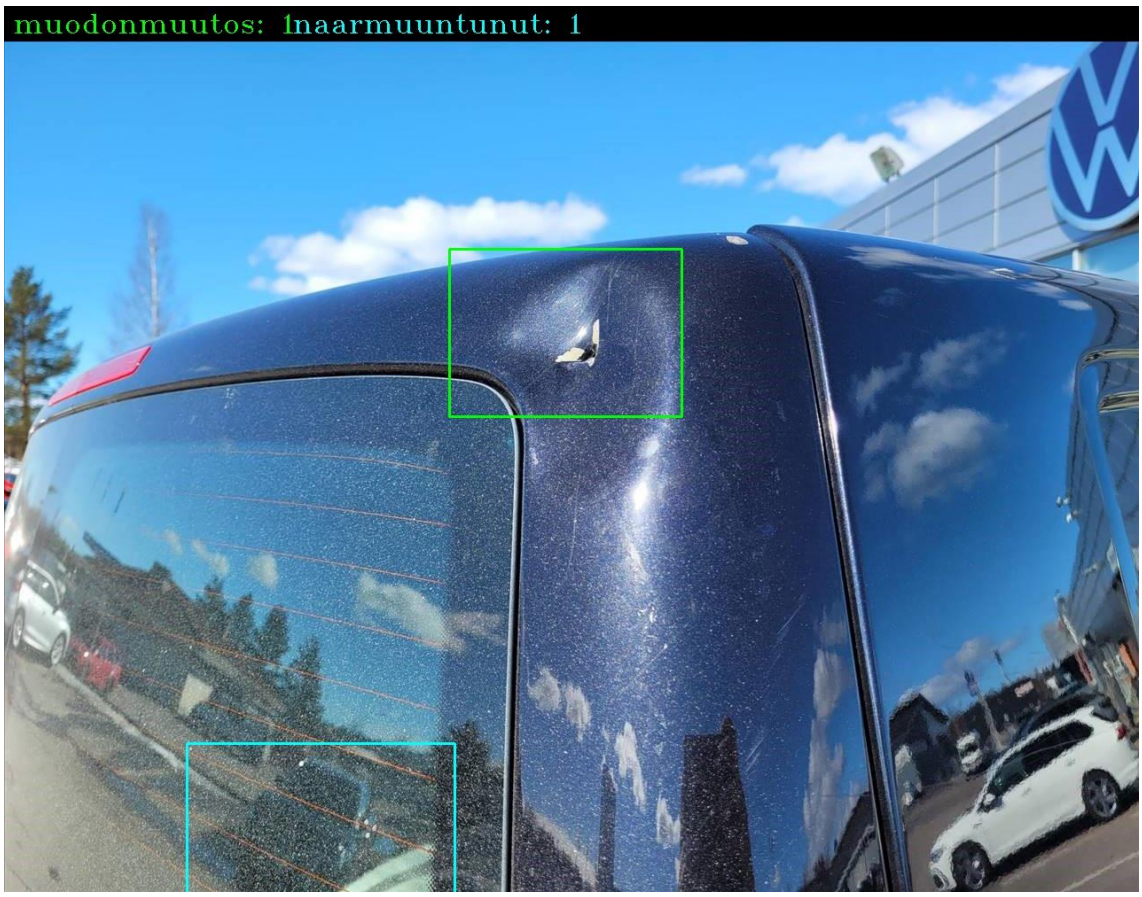


KUVA 17. Vaurio löytyi haastavasta kuvasta

Kuvassa 18 on sekä oikein tunnistettu halkeama, mutta myös väärin tunnistettu osa ei paikoillaan -kohde. Valaistuksesta johtuva vahva varjo rajana sumuvalon syvennyksessä todennäköisesti luo vaikutelman irrallisesta osasta. Opetusaineistossa vastaavia vahvan valon varjoja ei ehkä ollut tarpeeksi tätä virheellisen tunnistuksen mahdollisuutta pienentämässä.

KUVA 18. Oikea ja väärä tunnistus samassa kuvassa

Kuvassa 19 tunnistin löytää myös kaksi kohdetta, joista toinen on oikein tunnistettu muodonmuutos, mutta toisessa kohteessa malli luulee ilmeisesti takalasin lämmitysvastuslankoja naarmuiksi. Muodonmuutoksen rajaus on puutteellinen, koska lommo jatkuu merkityn alueen ulkopuolelle.

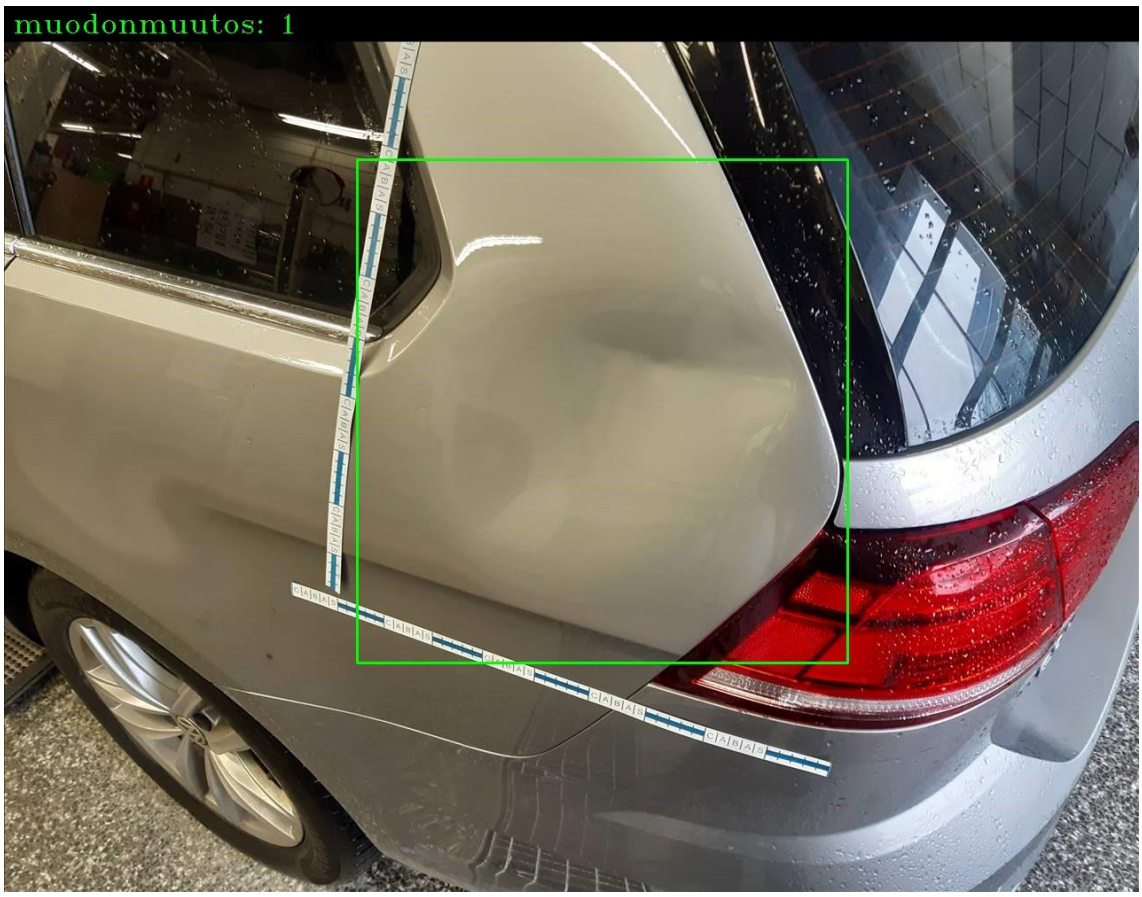


KUVA 19. Virheellinen kohde sekä oikein tunnistettu kohde

Kuvassa 20 on auton kylki, jossa on samanlaiset viistetyt osumat sekä auton etu- että takaovessa, josta malli havaitsee molemmat, mutta antaa vaurioiden samankaltaisuudesta huolimatta kohteille eri luokat: takaovessa havaitaan muodonmuutos ja etuovessa naarmuuntuminen. Molemmissa ovissa onkin selvästi oikeasti molemmat vaurioluokat eli muodonmuutos, jonka keskellä on poikittaissuuntainen naarmuvaurio. Tässä oikea tulos olisi ollut se, että molemmista ovista tunnistettaisiin molempien luokkien vauriot. Opetusdatassa ei ole ollut tarpeeksi vastaavia esimerkkitapauksia, koska näin samankaltaiset vauriot voidaan tulkita eri luokkiin kuuluviksi.

KUVA 20. Samankaltaiset vauriot luokiteltu eri luokkien vaurioiksi

Kuvassa 21 on Golfin takalokasuojassa suuri painauma, jonka malli havaitsi hyvin vaikka kohteen ääriviivat eivät ole kovin selvät. Sekä tunnistuksen luokka että bounding box kohdistettiin oikein. Kuvassa näkyvät magneettimittanauhat eivät häirinneet tunnistusta.

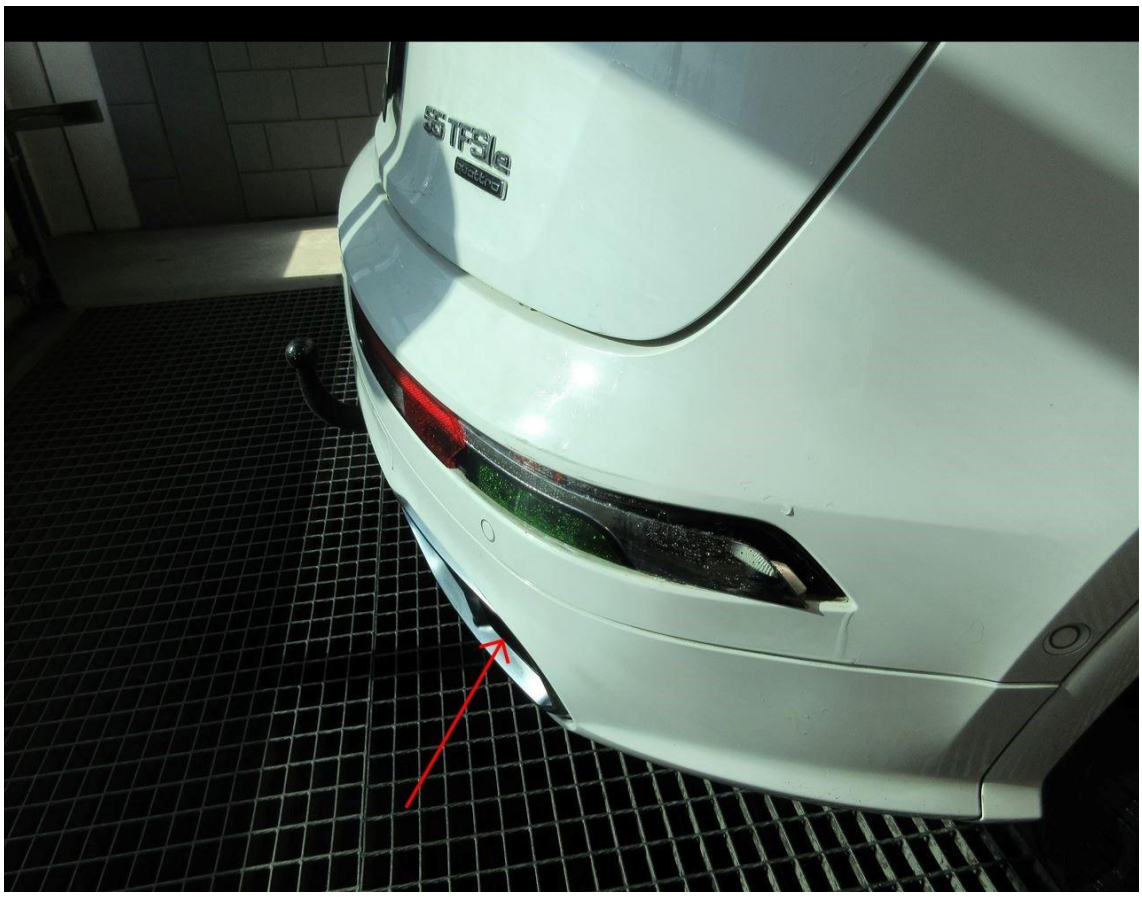


KUVA 21. Muodonmuutos tunnistettu Golfin takalokasuojasta

Kuvassa 22 tunnistetaan opetetusti rikkinäinen takavaloumpio, mutta vieressä olevaa muodonmuutosta ei havaita. Raitataululla peilatessa lommo näkyy hyvin ihmissilmälle, mutta johtuen juuri tällaisen opetusdatan puutteesta ei lommoa havaita tässä tapauksessa peilauksesta. Lommo olisi vaikea havaita ihmissilmälläkin kuvasta ilman peilausta raitataululla. Lisäopetusdatan avulla tämän tyyppisetkin peilauksesta erottuvat lommot havaittaisiin paremmin.

KUVA 22. Kaikkia kuvassa näkyviä vaurioita ei tunnisteta

Kuvassa 23 selvimmin erottuva vaurio on puskurissa olevan valoumpion vaurio, joka olisi oikein tunnistettuna kuulunut osa ei paikoillaan -luokkaan, mutta nyt malli ei sitä jostain syystä havaitse ollenkaan. Mahdollisuutena olisi ollut myös havaita umpion yläpuolella puskurin reunan naarmuuntuminen, mutta tätäkään ei havaittu. Harvinaisen muotoinen auton perän alaosan lisävaloumpion vaurio on harvinaisuus opetusaineistossa. Lisäopetusmateriaali rikkinäisistä umpioista auttaisi tähän tunnistusongelmaan.

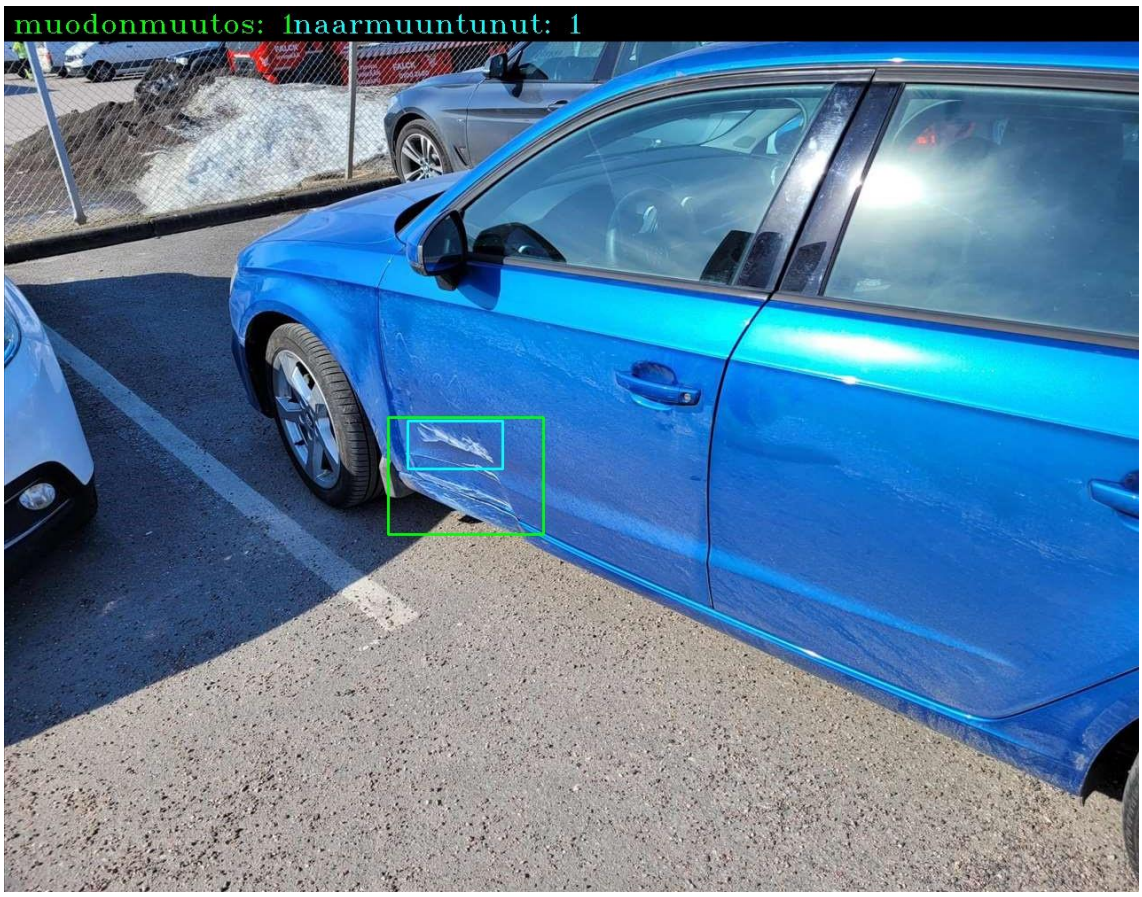


KUVA 23. Vaurioita ei havaittu

Kuvassa 24 Passatin vasemman takakyljen laaja naarmuuntuminen on havaittu opetetusti. Naarmuja opetettiin samaan vaurioluokkaan sekä yksittäisinä naarmuina, että laajempina naarmuuntuneina alueina, joista tämä osuu jälkimmäiseen kategoriaan. Merkityn alueen ulkopuolelle jää myös havaitsematta naarmuuntunut -luokan vaurioita oven kahvasta, oven alaosasta sekä takalokasuojan kaaresta. Kohdistus naarmuuntuneen pinnan koko alueelle jää vajaaksi, koska rajaus on tehty keskelle vauriota oven ja takalokasuojan rajaan, vaikka vauriojäljet jatkuvat takalokasuojassa. Toiseen suuntaan merkitty naarmualue taas jatkuu liian pitkälle todennäköisesti peilauksessa näkyvän parkkipaikan reunaviivan takia, joka muistuttaa paljon naarmua.

KUVA 24. Passatin kyljen naarmuvaurion tunnistus

Kuva 25 on sama kuva kuin ensimmäisen mallin tunnistettavana oli, mutta tulokset paremmat: tarkemmalla resoluutiolla opetettu toinen malli löytää vauriokohdan autosta ja tunnistaa kahden luokan vauriot oven etukulmasta, jotka ovat aivan oikein. Myös vaurioiden paikannus on onnistunut tarkasti. Auto on lisäksi myös likainen, mutta se ei haittaa tässä tapauksessa tunnistusta.



KUVA 25. Tarkempi malli tunnistaa vaurion kauempaakin

9 POHDINTA

9.1 Johtopäätökset

Tämän opinnäytetyön ajatuksena oli hyödyntää yrityksellä jo olemassa olevaa dataa uudella tavalla. Tutkimuskysymyksenä oli, voidaanko yrityksen vanhaa kuvadataa hyödyntää vauriotunnistimen opetuksessa? Johtopäätös vauriotunnistus mallien kokeiluista on, että arkistossa olevia vanhoja vauriokuvia voidaan käyttää vaurioita tunnistavan mallin opetukseen tätä tekniikkaa hyödyntäen. Soveltuvuusselvitys-tyyppinen kokeilu antoi hyvin vastauksen tutkimuskysymykseen. Laajemmalla opetusaineistolla ja reilusti suuremmalla määrällä opetettavia vaurioluokkia on mahdollista rakentaa kattava ja luotettava koneoppimismalli, jolla yleisimmät autojen kolarivauriot pystyttäisiin tunnistamaan mistä tahansa vaurioauton kuvasta tai videosta riittävällä tarkkuudella.

Pidemmälle vietynä uskon, että vauriokorjaamon vahinkotarkastusta pystyttäisiin kehittämään useammallakin vastaavalla ohjatun oppimisen tekoälysovelluksella. Ehkä esimerkiksi myös alustavat korjauskustannusarviot pystyttäisiin laskemaan riittävällä tarkkuudella automaattisesti ja etänä asiakkaan kuvien pohjalta. Jatkotutkimuksena voisikin opetusdataan ottaa mukaan vauriotapauskohtaiset korjauskustannusarviolaskelmat, joiden avulla ehkä myös pintaa syvemmällä olevat vauriot pystyttäisiin riittävän suuren datamäärän avulla saada ennustettua koneoppimismallin avulla jo vauriokuvan perustella, ilman osien purkamista. Riittävän tarkka malli tarvitsisi arviolta vähintään satojatuhansia, ellei jopa miljoonia vauriolaskelmia ja niihin liittyviä kuvia, koska eri automalleja, varaosia ja korjaustoimenpiteitä on paljon.

9.2 Tekniikan edut ja haasteet

Tämä tekniikka avaa uudenlaisia mahdollisuuksia monenlaisissa käyttökohteissa, joissa tarvitaan kuvista tai videoista tehtävää kohteiden tunnistusta. Kehittyvät algoritmit tekevät tekniikasta koko ajan nopeamman ja tarkemman. Suuremmat laskentatehot mahdollistavat mallien opetuksen nopeammin ja myös

uudet tarkemmat kamerat antavat mahdollisuuden luoda konenäöstä jatkossa entistä tarkemman.

Avainasemassa koko projektin onnistumisen kannalta on riittävä opetusdatan määrä ja laatu sekä varsinkin annotointien laatu. Suurimpia haasteita tällaisessa projektissa on juuri saada koottua tarpeeksi suuri opetusmateriaali, johon esimerkit on merkitty. Iteratiivisesti tehtynä annotointi on aikaa vievää. Laadukas kaupallinen malli, joka osaa lukemattomia luokkia ja tunnistaa luotettavasti erinäköisiä vauriokohteita vaatii annotoitua dataa todella paljon. Datan kerääminen ja tarkka annotointi pitäisikin ehkä yhdistää suoraan työprosessiin jo dataa luodessa.

Työtä tehdessä haasteena oli myös annotointien tekeminen ajatuksella ja vaurioiden jakaminen tarkkoihin luokkiin. Esimerkiksi vauriotyyppi naarmuuntunut laajeni annotointeja tehdessä monenlaisiin erityyppisiin pintavaurioihin, joka todennäköisesti hajaannuttaa tuloksia, koska erilaiset pintavauriot voivat poiketa toisistaan huomattavasti ja tällöin opetusesimerkkejä on vähemmän kunkin näköistä vauriota kohden. Uuteen malliin tekisin annotoinnin niin sanotusti tiukemmalla kurilla, eli tarkka annotoinnin rajaus pelkästään täsmälleen yhdenkaltaisiin vaurioihin per luokka ja tehdä tällöin enemmän vaurioluokkia, jos halutaan kaikenlaisia vaurioita tunnistaa.

Annotoinnissa haastava asia on myös se, että kuvassa voi olla useampi eri luokan vaurio päällekkäin, jolloin samaan kohteeseen pitäisi merkitä miltei samoilla ääriviivoilla kaksi eri luokan kohdetta. Todennäköisesti tämä olisi se oikea tapa tehdä tämänlaisten vaurioiden annotointi, koska malli oppii, että näiden luokkien vauriot voivat sijaita kuvassa niin sanotusti päällekkäin.

Etuna yritykselle on se, että mikäli opetukseen kelpaavaa dataa löytyy valmiiksi, ei sitä tarvitse ostaa ulkopuolelta. Mielekästä monelle yritykselle olisi jo nykyään pohtia, millaista potentiaalisesti arvokastakin dataa arkistoissa lojuu hyödyntämättömänä.

Haasteita aiheuttivat myös suomenkieliset termit tätä työtä tehdessä. Eri lähteissä englanninkielisten alan termien suomennokset voivat vaihdella ja käsit-

teet sekoittua keskenään. Myös englanninkielisiä alan termejä käytettiin monissa lähteissä ristiin. Virallisen termistön luonti olisi hyvä kehitysaskel suomenkieliselle tekoälyn kuvantunnistuksen kentälle.

9.3 Kohti parempaa mallia

Projektin aikana kirjallisuus sekä kokeilut auttoivat ymmärtämään kuvantunnistusmallin toiminnan periaatteita. Ensimmäinen parannus olisi kasvattaa annotoitua opetusaineistoa. Ymmärrys opetusaineiston määrästä kehittyi projektin edetessä huomattavasti. Lopulta juuri mallin käyttötarkoitus ja tavoite ratkaisee opetusaineiston määrän riittävyyden. Mitä enemmän tunnistettavia luokkia on, sen enemmän tarvitaan pohjaksi opetusdataa, jotta kaikenlaiset erilaiset mahdolliset esimerkkitapaukset esiintyvät siinä riittävästi. Darknet-ohjeistuksen mukaan tarkkuutta parantaa huomattavasti se, että jokaista tunnistettavaa luokkaa varten olisi vähintään 2000 erilaista kuvaa yhden tunnistettavan luokan kohteista. Vaikka kuvia tämän työn opetusaineistossa oli yhteensä vain noin 2000, opetusaineistoon merkittyjä kohteita oli moninkertainen määrä.

Mukaan opetusdataan olisi hyvä saada myös vauriokohteista tyhjiä kuvia eli kuvia ehjistä autoista ja niiden osista. Darknetin mukaan tämä opettaisi mallille samankaltaisista kuvista kohteita, joita ei haluta tunnistaa, lisäten taas mallin osaamista. Näitä kuvia ei K-Auton arkistossa suuremmin ole, vaan tarkempaa mallia varten niitä pitäisi manuaalisesti ottaa. Tämä ei kuitenkaan ollut tässä työssä tutkittavana vaan tarkoitus oli hyödyntää arkistossa jo olevaa kuvadataa.

Kontrolloitu tausta opetuskuvissa voisi kasvattaa tunnistustarkkuutta eli esimerkiksi valaistusolosuhteet eivät vaihtelisi niin paljon. Nyt oli mukana erilaisissa sisävalaistuksissa otettuja kuvia sekä myös ulkona erilaisissa ympäristöissä ja jopa eri vuoden aikoina otettuja kuvia. Myös kuvakulma vaihtelee paljon erilaisten vaurioiden kuvissa, mutta niin kauan kuin ihminen kuvaa on kuvakulmiakin lukematon määrä.

Tuloksia parantaisi opetusdatan balansointi eli kaikkien tunnistettavien luokkien esimerkkejä pitäisi olla tasan sama määrä opetusdatassa (Pietikäinen & Silvén

2019, 100). Tässä työssä tämä tarkoittaisi, että jokaisesta vaurioluokasta olisi sama määrä annotoituja esimerkkikohteita.

LÄHTEET

Belongie, S., Bourdev, L., Dollar, P., Girshick, R., Hays, J., Lin, T., Maire, M., Perona, P., Ramanan, D., Zitnick, C.L. 2015. Microsoft COCO: Common Objects in Context. Arxiv. Luettu 13.9.2022.

Bochkocskiy, A. 2020. YOLOv4 / Scaled-YOLOv4 / YOLO - Neural Networks for Object Detection (Windows and Linux version of Darknet). GitHub. Päivitetty 30.10.2021. Viitattu 8.3.2022. https://github.com/AlexeyAB/darknet

Bochkovskiy, A., Chien-Yao, W., Hong-Yuan, M. 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection. Arxiv. https://arxiv.org/pdf/2004.10934.pdf

CMake. 2022. Build with CMake. Build with Confidence. Verkkosivu. Viitattu 12.5.2022. https://cmake.org/

Coral. 2020. Retrain an object detection model. Internet-sivusto. Luettu 20.8.2022. https://coral.ai/docs/edgetpu/retrain-detection/#requirements

Elements of AI. 2022. Miten tekoäly määritellään? Verkkokurssi. Viitattu 24.5.2022.

Euroopan parlamentti. 2021. Mitä tekoäly on ja mihin sitä käytetään? Verkkosivu. Viitattu 24.5.2022. https://www.europarl.europa.eu/news/fi/headlines/society/20200827STO85804/mita-tekoaly-on-ja-mihin-sita-kaytetaan

Google Machine Learning Education. n.d. Descenting into ML: Training and Loss. Verkkosivu. Viitattu 5.10.2022. https://developers.google.com/machine-learning/crash-course/descending-into-ml/training-and-loss

Hui, J. 2020. YOLOv4. Internet artikkeli. Julkaistu 4.5.2020. Viitattu 5.10.2022. https://jonathan-hui.medium.com/yolov4-c9901eaa8e61

Kananen, H. & Puolitaival, H. 2019. Tekoäly – bisneksen uudet työkalut. Helsinki: Alma Talent.

Karimi, G. 2021. Introduction to YOLO Algorithm for Object Detection. EngEd Community. Luettu 30.3.2022. https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/

Lin, T. 2018. About LabelImg. GitHub. Päivitetty 3.12.2018. Viitattu 15.3.2022. https://github.com/tzutalin/labelImg

Liu, Y. 2018. The Confusing Metrics of AP and mAP for Object Detection / Instance Segmentation. Internet artikkeli. Julkaistu 28.10.2018. Luettu 9.9.2022. https://yanfengliux.medium.com/the-confusing-metrics-of-ap-and-map-for-object-detection-3113ba0386ef

Louridas, P. 2021. Algoritmit. Helsinki: Terra Cognita

Maindola, G. 2021. A Brief History of YOLO Object Detection Models From YOLOv1 to YOLOv5. MLK. https://machinelearningknowledge.ai/a-brief-history-of-yolo-object-detection-models/

Microsoft. 2022. It's how you make software. Verkkosivu. Viitattu 12.5.2022. https://visualstudio.microsoft.com/

Michelucci, U. 2019. Advanced Applied Deep Learning. Fundamentals of Convolutional Neural Network. E-kirja. Berkeley, Apress. Viitattu 11.7.2022. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/advanced-applied-deep/9781484249765/html/470317_1_En_3_Chapter.xhtml

Mätäsjärvi, K. 2016. Korikorjaamon vahinkotarkastustoiminnan kehittäminen. Tuotantotalouden koulutusohjelma. Lapin ammattikorkeakoulu. Opinnäytetyö. Viitattu 15.3.2022. https://www.theseus.fi/bitstream/handle/10024/120429/Matasjarvi_Kalle-Markus.pdf?sequence=1&isAllowed=y

Nelli, F. 2018. Python Data Analytics With Pandas, NumPy, and Matplotlib. 2nd ed. 2018. Berkeley, CA : Apress : Imprint: Apress.

Nvidia Developer. 2022. Nvidia. Verkkosivu. Viitattu 12.5.2022. https://developer.nvidia.com/cudnn

Nvidia Developer. 2022. Nvidia. Verkkosivu. Viitattu 12.5.2022. https://developer.nvidia.com/cuda-toolkit

OpenCV. 2022. About. Verkkosivu. Viitattu 12.5.2022. https://opencv.org/about/

Pietikäinen, M. & Silvén, O. 2019. Tekoälyn haasteet – Koneoppimisesta ja konenäöstä tunnetekoälyyn. E-kirja. Konenäön ja signaalianalyysin keskus, Oulun yliopisto.

Suomen vahinkotarkastus SVT Oy. n.d. Vahinkotarkastus. Verkkosivu. Viitattu 25.2.2022. http://www.svt.fi/vahinkotarkastus/

Świeżewski, J. 2020. YOLO Algorithm and YOLO Object Detection. Internet artikkeli. Julkaistu 22.5.2020. Luettu 25.8.2022. https://appsilon.com/object-detection-yolo-algorithm/

Tuominen, H. & Neittaanmäki. P. 2019. Tekoälyn perusteita ja sovelluksia. E-kirja. Informaatioteknologian tiedekunta, Jyväskylän yliopisto.

Varho, E. YLE. 2020. Kiina ja USA kisaavat tekoälyvallankumouksen herruudesta – EU myöhästyi jo lähdöstä. Uutisartikkeli. Julkaistu 1.7.2020. Päivitetty 2.7.2020. Luettu 31.7.2022. https://yle.fi/uutiset/3-11426520

Yohanandan, S. 2020. mAP (mean Average Precision) might confuse you! Internet artikkeli. Julkaistu 9.6.2020. Viitattu 5.10.2022. https://towardsdatascience.com/map-mean-average-precision-might-confuse-you-5956f1bfa9e2