Kanditutkielma

Tietojenkäsittelytieteen kandiohjelma

Aine

Jami Heljomaa

1.11.2021

Matemaattis-luonnontieteellinen tiedekunta

Helsingin yliopisto

Yhteystiedot

PL 68 (Pietari Kalmin katu 5)
00014 Helsingin yliopisto

Sähkopostiosoite: info@cs.helsinki.fi
URL: http://www.cs.helsinki.fi/

Sisällys

1 Johdanto

Tässä tekstissä käsitellään objektintunnistusta, eli objektien havainnoimista kuvamateriaalista laskennallisin keinoin. Objektintunnistus on nopeasti kehittyvä tutkimusalue, sillä tietokoneteknologia kehittyy jatkuvasti nopeammaksi ja tehokkaammaksi, sallien näin monimutkaisempien sovelluksien käytön. Samalla kaupungistuminen luo insentiivejä kameravalvontaan turvallisuuden ja kaupunkisuunnittelun tehokkuuden lisäämisen nimissä. Lisämausteensa asiaan tuovat pienet lennokit, joiden yleistyminen sallii niiden käyttämisen objektintunnistussovelluksissa, mikä tuo sekä uusia mahdollisuuksia, että uusia haasteita.

Ensimmäisessä luvussa käydään lyhyesti läpi hieman objektintunnistuksen historiaa sekä perusperiaatteita, mukaan lukien nykyaikaiset koneoppimismenetelmät. Toinen luku esittelee kaksi merkittävää viime vuosien objektintunnistusmenetelmää, jotka ovat edistäneet tehokkaan objektintunnistuksen kehitystä. Kolmannessa luvussa keskitytään objektintunnistukseen kapeammassa käyttötarkoituksessa: autojen tunnistamisessa lennokista käsin otetusta kuvamateriaalista.

2 Objektintunnistus

2.1 Objektintunnistuksen historiaa

Objektintunnistus on laskennallinen ongelma, jonka teknologisten ratkaisujen tavoitteena on pyrkiä tunnistamaan ja luokittelemaan kohteita kuvamateriaalista. Nykypäivänä kyetään luomaan ja käyttämään neuroverkkoihihin perustuvia syväoppivia algoritmeja, ja ala kehittyy edelleen nopeasti. Ennen näitä nykyaikaisempia menetelmiä käytettiin yksinkertaisempia tapoja analysoida kuvamateriaalia, ja oli esimerkiksi yleistä luoda jonkinlainen kartta siitä, minkä muotoiset jonkin objektin ulkoraunat ovat, ja verrata sitä syötekuvaan tai sen alueisiin.

Scale-invariant feature transform (SIFT) on eräs merkittävä esimerkki objektintunnistusmenetelmistä ennen kuin syväoppimisen valtasi alaa. Sen tunnistusmenetelmä perustuu piirrevektoreihin, jotka kerätään kuvista, jotka sille annetaan luokittelumateriaalina. Luokittelukuvasta kerätään tunnistettavia kohtia, jotka eivät ole herkkiä pienille häiriöille tai vääristymille ja nämä tallennetaan piirrevektoreiksi. Yhden luokan piirrevektoreille tallennetaan geometriset sijainnit joita voidaan sitten verrata syötekuvaan kohdetta etsiessä. (Lowe, 1999)

SIFT-menetelmä on jokseenkin vastustuskykyinen erilaisille häiriötekijöille, sillä vektorien geometrian huomioon ottava toimintamalli mahdollistaa objektin tunnistamisen, vaikka se olisikin eri kokoinen kuin alkuperäinen luokitteluun käytetty kuva, tai vaikka sen geometria olisi hieman vääristynyt, esimerkiksi poikkeavan kuvakulman vuoksi. Koska yhtä objektia varten on monta vektoria, voidaan myös sallia tilanteita, joissa jotkin vektoreista eivät ole näkyvissä, mutta objekti voidaan silti hyväksyä havaituksi; Objekti voidaan havaita, vaikka se olisi osittain piilossa. SIFT-menetelmän eräs merkittävä heikkous on kuitenkin juuri siinä, että luokittelu pohjaa luottamukseen siitä, että vektorien keskinäiset suhteet eivät merkittävästi muutu: Jos havaittava objekti on esimerkiksi esine, jossa on liikkuvia osia, eivätkä ne ole aina täysin samassa järjestyksessä, voi objekti jäädä helposti havaitsematta. (Lowe, 1999)

Toinen merkittävä syväoppimista edeltävä objektintunnistusmenetelmä on alun perin jalankulkijoiden tunnistamiseen suunniteltu Histogram of Oriented Gradients (HOG), jon-

ka toimintaperiaate nojaa luokittelumateriaalista luotuun karttaan, joka kuvaa objektin reunoja ja niiden suuntia. Luokittelumateriaalin kuvista saaduilla reunojen sijainneilla ja suunnilla ikään kuin "koulutetaan"eräänlainen "muotti", jota verrataan syötekuvista eroteltujen reunojen sijainteihin ja suuntiin. Dalal ja Briggs saivat HOG:n toimimaan varsin hyvin ihmisten tunnistamisessa, kunhan ihminen oli kuvassa pystyasennossa, eikä pitänyt useita raajoja erikoisissa asennoissa. (Dalal ja Triggs, 2005)

2.2 Koneoppiminen ja neuroverkot

Koneoppiminen, eli tietokoneohjelman käyttäminen tarkoitukseen, jossa se tekee suhteellisen itsenäisesti päätelmiä annetun datan perusteella, on ollut suosittu tutkimuksen aihe niin kauan kuin tietokoneita on ollut olemassa. Koneoppiminen on myös hyvin olennainen osa objektintunnistusta, ja siinä missä lineaarisen regression ja naiivin bayesilaisen luokittelijan sovellukset hyödyntävät koneoppimista, myös edellisessä osassa mainitut SIFT ja HOG ovat koneoppimiseksi luokiteltavien sovelluksien piirissä. (Goodfellow et al., 2016)

Viimeisen noin kymmenen vuoden aikana tietotekniikan laskentatehon kasvu sekä innovaatiot algoritmien suunnittelussa ovat tuoneet suureen suosioon neuroverkkoihin pohjautuvat syväoppimismenetelmät, jotka avaavat uusia ulottuvuuksia verrattuna yksinkertaisempiin koneoppimisen menetelmiin. Nämä yksinkertaisemmat menetelmät perustavat toimintansa siihen, että niille annetaan analysoitavasta datasta valmiita ominaisuuksia, joita sitten verrataan varhaisemman datan vastaaviin ominaisuuksiin, tehden tämän perusteella päätelmiä. Jos esimerkiksi haluttaisiin koneen päättelevän, onko jokin kulkuneuvo polkupyörä, henkilöauto vai linja-auto, yksinkertaista koneoppimista hyödyntävä sovellus voisi käyttää esimerkiksi lukittuja ominaisuuksia 'pituus', 'korkeus', 'paino' ja 'renkaiden lukumäärä' ratkaisun etsimiseen, eikä sovellus tekisi omaa arviointejaan näiden ominaisuuksien määrittelemiseksi. (Goodfellow et al., 2016)

Luokittelun kriteereinä käytettävien ominaisuuksien hyvä määrittely on useinmiten keskeistä onnistuneen koneoppimisen saavuttamiseksi. Haasteena on kuitenkin se, että ihmiskäyttäjän voi olla hyvin vaikeaa luoda oikeanlaisia kriteerejä niiden suuren määrän tai monimutkaisuuden vuoksi. Objektintunnistus on tästä hyvä esimerkki, sillä usein kuvamateriaalissa on paljon pikseleitä ja objektintunnistuksessa on kyse abstraktien asioiden havainnoimisesta näiden pikselien arvojen perusteella. On suhteellisen helppoa saada algoritmi oppimaan esimerkkinä mainittujen ajoneuvojen ominaisuuksia, kun abstraktit ominaisuudet, kuten pituus, korkeus ja muoto ovat valmiiksi määritelty ja annettu.

On huomattavasti monimutkaisempi tehtävä luoda algoritmi, joka sisäistää edellä mainitun kaltaisia abstrakteja ominaisuuksia suoraan pikselien perusteella. Seuraava luonnollinen edistysaskel kohti kehittyneempää koneoppimisjärjestelmää onkin järjestelmän kyky erotella luokiteltavia ominaisuuksia itsenäisesti, ilman että niitä määritellään etukäteen. (Goodfellow et al., 2016)

Nykyaikaiset koneoppimismenetelmät toimivat juuri edellä mainitun perusajatuksen mukaisesti: annetaan koneoppimisalgoritmille syötedataa ja annetaan sen itse tehdä päätelmiä siitä, mikä datan eri osissa on merkittävää. Tavoitteena usein onkin, etenkin objektintunnistuksessa, että datasta voitaisiin kerätä erottavat tekijät, joiden perusteella luokitteluja voidaan tehdä. Tällaisen, erottavia tekijöitä havaitsevan mallin luominen käsin voisi tehtävästä riippuen olla hyvin vaikeaa tai käytännössä mahdotonta, mutta kun työ saadaan laskennallisesti automatisoitua, on kyse minuuttien, tuntien tai kuukausien tietokonelaskennasta, riippuen ratkaistavan tehtävän monimutkaisuudesta. (Goodfellow et al., 2016)

Abstraktien ominaisuuksien poimiminen datasta tapahtuu nykypäivän sovelluksissa syväoppimisen (deep learning) keinoin. Sen toiminpaperiaate nojaa abstraktion tasoihin, jossa ylemmän tason päätelmät perustuvat alemman tason päätelmiin. Näin raakadatan päälle rakentaa kerros kerrokselta yhä laajempi käsitys siitä, mitä data edustaa, ylimmällä kerroksella ollen lopullinen päätelmä. Kuvatiedoston tapauksessa tämä voisi tarkoittaa, että ensimmäisessä abstraktiokerroksessa ovat kuvasta havaitut reunat, toisessa kerroksessa reunoista koostuvat kulmat ja kolmannessa kulmista koostuvat muodot – näiden muotojen perusteella on mahdollista jo muodostaa luokitteluja kuvan objekteista. (Goodfellow et al., 2016)

Neuroverkot ovat työkalu abstraktiotasoja kerroksittain nostavan rakenteen saavuttamiseksi. Neuroverkko on nimensä mukaisesti verkko, joka koostuu neuroneiksi kutsutuista solmuista, joista jokainen toteuttaa jonkin funktion. Kokonaisuudessaan neuroverkko ottaa syötteen syötekerroksen neuroneille ja antaa tulosteen tulostuskerroksen neuroneille. Syöte- ja tulostuskerroksia kutsutaan näkyviksi kerroksiksi, sillä ne ovat täysin tarkasteltavissa ulkoa päin, kun taas näiden välissä olevia kerroksia kutsutaan piilotetuiksi kerroksiksi. Nimi 'piilotetut kerrokset' viittaa siihen, että näiden kerroksien neuronien painoarvoja ei anneta etukäteen, vaan koneoppimismalli optimoi näille parhaimmat mahdolliset arvot. Varsin tyypillinen neuroverkko on monikerroksinen perseptroniverkko (multilayer perceptron, jatkossa MLP), joka on yksisuuntainen ja yleensä täysin kytketty, mikä tarkoittaa, että jokainen solmu saa syötteenään jokaisen edellisen kerroksen solmun tulosteen ja antaa tulosteensa eteenpäin jokaiselle seuraavan kerroksen neuronille. (Goodfellow et al., 2016)

Nykyaikaisissa objektintunnistussovelluksissa käytetään hyvin usein jonkinlaista konvoluutioneuroverkkoa. (Ren et al., 2016) (Redmon, Joseph and Farhadi, Ali, 2018) Konvoluutioneuroverkko on nimitys neuroverkolle, jossa käytetään konvoluutio-operaatioita. Konvoluutio-operaatio on kahden funktion operaatio, joka tuottaa näistä tuotetun uuden funktion. Tämän funktion tulosta kutsutaan usein piirrekartaksi. (Goodfellow et al., 2016)

2.3 Objektintunnistusmallien toiminta

Objektien tunnistaminen kuvamateriaalista käsittää nykyisissä ratkaisuissa kaksi erillistä osaa: potentiaalisten kiinnostavien objektien paikantamisen ja niiden luokittelun. Eri mallit käyttävät hyvin toisistaan poikkeavia ratkaisuja näiden tavoitteiden saavuttamiseen.

Pääosin objektintunnistusmallit jaetaan kahteen luokkaan – yksi- ja kaksivaiheisiin – sen perusteella, jakavatko ne objektien paikannuksen ja luokittelun kokonaan erillisiksi osiksi prosessia. Yksivaiheiset mallit ovat tyypillisesti kaksivaiheisia nopeampia, mutta joutuvat tekemään kompromisseja paikannuksen tarkkuuden suhteen. Klassinen esimerkki kaksivaiheisesta mallista on konvoluutioneuroverkot objektintunnistuksen eturintamalle tuonut R-CNN. Sen toimintamallissa aluksi kuvasta poimitaan joukko kiinnostavia alueita, jotka myöhemmin viedään käsiteltäväksi. (Girshick et al., 2014) Yksivaiheiset mallit pyrkivät luomaan arvioitavat alueet ja itse arvioinnit rinnakkaisina operaatioina ja yhdistämällä näiden tulokset loppupäätelmäksi. Yksivaiheisia objektintunnistusmalleja ovat esimerkiksi YOLO ja SSD, joita käsitellään seuraavassa osassa. (Redmon et al., 2016) (Liu et al., 2016)

Objektintunnistusmallien keskiössä on luokittelijarunko (classification backbone), joka nykypäivänä on käytännössä aina jonkinlainen konvoluutioneuroverkko. Erilaisilla rungon rakenteilla on etuja toisiinsa nähden, ja eri tarkoituksiin on saatetaan käyttää verkkoja, joissa on konvoluutiokerroksien lisäksi täysin kytkettyjä kerroksia tai jäännöskerroksia (Residual neural network, ResNet). (Kyrkou et al., 2018) (Ren et al., 2016) (Redmon, Joseph and Farhadi, Ali, 2018)

Mallien toimintaa arvioidaan yleensä kahden metriikan kautta: tunnistustarkkuus ja -nopeus. Tunnistusnopeus ilmoitetaan tyypillisesti ruutujen lukumääränä, jonka malli pystyy käsittelemään sekunnin aikana. Nopeus on täten riippuvainen käytetyn laitteiston suorituskyvystä. Tunnistustarkkuus ilmoitetaan yleensä mAP-yksikkönä (Mean Average Presicion, keskimääräinen tarkkuus), joka ottaa huomioon oikein luokitellut kohteet ja sen, kuinka lähellä mallin antamat rajat objekteille ovat todellisia rajoja. Rajojen osuma-

tarkkuus lasketaan mallin antamien rajojen ja todellisten rajojen leikkauksen ja yhdisteen välisen suhteen mukaan (Intersection over Union, IoU). (Liu et al., 2016) (Ren et al., 2016)

3 Hahmontunnistusmallit

Tässä luvussa käsitellään kahta hahmontunnistusmallia, joita tapaa lennokkeihin rakennetuissa sovelluksissa. Näitä hahmontunnistusmalleja yhdistää se, että ne ovat yksivaiheisia ja niiden suunnittelussa on annettu erityistä painoarvoa suorittamisen keveydelle ja nopeudelle, mikä tekee niistä hyvin soveltuvia mobiileille tai muuten järjestelmän suorituskykyä jollakin tapaa rajoittaville käyttökohteille, esimerkiksi lennokeille.

3.1 YOLO

YOLO (You Only Look Once) syntyi pyrkimyksestä saada aikaan silloisia menetelmiä nopeampi, reaaliajassa toimiva objektintunnistusmalli. Tuohon aikaan käyttökelpoista tunnistustarkkuutta tarjoavat mallit eivät kyenneet riittävän nopeaan tunnistamiseen salliakseen reaaliaikaisen toiminnan kevyellä laitteistolla. Nopeuden suhteen YOLO:n suurin etu kilpaileviin malleihin nähden oli sen tapa löytää kiinnostavia alueita (Region of Interest, RoI). Siinä missä silloin yleisesti parhaaksi mielletty malli, R-CNN, etsii kuvasta hahmojen rajoja ja tätä kautta generoi RoI-ehdotuksia jatkokäsittelyä varten, YOLO toimii yksinkertaisemmalla ja kaavamaisemmalla, mutta eritoten nopeammalla tavalla. (Redmon et al., 2016)

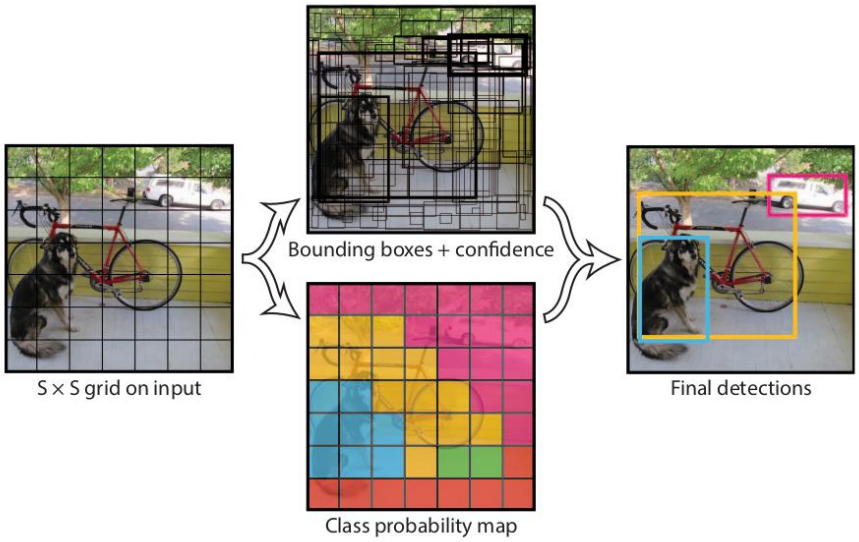
YOLO:n tapa löytää kiinnostavia alueita voidaan kuvata neljällä eri vaiheella. Ensimmäinen vaihe on jakaa kuva soluiksi $S \times S$ -kokoisena ruudukkona. Kun havainnoitavan objektin varaaman alueen keskikohta osuu solun sisälle, tämän solun vastuulle asetetaan kyseisen objektin luokitteleminen. Jokainen solu arvioi B rajaavaa laatikkoa (suorakulmion muotoista aluetta) ja antaa näille pistearvon sillä perusteella, kuinka varma objektintunnistusmalli on siitä, että laatikko sisältää jonkin objektin. Kukin laatikko sisältää viisi arvoa: x, y, w, h ja edellämainitun varmuusarvon. x ja y edustavat laatikon keskipisteen koordinaatteja solun x- ja y-akseleilla ja w ja h laatikon korkeutta ja leveyttä. Varmuusarvo siitä, sisältääkö laatikko objektin, edustaa kyseisen laatikon ja objektin alueen todellisesti ympäröivän laatikon leikkauksen ja yhdisteen kokojen suhdetta (Intersection over Union, IoU), eli haluttu arvo mahdollisimman lähellä yhtä. Jos taas alueella ei ole objektia, arvon tulisi olla nolla. Kuva 3.1 (Redmon et al., 2016)

Jokainen $S \times S$ -kokoisen ruudukon solu laskee todennäköisyydet jokaiselle objektiluo-

kalle. Koska objektiluokkien todennäköisyydet lasketaan solukohtaisesti sen sijaan, että ne laskettaisiin erikseen jokaiselle edellisessä kappaleessa mainituille kiinnostavien alueiden laatikolle, saavutetaan merkittävä nopeushyöty, samalla kuitenkin uhraten objektin paikannustarkkuutta hieman. (Redmon et al., 2016)

2016 julkaistun alkuperäisen YOLO:n runkoneuroverkon suunnittelussa on otettu mallia GoogLeNet:stä, ja se koostuu syötekerroksen jälkeen 24 konvoluutiokerroksesta, joiden jälkeen seuraa kaksi täysin kytkettyä kerrosta ja tulostekerros. (Redmon et al., 2016) YOLO:n uudemmassa versiossa, YOLOv3:ssa, runkoneuroverkko koostuu 53:sta konvoluutiokerroksesta, joiden välissä on jäännöskerroksia (Residual neural network, ResNet), jotka mahdollistavat kerroksien yli loikkaamisen, ja lopussa yksi täysin kytketty kerros. (Redmon, Joseph and Farhadi, Ali, 2018)

Vuoden 2016 artikkelia varten YOLO oli esikoulutettu tuhatluokkaisella ImageNet-datasarjalla, joka on suunniteltu kilpailukäyttöön. (Russakovsky et al., 2015) Pascal VOC 2007 -sarjalla testattuna YOLO oli tarkkuudeltaan varsin vertailukelpoinen ja selvästi nopeampi kuin tuolloin edistyneet Faster R-CNN -mallista jatkokehitetyt ratkaisut: yhtä suurella, noin 63 mAP:n tarkkuudella, YOLO saavutti 45 ruudun sekuntinopeuden, kun Faster R-CNN ZF sai nopeuslukemakseen 18 ruutua sekunnissa. (Redmon et al., 2016) YOLOv3 on sittemmin pitänyt YOLO:n nopeuden ja kyvyn reaaliaikaiseen toimintaan, mutta on parantanut tarkkuuttaan optimoinneilla objektien etsimisen prosessissa. (Redmon, Joseph and Farhadi, Ali, 2018)



Kuva 3.1: Tapa, jolla YOLO piirtää objektien rajat ja suorittaa luokittelun

3.2 SSD

Vuonna 2016 esitelty SSD haastoi YOLO:n ensimmäisen version reaaliaikaisen objektintunnistuksen tarjoamisessa. Sen yksivaiheinen toimintatapa on samankaltainen kuin YOLO:n vastaava siinä mielessä, että se soveltaa ennaltamäärättyjen soluruudukkojen periaatetta arvioidessaan mahdollisesti objekteja sisältäviä alueita. Tästä eteenpäin SSD:n toiminta kuitenkin poikkeaa YOLO:n vastaavasta täysin, sillä yhden soluruudukon sijaan SSD muodostaa niitä useita – alkuperäisen artikkelin esimerkki on luoda 4×4 ja 8×8 kokoiset ruudukot. YOLO:n tapaan SSD arvioi soluruudukkojen jokaisen solun kohdalla luokkaosuman todennäköisyyden. (Liu et al., 2016)

Kukin soluruudukon solu muodostaa eri kokoisia ja kuvasuhteen omaavia laatikoita, jotka muodostavat joukon laatikoita, joiden osuvuus objektien sijaintien suhteen myöhemmin arvioidaan. Laatikoiden koot ja mittasuhteet riippuvat siitä, mitä luokkia kyseisen solun arvellaan edustavan. Tämän jälkeen laatikot käydään vielä läpi antaen niille lopullisen luokan ja niiden sijainti hiotaan lopulliseen muotoon. (Liu et al., 2016)

SSD:n neuroverkon rakenne ei sisällä täysin kytkettyjä kerroksia, vaan perättäisiä konvoluutiokerroksia, sekä rinnakkaisia konvoluutiokerroksia vastaamaan eri kokoisia soluruudukoita. (Liu et al., 2016)

Alkuperäisessä artikkelissa SSD:tä verrattiin YOLO:on Pascal VOC 2007 -testin avulla. Yhtä nopeilla, noin 21-22 ruudun sekuntinopeudella toimivilla konfiguraatioilla YOLO:n mAP-pisteytys oli noin 66, missä SSD:n vastaava oli noin 76. SSD:n nopein konfiguraatio pääsi 59 ruudun sekuntinopeuteen mAP-tarkkuudella 74,3. Tällä metriikalla mitattuna SSD toimii siis YOLO:a paremmin. (Liu et al., 2016)

4 Autojen tunnistaminen lennokin kuvakulmasta

Erilaiset lennokit, eli miehittämättömät lentävät laitteet, ovat yleistyneet ratkaisuna, kun on tarve valvoa tai tarkkailla laajoja alueita, joilla esiintyy näköesteitä. Tällaisia ovat esimerkiksi urbaanit alueet ja seudut, joiden maastoissa on suuria korkeuseroja. Liikenne ja ajoneuvot ovat yksi yleinen tarkastelun kohde, sillä näitä seuraamalla voi löytää dataa tukemaan kaupunkisuunnittelua. Tässä luvussa tarkastellaan joitakin erityispiirteitä, jotka liittyvät objektien – erityisesti autojen – tunnistukseen lento-olosuhteissa, mukaanlukien joitakin haasteita ja keinoja näiden voittamiseksi.

4.1 Erilaisia käyttötarkoituksia

Autojen tunnistamiseen käytettävien sovelluksien tarkoitusperiä on monenlaisia, joten ratkaisutkin poikkeavat toisistaan. Tarkoituksena voi olla esimerkiksi laskea autopaikoitusalueen autojen lukumäärää (Hsieh et al., 2017), arvioida liikenteen runsautta ja sujuvuutta tiiviisti asutussa kaupunkiympäristössä (Zhu et al., 2018) tai seurata yhtä, tiettyä ajoneuvoa (Wu et al., 2019).

Lennokin käyttökohteesta ja rakenteesta riippuen sen toimintaan saattaa liittyä huomattavia rajoituksia. Ensinnäkin yleensä lennokki toimii varastoidun energian – esimerkiksi akun – voimalla, mikä voi asettaa rajoitteita käytetyn tietoteknisen laitteiston laskentateholle ja ohjelmiston raskaudelle. Jos lennokin on tarkoitus kyetä tekemään itsenäisiä päätöksiä käsittelemänsä datan perusteella, tulee laskennan voida tapahtua itse lennokin rakenteiden piirissä. Jos taas lennokki on toiminta-aikansa siihen sopivan tietoliikenneyhteyden päässä, voi olla tehokkaampaa lähettää prosessoitava datavirta keskuspalvelimen käsiteltäväksi. Oman lisämuuttujan asiaan tuo se, onko sovelluksen tarpeen pystyä tunnistamiseen reaaliajassa, esimerkiksi useita kertoja sekunnissa, vai riittääkö pienempi päivitysnopeus.

4.2 Ratkaisuja autojen tunnistamseen ilmasta käsin

(Benjdira et al., 2019) vertasi Faster R-CNN:n ja YOLOv3:n toimintaa monen auton yhtäaikaisessa tunnistamisessa laitteistokokoonpanolla, jossa lennokki lähetti videovirtaa keskuspalvelimena toimivalle työasemalle. Havaintona oli, että molempiin malleihin perustuvat ratkaisut osoittivat miltei täydellistä tarkkuutta, R-CNN:ään perustuvan ratkaisu jäi 79.17% osuvuudellaan YOLOv3:n vastaavasta 98.81%:sta huomattavasti jälkeen. YOLOv3 oli näistä kahdesta myös nopeampi ja kykeni 0.057ms keskiarvoisella päivitysnopeudellaan reaaliaikaiseen tunnistamiseen, kun taas R-CNN:n tulos 1.39s ei soveltuisi reaaliaikaiseen tunnistamiseen.

(Hsieh et al., 2017) pyrkii tarjoamaan mahdollisimman tehokkaan kokonaisuuden pysäköityjen autojen lukumäärän laskemiseen paikoitusalueella. Artikkeli esittelee CARPK-datasarjan, jonka kuvissa on ylhäältä päin kuvattuna lähes 90 000 pysäköityä autoa erilaisilla paikoitusalueilla. Artikkelissa esitellään myös tarkoitusta varten räätälöity konvoluutioneuroverkkoon perustuva malli kiinnostavien alueiden (RoI, Region of Interest) poimimiseksi: malli ottaa huomioon tavat, joilla paikoitusalue tyypillisesti täyttyy ja suosii autojen lähellä olevia alueita etsiessään uusia autoja.

(Zhu et al., 2018) soveltaa Single-Shot Multibox Detectorin (SSD) rakennetta ja esittelee Enhanced-SSD -objektintunnistusmallin haastamaan SSD:n ja YOLO:n, erityisenä käyttökohteena korkeatarkkuuksinen (4k-resoluutioinen) video, jossa liikennettä kuvataan ylhäältä käsin. Enhanced-SSD on hieman hitaampi kuin SSD, mutta selvästi tarkempi objektien paikannuksen suhteen. YOLO:lle Enhanced-SSD häviää nopeudessa selvästi, mutta YOLO häviää paikannustarkkuudessa SSD:lle ja Enhanced-SSD:lle huomattavasti. Artikkelissa esitellään myös uusi datasarja UavCT, joka tarjoaa 100 000 ruutua 4k-tarkkuuksista videokuvaa urbaanista liikenteestä ylhäältäpäin kuvattuna.

(Kyrkou et al., 2018) esittelee DroNet-mallin, jonka pyrkimyksenä on tarjota hyvin nopea ja erittäin optimoitu ratkaisu autojen tunnistamiseen. DreamNetin eräs erikoisuus on optimoida kiinnostavien alueiden löytämistä syöttämällä kuvamateriaalin mukana tieto siitä, kuinka korkealla maanpintaan verrattuna lennokki on kuvaushetkellä, mikä auttaa mallia päättelemään, minkä kokoisia objekteja autot saattavat kuvissa olla. Artikkelissa vertailukohtina käytetään YOLO:n kevyitä versioita TinyYoloVoc, TinyYoloNet ja SmallYoloV3, joihin suhteutettuna DroNet on jokseenkin kilpailukykyinen, mutta ei selvä parannus.

Lähteet

Benjdira, B., Khursheed, T., Koubaa, A., Ammar, A. ja Ouni, K. (2019). "Car Detection using Unmanned Aerial Vehicles: Comparison between Faster R-CNN and YOLOv3". Teoksessa: *2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS)*, s. 1–6. DOI: 10.1109/UVS.2019.8658300.

Dalal, N. ja Triggs, B. (2005). "Histograms of oriented gradients for human detection". Teoksessa: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 1, 886–893 vol. 1. DOI: 10.1109/CVPR.2005.177.

Girshick, R., Donahue, J., Darrell, T. ja Malik, J. (2014). "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". Teoksessa: *2014 IEEE Conference on Computer Vision and Pattern Recognition*, s. 580–587. DOI: 10.1109/CVPR.2014.81.

Goodfellow, I., Bengio, Y. ja Courville, A. (2016). *Deep Learning*. http://www.deeplearningbook.org. MIT Press.

Hsieh, M.-R., Lin, Y.-L. ja Hsu, W. H. (2017). "Drone-Based Object Counting by Spatially Regularized Regional Proposal Network". Teoksessa: *2017 IEEE International Conference on Computer Vision (ICCV)*, s. 4165–4173. DOI: 10.1109/ICCV.2017.446.

Kyrkou, C., Plastiras, G., Theocharides, T., Venieris, S. I. ja Bouganis, C.-S. (2018). "DroNet: Efficient convolutional neural network detector for real-time UAV applications". Teoksessa: *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*, s. 967–972. DOI: 10.23919/DATE.2018.8342149.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. ja Berg, A. C. (2016). "SSD: Single Shot MultiBox Detector". *Lecture Notes in Computer Science*, s. 21–37. ISSN: 1611-3349. DOI: 10.1007/978-3-319-46448-0_2. URL: http://dx.doi.org/10.1007/978-3-319-46448-0_2.

Lowe, D. (1999). "Object recognition from local scale-invariant features". Teoksessa: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Vol. 2, 1150–1157 vol.2. DOI: 10.1109/ICCV.1999.790410.

Redmon, J., Divvala, S., Girshick, R. ja Farhadi, A. (2016). "You Only Look Once: Unified, Real-Time Object Detection". *arXiv*. DOI: 10.1109/CVPR.2016.91.

Redmon, Joseph and Farhadi, Ali (2018). "YOLOv3: An Incremental Improvement". *arXiv*.

Ren, S., He, K., Girshick, R. ja Sun, J. (2016). *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. arXiv: 1506.01497 [cs.CV].

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C. ja Fei-Fei, L. (2015). *ImageNet Large Scale Visual Recognition Challenge*. arXiv: 1409.0575 [cs.CV].

Wu, H.-H., Zhou, Z., Feng, M., Yan, Y., Xu, H. ja Qian, L. (2019). "Real-Time Single Object Detection on The UAV". Teoksessa: *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, s. 1013–1022. DOI: 10.1109/ICUAS.2019.8797866.

Zhu, J., Sun, K., Jia, S., Li, Q., Hou, X., Lin, W., Liu, B. ja Qiu, G. (2018). "Urban Traffic Density Estimation Based on Ultrahigh-Resolution UAV Video and Deep Neural Network". *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 11.12, s. 4968–4981. DOI: 10.1109/JSTARS.2018.2879368.