Anna Arikainen

Darknet-liikenteen analysointi koneoppimisalgoritmeilla

Tietotekniikan Pro gradu -tutkielma

12. toukokuuta 2023

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Anna Arikainen

Yhteystiedot: anna.a.arikainen@student.jyu.fi

Ohjaaja: Timo Hämäläinen

Työn nimi: Darknet-liikenteen analysointi koneoppimisalgoritmeilla

Title in English: Darknet traffic analysis with machine learning algorithms

Työ: Pro gradu -tutkielma

Opintosuunta: Ohjelmisto- ja tietoliikennetekniikka

Sivumäärä: 66+0

Tiivistelmä: Tämä pro gradu -tutkielma käsittelee Darknet 2020 -nimisen datasetin testaamista random forest-, gradient boosting- ja logistic regression-algoritmeilla. Tutkimus toteutettiin konstruktiivisena tutkimuksena. Tutkimuksen aineisto koostuu New Brunswick yliopiston tutkijoiden Habibi Lashkarin, Kaurin ja Rahalin tekemästä artikkelista *DIDarknet: A Contemporary Approach to Detect and Characterize the Darknet Traffic using Deep Image Learning* sekä heidän tuottamastaan Darknet 2020 -datasetistä. Tutkimuksen tarkoituksena oli selvittää, miten koneoppimisen algoritmit selviytyvät datasetissä olevan darknet-tietoliikennettä imitoivan datan luokitellusta sekä verrata saatuja tuloksia tutkijoiden esittelemään syväoppimisen malliin nimeltä DIDarknet.

Tutkimuksen lopputuloksena voidaan nähdä useamman eri koneoppimisalgoritmin tarkkudet luokitella datasetin tietoliikenne Label-ominaisuuden perusteella. Random forest -algoritmi suoriutui luokittelutehtävästä huomattavasti kahta muuta algoritmia paremmin. Tutkimuksen perusteella voidaan nähdä, että DIDarknet on suoriutunut darknet-liikenteen luokittelusta ylivoimaisesti paremmin kuin tutkielmassa esiintyvät ML-algoritmit.

Avainsanat: darknet, koneoppiminen, syväoppiminen, random forest, gradient boosting, logistic regression, konvoluutioneuroverkko

Abstract: This master's thesis deals with testing the Darknet 2020 dataset with random forest, gradient boosting and logistic regression algorithms. The study was carried out as a

constructive study. The material of the study consists of the article *DIDarknet: A Contemporary Approach to Detect and Characterize the Darknet Traffic using Deep Image Learning* by researchers Habibi Lashkari, Kaur and Rahali of the University of New Brunswick and the Darknet 2020 dataset produced by them. The purpose of the study was to find out how the machine learning algorithms cope with the classification of the data simulating darknet communication in the dataset, and to compare the obtained results with the deep learning model presented by the researchers called DIDarknet.

The final result of the research is the accuracy of several different machine learning algorithms to classify data traffic based on the Label feature. The random forest algorithm performed the classification task significantly better than the other two algorithms. On the basis of the research, it can be concluded that DIDarknet has performed by far better than the ML algorithms appearing in the thesis in the classification of darknet traffic.

Keywords: darknet, machine learning, deep learning, random forest, gradient boosting, logistic regression, convolutional neural network

Kuviot

Taulukot

Sisältö

1 Johdanto

Darknet on internetin sisäinen aliverkko, johon pääsee kiinni vain erityisillä ohjelmistoilla, konfiguraatioilla tai oikeuksilla. Usein tämä verkko käyttää jotain muuta tiedonsiirtoprotokollaa kuin HTTP- tai HTTPS-protokollaa. Tämän seurauksena perinteisen hakukokeet, kuten esimerkiksi Google, eivät pysty löytämään näitä darknetin sivustoja. Darknet-termillä voidaan kuvailla erillisten darknettien muodostamaa yhtenäistä kokonaisuutta, mutta voidaan myös tarkoittaa yksittäisiä darknet-verkkoja, joiden koko voi vaihdella suuresti. Internetin muodostamasta valtavasta kokonaisuudesta voidaan erottaa kolme osajoukkoa: pintaverkon, johon pääsee käsiksi perinteisillä hakukoineilla; syväverkon, valtava kokonaisuus, joka koostuu indeksoimattomasta, yksityisestä tai vaikeasti saatavilla olevasta sisällöstä; sekä darknetin, piilotetun internetin, joka koostuu erillisistä darkneteistä ja vaihtoehtoisista verkoista (Gayard 2018, 9–10).

Nykypäivänä darknet on esillä mediassa useimmiten huonossa valossa ja siihen assosioidaan helposti monia laittomia asioita, kuten palkkamurhia ja huumekauppaa. Tästä huolimatta darknettiä ei alunperin kuitenkaan kaavailtu olevan lainvastaisen toiminnan mahdollistaja, mutta sen tarjoama vahva anonymiteetti väistämättä houkutteli paikalle myös rikollisia (Mirea, Wang ja Jung 2019). Koska pimeässä verkossa ei ole laillisia palvelimia, kaikki liikenne katsotaan ei-toivotuksi ja sitä käsitellään yleensä koettimena (engl. probe), takaisinsironnalla (engl. backscatter) tai virheellisillä asetuksilla (engl. misconfiguration). Darknetit tunnetaan myös verkkoteleskooppeina, nielureikinä tai mustina aukkoina. (Habibi Lashkari, Kaur ja Rahali 2020, 1)

Darknet-liikenteen luokittelu on hyvin haastavaa johtuen vanhoista ja puutteellisista dataseteistä sekä darknet-liikenteen vaikean luonteen vuoksi, kuten Habibi Lashkari ym. (2020, 1) tutkimuksessaan toteavat. Koska darknettien voidaan ajatella olevan sinkoja, jotka hyväksyvät ainoastaan saapuvat paketit, mutta ei vastaavasti tue lähteviä paketteja, on hyvin haastavaa saada luotettavaa dataa, jolla voidaan kouluttaa uusia koneoppimismalleja. (Habibi Lashkari, Kaur ja Rahali 2020)

Tämän pro gradu -tutkielman pohjana toimii Habibi Lashkarin ym. (2020) tekemä darknet-

liikennettä analysoiva tutkimus *DIDarknet: A Contemporary Approach to Detect and Characterize the Darknet Traffic using Deep Image Learning*. Habibi Lashkarin ym. (2020) tutkimuksen pohjalta toteutettiin tässä pro gradu -tutkielmassa konstruktiivinen tutkimus, joka käsittelee aiemman tutkimuksen datasettiä ja sen analysointia koneoppimisalgoritmeilla. Pro gradu -työn ensisijaisena tavoitteena on tutkia, miten yleiset koneoppimisalgoritmit selviytyvät Darknet 2020 -datasetin luokittelusta sekä verrata näitä tuloksia Habibi Lashkarin ym. (2020) kehittämän mallin tuloksiin. Suurin ero pohjatutkimukseen on se, että tässä työssä datasettiä käsitellään sellaisenaan eikä sitä muuteta kuvaksi.

Tutkielmassa käydään ensiksi läpi darknet-termi, sen paikka internetin kerrosrakenteessa sekä mahdolliset käyttötarkoitukset. Luku 3 perehdyttää lukijan kone- ja syväoppimiseen sekä konvoluutioneuroverkkojen toimintaan ja arkkitehtuuriin. Luvussa 4 puolestaan syvennytään Habibi Lashkarin ym. (2020) tutkimuksessa ehdottamaan DeepImage-malliin, joka perustuu vahvasti konvoluutioneuroverkkojen toimintaan. Luku 5 kertoo enemmän tutkielman testausalgoritmeista ja -ympäristöstä sekä Darknet 2020 -datasetistä ja siitä, minkälaista dataa siinä on ja miten se on tuotettu. Luvussa 6 esitellään saadut tulokset sekä tehdään niiden pohjalta analyysi. Luku 7 toimii yhteenvetolukuna, jossa esitellään kertaalleen tutkimuskysymykset, tärkeimmät havainnot sekä pohditaan mahdollista jatkotutkimusta.

2 Darknet

Digitalisaation ja koko ajan laajenevan internetin käytön vuoksi tietoturva on alkanut kiinnostamaan ihmisiä yhä enemmän. Tämän luvun tarkoituksena on määrittää darknet-termi sekä kertoa pimeän verkon käyttötarkoituksista. Luvussa 2.1 määritetään darknet-termi sekä käydään läpi pimeän verkon suosion kasvua. Luvussa 2.2 perehdytään lyhyesti verkon kerrosmaiseen rakenteeseen ja siihen, minne darknet kyseisessä rakenteessa sijoittuu. Luvussa 2.3 puolestaan käsitellään pimeän verkon käyttötarkoituksia.

2.1 Darknet terminä ja sen suosion kasvu

Termi darknet, pimeä verkko, alkoi muodostumaan jo 1970-luvulla viitatakseen verkkoihin, jotka olivat eristetty The Advanced Research Projects Agency Network:ista (ARPANET) (Mirea, Wang ja Jung 2019, 104). Pimeällä verkolla tarkoitetaan World Wide Web:in (WWW) sisällä toimivia verkkosivuja, joita ei ole indeksoitu perinteisillä hakukoneilla ja joiden sisältö on tarkoituksella salattu (Weimann 2016, 40). Pääsyyn pimeään verkkoon tarvitaan erillinen ohjelmisto (Jardine 2015, 1). Nykypäivän merkittävimmät esimerkit kyseisistä ohjelmistoista ovat Tor, I2P ja Freenet (Moore ja Rid 2016).

Darknet ei saanut juurikaan sen laajempaa huomiota ennen kuin viranomaiset pidättivät Ross William Ulbrichtin lokakuussa 2013. Ulbricht oli helmikuussa 2011 ovensa avanneen darknet-verkkokaupan nimeltä Silk Road luoja ja operaattori. (Rudesill, Caverlee ja Sui 2015, 5) Markkinapaikka tarjosi mahdollisuuden myyjille ja ostajille käydä sähköistä kaupankäyntiä samaan tapaan kuin Amazon Marketplace (Soska ja Christin 2015, 33).

Sivuston käyttäjämäärä ja liikevaihto kasvoi todella nopeasti ja Silk Road ohitti suosiossa piakkoin muut darknetin laittomat verkkokaupat (Jardine 2015, 3). Silk Road:in suuri läpimurto perustui vahvemman anonymiteetin takaamiseen sen käyttäjille verrattuna muihin sivustoihin. Tämä anonymiteetti saavutettiin yhdistämällä hajautettu Bitcoin-maksujärjestelmä sekä Tor-verkon anonymiteettiominaisuudet, jotka tekevät asiakkaan ja palvelimen IP-osoitteista tuntemattomia niin toisilleen kuin myös ulkopuolisille tarkkailijoille. (Soska ja Christin 2015, 33)

2.2 Internetin kerrosrakenne

Rudesill ym. Rudesill, Caverlee ja Sui (2015, 6) mainitsee, että verkko voidaan esittä datan valtamerenä (engl. data ocean), ja että suurin osa käyttäjistä on vuorovaikutuksessa aaltoilevan, läpinäkyvän ja helposti navigoitavissa olevan pintaverkon (engl. surface web) kanssa. On arvioitu, että syvä verkko on noin 400-500 kertaa suurempi kuin pintaverkko (Rudesill, Caverlee ja Sui 2015, 4–6). Pintaverkosta voidaan käyttää myös termiä indeksoitu verkko, johon käyttäjät siis pääsevät käsiksi tavallisten hakukoneiden kautta (Kaur ja Randhawa 2020). Rudesill ym. Rudesill, Caverlee ja Sui (2015, 6) esittää, että on käytännössä mahdotonta arvioida syvän verkon oikeaa kokoa, mutta nykyään suurin hakukone Google on esimerkiksi indeksoinut vain 4-16% pintaverkosta.

Syväverkko (engl. deep web) sijaitsee pintaverkon alla ja sitä voidaan kutsua myös näkymättömäksi tai piilotetuksi verkoksi (Kaur ja Randhawa 2020). Darknet on osa syvää verkkoa, jonka verkkosivuja tavalliset hakukoneet ei pysty indeksoimaan eikä siten löytämään. Tavalliset hakukoneet ei pysty näkemään tai hakemaan sisältöä syväverkosta, koska nämä sivut eivät ole olemassa ennen kuin ne luodaan dynaamisesti tietyn haun seurauksena. Verkkosivu on löydettävissä tavallisilla hakukoneilla, jos se on staattinen ja on linkitetty muihin sivuihin. (Bergman 2001)

Kaikki darknetistä tuleva viestintä on epäilevää sen passiivisen kuunteluluonteen vuoksi, sillä se ainoastaan hyväksyy saapuvat paketit, mutta ei tue lähteviä paketteja. Eri darknetit vastaanottavat merkittävästi erilaista liikennettä riippuen seurantaan varatun IP-alueen koosta. Jopa darknet-verkon koko voi vaihdella yksittäisestä palvelimesta suurimpaan saatavilla olevaan IP-osoiteavaruuteen. (Habibi Lashkari, Kaur ja Rahali 2020, 1)

2.3 Darknetin käyttötarkoitukset

Mirea, Wang ja Jung (2019, 104) toteavat, että pimeää verkkoa voidaan käyttää monenlaiseen sosiaaliseen toimintaan eikä kaikki siitä ole välttämättä lainvastaista. Vaikka darknet saikin paljon mainetta etenkin laittomasta toiminnasta, on olemassa useita eri laillisia käyttötapoja. Jotkut tällaiset käyttötavat perustuvat meille tuttuihin käsitteisiin, kuten kuvien jakamiseen, hyödyntäen vain syvän verkon tarjoamaa tietoturvaa. Muut käyttötavat saattavat

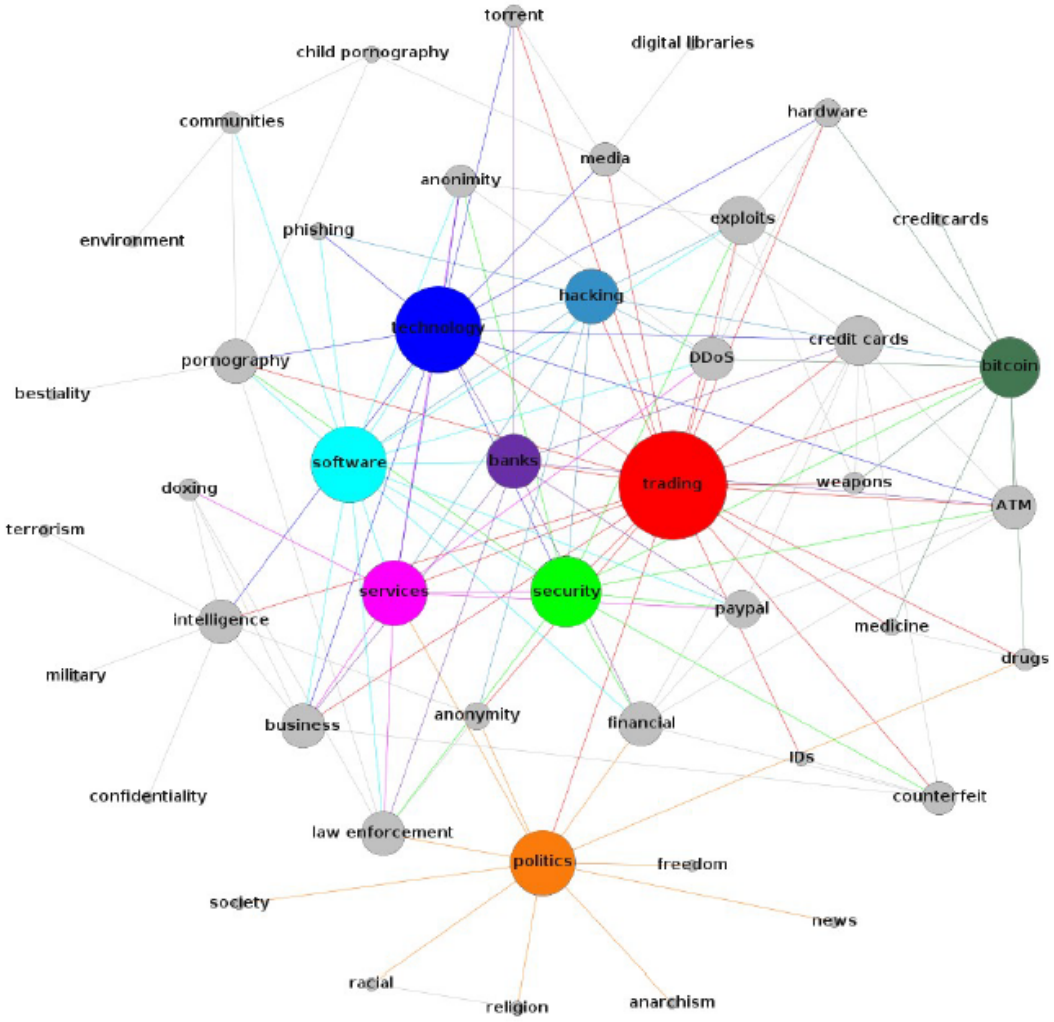
olla enemmän syvälle verkolle ominaisempia, kuten esimerkiksi suojatut ilmiantamissivustot. Myöskin toimittajat ovat käyttäneet SecureDrop-ohjelmistoa jakaakseen tiedostoja Tor-verkon kautta. (Rudesill, Caverlee ja Sui 2015, 9)

Darknet-toiminta voidaan jakaa kolmeen eri kategoriaan: (1) aktivismi, journalismi ja ilmiantaminen; (2) rikollinen toiminta darknet-verkkokaupoissa; ja (3) kyberturvallisuusuhat, mukaan lukien bottiverkot, haittaohjelmat ja kiristysohjelmat. Darknetin tarjoamaa anonymiteettiä käytetään sosiaalisiin ja poliittisiin tarkoituksiin. (Mirea, Wang ja Jung 2019, 104–105) Esimerkkitoimintoja ensimmäisestä kategoriasta: Tor-verkko mahdollistaa sensuurin kiertämisen sortavissa maissa sekä jossain määrin välttyä valtion ja yritysten valvonnan uteliailta katseilta (Jardine 2015, 5).

Esimerkkejä toisesta kategoriasta: monet Darknetin virtuaaliset kauppapaikat ovat erikoistuneet laittomien huumeiden myyntiin (Mirea, Wang ja Jung 2019, 105). Soska ja Christin (2015, 33) mainitsevat, että anonymiteetin rohkaisemana Silk Roadin myyjät ja ostajat kävivät enimmäkseen kauppaa huumeista ja salakuljetuksesta. Muita esimerkkejä toisen kategorian toiminnasta ovat identiteettivarkaudet ja luottokorttipetokset; arkaluonteisten tietojen vuotaminen; eksoottiset eläimet; asekauppa sekä palkkamurhat ja terrorismi. (Chertoff ja Simon 2015, 3–5)

Esimerkkejä kolmannesta kategoriasta: joissain darknet-verkkokaupoissa käydään kauppaa hakkerointityökaluilla, joita voidaan käyttää joko suoraan tai epäsuorasti hyökkäämiseen yrityksiä tai yksityishenkilöitä vastaan. Kyseisten haittaohjelmien tekijät ovat käyttäneet pimeää verkkoa kommunikointiin ja ideoiden vaihtamiseen. Darknet yhdessä Bitcoin-kryptovaluutan keksimisen kanssa on tarjonnut kannattavaa liiketoimintaa rikollisille. (Mirea, Wang ja Jung 2019, 106)

Darknet tarjoaa merkittäviä piilotettuja palveluita virtuaalisilla toreillaan (Habibi Lashkari, Kaur ja Rahali 2020, 1). Kuviosta 1, joka esittelee darknetin tarjoamien piilopalveluiden suhdetta, voidaan selkeästi nähdä, että kaikista suurin keskittymä liittyy kaupankäyntiin. Sitä seuraa teknologia toiseksi suurimpana keskittymänä ja sen jälkeen ohjelmisto-, turvallisuus-, palvelu- ja politiikkakeskittymät.



Kuvio 1. Darknetin tarjoamat piilopalvelut (Habibi Lashkari, Kaur ja Rahali 2020, 2)

Piratismi kukoisti verkon pimeällä puolella, koska piilotettuja palvelimia oli vaikeampi haastaa oikeuteen (Moore ja Rid 2016). Ainutlaatuisten ominaisuuksiensa, kuten anonymiteetin, virtuaalisten markkinapaikkojen ja kryptovaluutan käytön ansiosta, darknetissä voidaan tehdä helposti paljon rikollista toimintaa. Darknet ei ole kuitenkaan loppujen lopuksi yhteisö, jossa rikollisuus on normi. Se on vain teknologinen alusta, jota eri henkilöt voivat käyttää moniin eri tarkoituksiin. (Mirea, Wang ja Jung 2019, 114)

3 Tekoäly

Voidaksemme paremmin ymmärtää konvoluutioneuroverkkoja ja niiden toimintaa seuraavassa luvussa, on hyvä käydä läpi myös tekoälyn, kone- ja syväoppimisen käsitteet. Luvussa 3.1 perehdytään koneoppimisen käsitteeseen ja luvussa 3.2 käsitellään enemmän syväoppimista sekä kone- ja syväoppimisen eroja. Luku 3.3 avaa enemmän feature extraction -tekniikkaa, jota hyödynnetään Habibi Lashkarin ym. (2020) ehdottamassa mallissa. Luku 3.4 keskittyy konvoluutioneuroverkkojen rakenteeseen ja toimintaan.

3.1 Koneoppiminen

Koneoppimisen (engl. machine learning, ML) termin loi ensimmäisen kerran 50-luvulla IBM-tutkija nimeltä Arthur Samuel, ja sen oli tarkoitus kattaa monia älykkäitä toimintoja, jotka voidaan siirtää ihmiseltä koneelle (Guyon ym. 2008; Howard ja Gugger 2020). Käsite *kone* (engl. machine) tulee ymmärtää abstraktisti: ei fyysisesti instantoituna koneena, vaan automatisoituna järjestelmänä, joka voidaan toteuttaa esimerkiksi ohjelmistossa (Guyon ym. 2008, 1). Zhou (2021, 2) esittääkin, että koneoppiminen on tekniikka, joka parantaa järjestelmän suorituskykyä oppimalla kokemuksesta laskennallisten menetelmien avulla. Tietokonejärjestelmissä kokemus (engl. experience) on olemassa datan muodossa, ja koneoppimisen päätehtävänä on kehittää datasta malleja (engl. models) rakentavia oppimisalgoritmeja. Syöttämällä oppimisalgoritmiin kokemusdataa saadaan malli, joka voi ennustaa uusia havaintoja. (Zhou 2021, 2)

Jotta koneoppiminen voidaan suorittaa, meillä on ensin oltava dataa (Zhou 2021, 2). Datasetti on yksinkertaisesti tietojoukko – se voi olla kuvia, sähköposteja, taloudellisia indikaattoreita, ääniä tai mitä tahansa muuta (Howard ja Gugger 2020, 15). Koneoppimisen avulla opetetaan koneita käsittelemään dataa tehokkaammin. Monet toimialat soveltavat koneoppimista olennaisen tiedon poimimiseen. Koska saatavilla on runsaasti erilaisia tietojoukkoja, koneoppimisen kysyntä kasvaa koko ajan. (Mahesh 2020, 381)

Koneoppiminen turvautuu erilaisiin algoritmeihin dataongelmien ratkaisemiseksi. Datatieteilijät huomauttavat, että ei ole olemassa yhtä ainoaa kaikille sopivaa algoritmia, joka olisi

paras ongelman ratkaisemiseksi. Algoritmi riippuu ratkaistavana olevan ongelman tyypistä, muuttujien määrästä, sille parhaiten sopivasta mallista ja niin edelleen. (Mahesh 2020, 381)

Mahesh (2020, 381) jakaa koneoppimisen seruaaviin menetelmiin: ohjattu oppiminen (engl. supervised learning), ohjaamaton oppiminen (engl. unsupervised learning), puoliohjattu oppiminen (engl. semi-supervised learning), vahvistusopppiminen (engl. reinforcement learning), monitehtävä oppiminen (engl. multi-task learning), ryhmäoppiminen (engl. ensemble learning) ja neuroverkko (engl. neural network). Zhou (2021, 4) toisaalta esittää, että riippuen siitä, onko harjoitusdatasetti merkitty (engl. labeled) vai ei, oppimisongelmat voidaan karkeasti jakaa kahteen luokkaan: ohjattuun oppimiseen (esim. luokittelu ja regressio) ja ohjaamattomaan oppimiseen (esim. klusterointi).

Ohjatun oppimisen koneoppimisalgoritmit ovat niitä algoritmeja, jotka tarvitsevat ulkopuolista valvontaa. Syöttödatasetti jaetaan harjoitus- ja testidatasetteihin. Harjoitusdatasetissä on lähtömuuttuja, joka mallin on ennustettava tai luokiteltava. Kaikki algoritmit oppivat jonkinlaisia malleja harjoitusdatasetistä ja soveltavat niitä testiaineistoon ennustamista tai luokittelua varten. (Mahesh 2020, 381) Esimerkiksi tämän tutkielman testausalgoritmit kuuluvat ohjatun koneoppimisen alle. Ohjaamattomassa oppimisessa algoritmien annetaan itse löytää ja esittää suuressa datasetissä piilevä rakenne ilman tutkijan tai kehittäjän apua. Sitä käytetäänkin pääasiassa klusterointiin ja ominaisuuksien määrän vähentämiseen. (Mahesh 2020, 383)

3.2 Syväoppiminen

Syväoppiminen on vain moderni alue yleisemmällä koneoppimisen alalla. Syväoppiminen mahdollistaa useista prosessointikerroksista koostuvien laskennallisten mallien oppia datan esityksiä useilla abstraktiotasoilla. Syväoppiminen löytää monimutkaisen rakenteen suurista tietojoukoista käyttämällä backpropagation-algoritmia osoittamaan, kuinka koneen tulisi muuttaa sisäisiä parametrejaan, joita käytetään kunkin kerroksen esityksen laskemiseen edellisen kerroksen esityksestä. Nämä menetelmät ovat dramaattisesti parantaneet puheentunnistuksen, visuaalisen objektin tunnistuksen, objektien havaitsemisen ja monien muiden alojen, kuten lääkekehityksen ja genomiikan, huippua.(LeCun, Bengio ja Hinton 2015)

Nykyään tekoäly (engl. artificial intelligence, AI) on kukoistava ala, jolla on monia käytännön sovelluksia ja aktiivisia tutkimusaiheita. Tekoälyn todelliseksi haasteeksi osoittautuivat ne tehtävät, joiden suorittaminen on helppoa ihmisille, mutta joiden kuvaileminen muodollisesti koneille on erittäin hankalaa. Tällaisia ovat ongelmat, jotka me ratkaisemme intuitiivisesti ja jotka tuntuvat meistä tapahtuvan automaattisesti, kuten esimerkiksi puhuttujen sanojen tunnistaminen tai kasvojen tunnistus kuvista. (Goodfellow, Bengio ja Courville 2016, 1)

Ratkaisuna edellä mainittuihin intuitiivisiin ongelmiin on antaa tietokoneille mahdollisuus oppia kokemuksista ja ymmärtää maailmaa suhteessa käsitehierarkiaan, jossa jokainen käsite määritellään sen suhteessa muihin yksinkertaisempiin käsitteisiin. Keräämällä tietoa kokemuksista vältytään ihmisten tarpeesta muodollisesti määritellä kaikki se tieto, jota tietokone tarvitsee. Käsitteiden hierarkia puolestaan mahdollistaa tietokoneen oppivan monimutkaisia käsitteitä johtamalla ne yksinkertaisemmista käsitteistä. Jos me piirretään kaavio, joka kuvaa, kuinka nämä käsitteet rakentuvat toistensa päälle, se muodostuu syväksi ja monia eri kerroksia sisältäväksi. Tämän vuoksi kutsumme tätä tekoälyn lähestymistapaa syväoppimiseksi. (Goodfellow, Bengio ja Courville 2016, 1–2)

Kuvio 2 kuvaa Pattersonin ja Gibsonin (2017, 4) käsitystä tekoälyn, koneoppimisen ja syväoppimisen välisestä suhteesta. Siinä syväoppiminen nähdään koneoppimisen osajoukkona, joka puolestaan on tekoälyn osa-alue. Monille on ollut haaste määritellä syväoppiminen, koska se on muuttanut muotoaan hitaasti viime vuosikymmeninä.

Yksi määritelmistä sanoo, että syväoppiminen käsittelee neuroverkkoa (engl. neural network), jossa on enemmän kuin kaksi kerrosta. Tämän määritelmän ongelmana on, että se saa syväoppimisen kuulostamaan ikään kuin se olisi ollut olemassa jo 1980-luvulta lähtien. (Patterson ja Gibson 2017, 6) Pattersonin ja Gibsonin (2017, 6) mielestä neuroverkkojen täytyi arkkitehtuurillisesti ylittää aikaisemmat verkkotyylit (yhdessä paljon suuremman prosessointitehon kanssa), ennen kuin ne pystyivät esittelemään vaikuttavia tuloksia viime vuosilta. Seuraavassa on lueteltu joitain neuroverkkojen tämän kehityksen vaiheita:

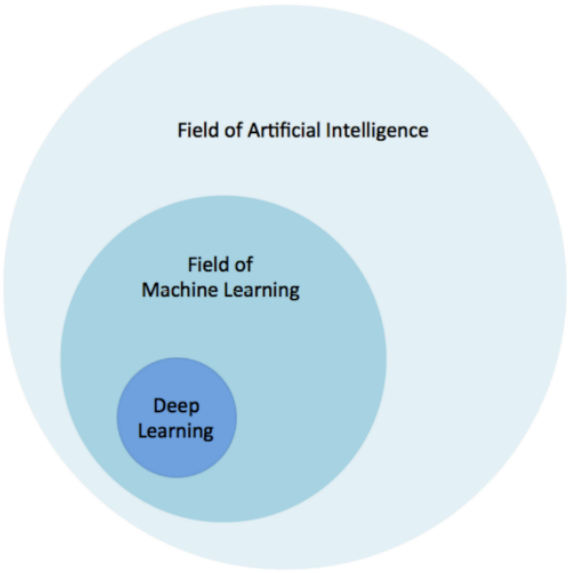
- Enemmän neuroneja kuin aiemmissa verkoissa,
- Monimutkaisempia tapoja yhdistää kerroksia tai neuroneja neuroverkoissa,

- Räjähdysmäinen kouluttamiseen käytettävissä olevan laskentatehon määrä ja
- Automaattinen piirteiden poiminta (engl. feature extraction).

Patterson ja Gibson (2017, 6) kuitenkin määrittelevät syväoppimisen heidän kirjassaan neuroverkoksi, jossa on suuri määrä parametreja ja kerroksia yhdessä neljästä perusverkkoarkkitehtuurista:

- Ohjaamattomat esikoulutetut neuroverkot,
- Konvoluutioneuroverkot,
- Toistuvat neuroverkot ja
- Rekursiiviset neuroverkot.

Syvät konvoluutioverkot ovat tuoneet läpimurtoja kuvien, videon, puheen ja äänen käsittelyssä, kun taas toistuvat verkot ovat selkeyttäneet peräkkäistä dataa, kuten tekstiä ja puhetta (LeCun, Bengio ja Hinton 2015). Syväoppiminen on ylittänyt perinteiset algoritmit tarkkuudessa lähes kaikissa tietotyypeissä vähäisellä hienosäädöllä ja inhimillisellä vaivalla (Patterson ja Gibson 2017, 6–7).



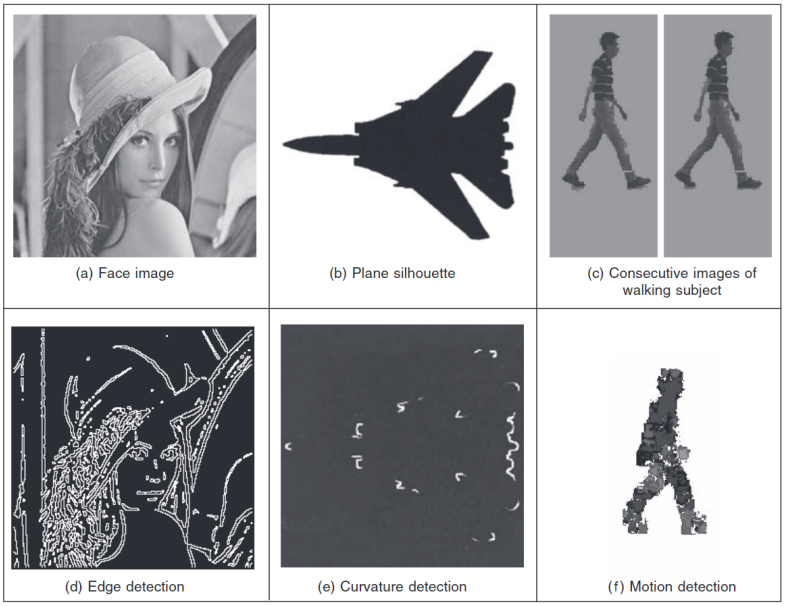
Kuvio 2. Visuaalinen havainnollistaminen tekoälyn, koneoppimisen ja syväoppimisen välisestä suhteesta (Patterson ja Gibson 2017, 4)

3.3 Feature extraction -tekniikka

Automaattinen piirteiden pomiminen (engl. feature extraction) on yksi syvän oppimisen suurista eduista perinteisiin koneoppimisalgoritmeihin nähden (Patterson ja Gibson 2017, 6). Piirteiden poiminnalla Patterson ja Gibson (2017, 6) tarkoittavat verkon prosessia, joka päättää, mitä tietojoukon (engl. dataset) ominaisuuksia voidaan käyttää indikaattoreina tietojen luotettavuuden merkitsemiseksi. Koneoppimisen ammattilaiset ovat käyttäneet paljon aikaa luomalla manuaalisesti tyhjentäviä ominaisuusjoukkoja (engl. feature sets) datan luokittelua varten. (Patterson ja Gibson 2017, 6)

Ominaisuuksilla (engl. feature) on erittäin tärkeä rooli kuvankäsittelyssä (engl. image processing). Hyvä ominaisuusjoukko sisältää erottelevaa tietoa, joka erottaa yhden objektin muista objekteista. Valitun ominaisuusjoukon tulee olla pieni joukko, jonka arvot erottavat tehokkaasti eri luokkien mallit toisistaan, mutta ovat kuitenkin samanlaisia saman luokan malleille. Ominaisuudet voidaan jakaa kahteen luokkaan: (1) lokaalit ominaisuudet, jotka ovat yleensä geometrisia (esim. koverat/kuperat osat, päätepisteiden lukumäärä, haarat, liitokset jne.); (2) globaalit ominaisuudet, jotka ovat yleensä topologisia (liitettävyys, projektioprofiilit, reikien määrä jne.) tai tilastollisia (esim. invarianttimomentit). (Kumar ja Bhatia 2014, 5–6)

Nixon ja Aguado (2002, 99) toisaalta jakavat ominaisuudet matalan tason ja korkean tason ominaisuuksiin. He määrittelevät matalan tason piirteet sellaisiksi perusominaisuuksiksi, jotka voidaan poimia kuvasta automaattisesti ilman muototietoja (tietoa tilasuhteista), kuten esimerkiksi kynnysarvoa (engl. tresholding). Sellaisenaan kynnysarvo on itse asiassa matalan tason piirteiden poiminnan muoto, joka suoritetaan pisteoperaationa (engl. point operation). Esimerkkejä matalan tason ominaisuuksista voidaan nähdä kuviossa 3: reunan tunnistus (engl. edge detection) (3 (a) ja (d)), kaarevuus (engl. curvature) (3 (b) ja (e)), ja optinen virtausarvo (engl. optical flow estimation) (3 (c) ja (f)). Luonnollisesti kaikkia näitä lähestymistapoja voidaan myös käyttää korkean tason piirteiden poiminnassa, jossa kuvista löydetään erilaisia muotoja. (Nixon ja Aguado 2002, 99–100)



Kuvio 3. Matalatasoinen ominaisuustunnistus (Nixon ja Aguado 2002, 101), muokattu

Korkean tason piirteiden poimiminen koskee muotojen löytämistä tietokonekuvista. Jotta esimerkiksi kasvot voidaan tunnistaa automaattisesti, yksi tapa on poimia komponenttien ominaisuudet. Tämä edellyttää kasvojen tärkeimpien piirteiden, kuten silmien, korvien ja nenän, poimimista. Näiden piirteiden löytämiseksi voimme käyttää hyväksi niiden muotoa: silmien valkoinen osa on ellipsoidinen; suu voi näkyä kahdella viivalla, kuten myös kulmakarvat. Muodon poimiminen tarkoittaa niiden sijainnin, suunnan ja koon löytämistä, joten monimutkaisemmat kuvat voidaan hajottaa yksinkertaisten muotojen rakenteeksi. (Nixon ja Aguado 2002, 161)

Ominaisuuksien poimimisessa etsitään yleensä invarianssiominaisuuksia (engl. invariance properties), jotta poimintaprosessi ei vaihtele valittujen tai määritettyjen olosuhteiden mukaan. Toisin sanoen, tekniikoiden tulisi löytää muodot luotettavasti ja vankasti riippumatta minkä tahansa parametrin arvosta, joka voi ohjata muodon ulkonäköä. Perusinvarianttina etsitään immuniteettia valaistustason muutoksille eli pyritään löytämään muoto, rippumatta siitä, onko se vaalea tai tumma. Periaatteessa niin kauan kuin muodon ja sen taustan välillä on kontrastia, muodon voidaan sanoa olevan olemassa ja se voidaan siten havaita. Valaistuksen jälkeen seuraavaksi tärkein parametri on sijainti: pyritään löytämään muodon missä tahansa se esiintyy. Tätä kutsutaan yleensä paikka-, sijainti- tai translaatioinvarianssiksi.

(Nixon ja Aguado 2002, 161)

Ennen ominaisuuksien saamista näytekuvaan sovelletaan erilaisia kuvan esikäsittelytekniikoita, kuten esimerkiksi binarisointia (engl. binarization), kynnysarvoa, koon muutosta ja normalisointia (engl. normalization). Tämän jälkeen ominaisuuksien poimintatekniikoita sovelletaan sellaisten ominaisuuksien saamiseksi, joista on hyötyä kuvien luokittelussa ja tunnistamisessa. Ominaisuuksien poimintatekniikat ovat hyödyllisiä erilaisissa kuvankäsittelysovelluksissa, kuten esimerkiksi hahmontunnistuksessa ja asiakirjojen aitouden vahvistuksessa. (Kumar ja Bhatia 2014, 5)

Ominaisuuksien poiminta kuvaa kuvion sisältämää oleellista muotoinformaatiota niin, että kuvion luokittelu on helpompaa muodollisen menettelyn avulla. Hahmontunnistuksessa ja kuvankäsittelyssä piirteiden poiminta on erityinen muoto dimensioiden vähentämiseksi. Ominaisuuksien poimimisen päätavoite on saada alkuperäisestä tiedosta olennaisimmat tiedot ja edustaa sitä matalamman ulottuvuuden avaruudessa. Syötteen (engl. input) muuntamista ominaisuusjoukoksi kutsutaan ominaisuuden poimimiseksi. Jos poimitut ominaisuudet valitaan huolellisesti, ominaisuusjoukon odotetaan poimivan tarvittavat tiedot syötteestä halutun tehtävän suorittamiseksi käyttämällä tätä supistettua esitystä täysikokoisen syötteen sijaan. (Kumar ja Bhatia 2014, 5)

Ominaisuuksien poiminta on tärkeä vaihe minkä tahansa malliluokituksen (engl. pattern classification) rakentamisessa ja sen tavoitteena on poimia kullekin luokalle ominaista tietoa. Tässä prosessissa olennaiset piirteet poimitaan objekteista piirrevektoreiden (engl. feature vectors) muodostamiseksi. Luokittimet (engl. classifiers) käyttävät sitten näitä piirrevektoreita tunnistamaan syöttöyksikön (engl. input unit) tulostusyksikön (engl. output unit) kanssa. Laajasti käytettyjä piirteiden poimintamenetelmiä ovat esimerkiksi template matching-, graph description-, contour profiles-, fourier descriptors- ja gradient feature-menetelmät. (Kumar ja Bhatia 2014, 5)

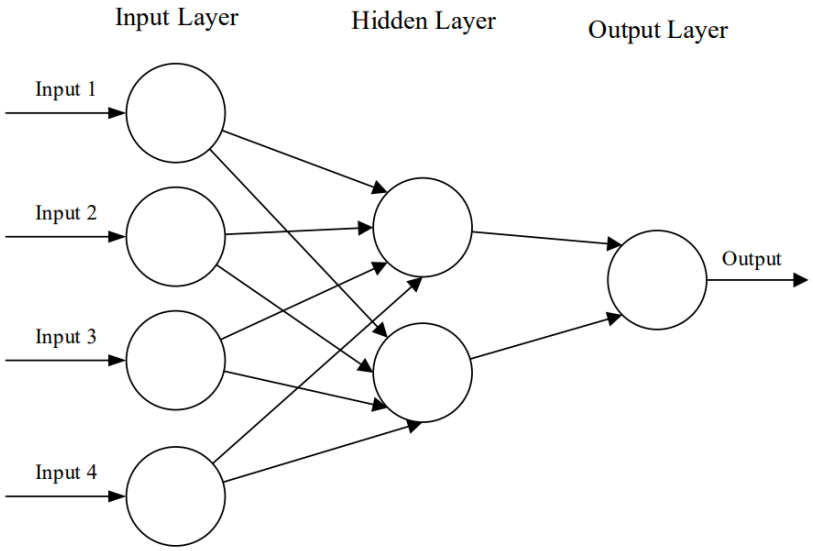
3.4 Konvoluutioneuroverkot

Koneoppimisen ala on viime aikoina saanut dramaattisen käänteen keinotekoisen neuroverkon (engl. Artificial Neural Network, ANN) nousun myötä. Nämä biologisesti inspiroi-

dut laskennalliset mallit pystyvät ylittämään aikaisempien tekoälymuotojen suorituskyvyn yleisissä koneoppimistehtävissä. (O'Shea ja Nash 2015, 1) Konvoluutioneuroverkko (engl. convolutional neural network, CNN) on tunnettu syväoppimisen arkkitehtuuri, joka on saanut inspiraationsa elävien olentojen luonnollisesta visuaalisesta havaintomekanismista (Gu ym. 2018). Erityisesti tehtävät, kuten kuvien luokittelu, riippuvat voimakkaasti konvoluutioneuroverkoista (Howard ja Gugger 2020, 41).

ANN:t koostuvat pääasiassa suuresta määrästä toisiinsa yhteydessä olevista laskennallisista solmuista, neuroneista. Syöte ladataan yleensä moniulotteisen vektorin muodossa syötekerrokseen (engl. input layer), joka jakaa sen piilotettuihin kerroksiin (engl. hidden layer). (O'Shea ja Nash 2015, 1) Monikerroksisen neuroverkon piilotetut kerrokset oppivat esittämään verkon syötteitä tavalla, joka helpottaa kohdetulosteiden (engl. target outputs) ennustamista (LeCun, Bengio ja Hinton 2015).

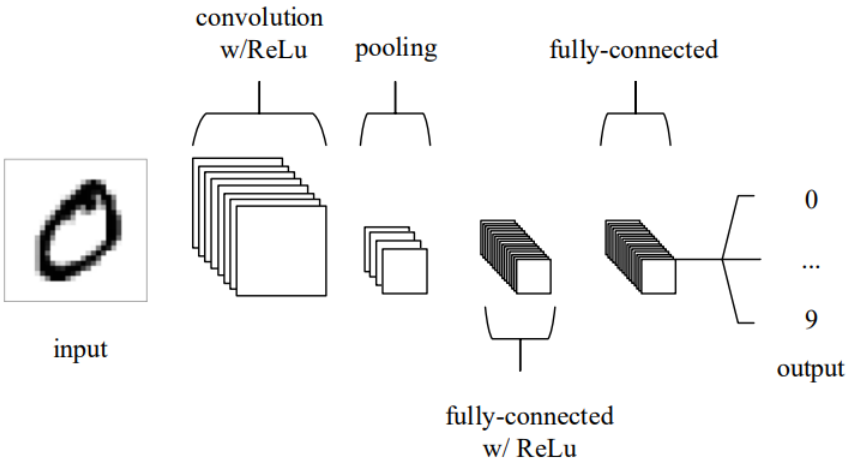
Kuviossa 4 näkyy kolmikerroksinen myötäkytkentäinen neuroverkko (engl. feedforward neural network, FNN), joka sisältää syötekerroksen, piilotetun kerroksen (engl. hidden layer) ja tulostekerroksen (engl. output layer). Tämä rakenne on useiden yleisten ANN-arkkitehtuurien perusta. (O'Shea ja Nash 2015, 2) Vuonna 1990 LeCun ym. julkaisi tärkeän paperin konvoluutioneuroverkon nykyaikaisen kehyksen luomisesta ja paransi sitä myöhemmin vuonna 1998. He kehittivät monikerroksisen keinotekoisen neuroverkon nimeltä LeNet-5, joka pystyi luokittelemaan käsin kirjoitettuja numeroita. (Gu ym. 2018)



Kuvio 4. ANN:n perusrakenteen malli (O'Shea ja Nash 2015, 2)

Kuten muissakin neuroverkoissa, LeNet-5:ssa on useita kerroksia, ja sitä voidaan opettaa backpropagation-algoritmilla. Se voi saada tehokkaita representaatioita alkuperäisestä kuvasta, mikä mahdollistaa visuaalisten kuvioiden (engl. pattern) tunnistamisen suoraan raakapikseleistä vähäisellä esikäsittelyllä. Kuitenkin suuren koulutusdatan ja laskentatehon puutteen vuoksi verkko ei pysty toimimaan hyvin monimutkaisemmissa ongelmissa, kuten suuren mittakaavan kuvien ja videoiden luokittelussa. (Gu ym. 2018)

Kirjallisuudessa on lukuisia muunnelmia CNN-arkkitehtuureista, mutta niiden peruskomponentit ovat kuitenkin hyvin samankaltaisia (Gu ym. 2018). Konvoluutioneuroverkot koostuvat kolmen tyyppisistä kerroksista: kovoluutiokerroksista (engl. convolution layer), poolauskerroksista (engl. pooling layer) ja täysin kytketyistä kerroksista (engl. fully-connected layer). Pinomalla nämä kerrokset muodostetaan CNN-arkkitehtuuri, joka esitetään kuviossa 5. (O'Shea ja Nash 2015, 4)



Kuvio 5. Yksinkertaistettu CNN-arkkitehtuuri, joka koostuu viidestä kerroksesta (O'Shea ja Nash 2015, 4)

Konvoluutioneuroverkot on suunniteltu käsittelemään dataa, joka tulee usean taulukon muodossa, kuten esimerkiksi värikuva, joka koostuu kolmesta 2D-taulukosta, jotka sisältää pikseliintensiteetit kolmessa värikanavassa. Konvoluutioneuroverkkojen taustalla on neljä avainideaa, jotka hyödyntävät luonnollisten signaalien omianisuuksia: paikalliset yhteydet (engl. local connections), jaetut painot (engl. shared weights), poolaus (engl. pooling) ja useampien kerrosten käyttö. (LeCun, Bengio ja Hinton 2015, 439) Monet datamodaliteetit ovat useiden taulukoiden muodossa, kuten esimerkiksi:

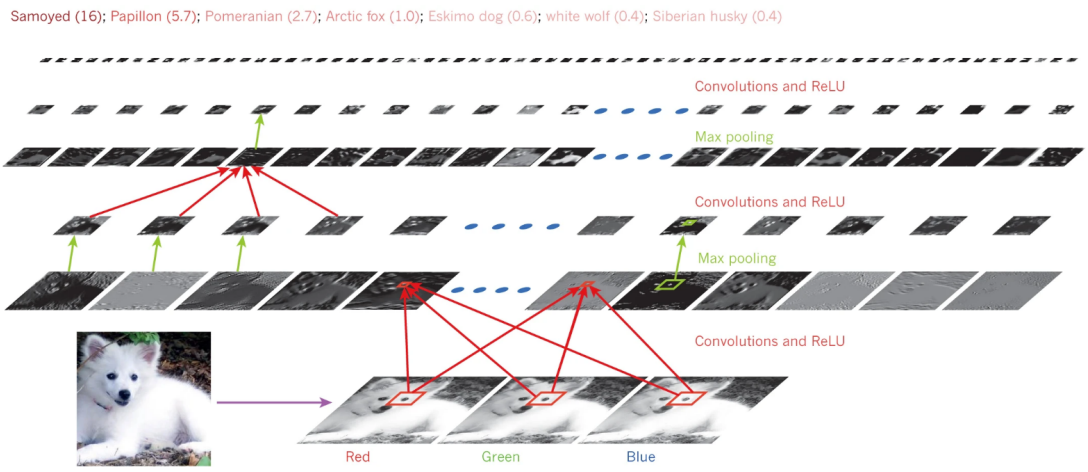
- 1D signaaleille ja sarjoille, mukaan lukien kieli,
- 2D kuville tai äänispektrogrammeille ja
- 3D videoille tai tilavuuskuville.

Tyypillisen konvoluutioneuroverkon arkkitehtuuri on jäsennetty sarjaksi vaiheita. Muutamat ensimmäiset vaiheet koostuvat kahden tyyppisistä kerroksista: konvoluutiokerroksista ja poolauskerroksista. Konvoluutiokerroksen yksiköt ovat järjestetty ominaisuuskartoiksi, joissa jokainen yksikkö on yhdistetty edellisen kerroksen piirrekarttojen paikallisiin patcheihin painosarjan kautta, jota kutsutaan suodatinpankiksi. Tämän paikallisen painotetun summan tulos johdetaan sitten epälineaarisuuden, kuten ReLU:n, läpi. ReLU (engl.rectified linear

unit) tarkoittaa tasasuuntautunutta lineaarista yksikköä. (LeCun, Bengio ja Hinton 2015)

Kaikilla ominaisuuskartan yksiköillä on sama suodatinpankki, mutta tason eri piirrekartat käyttävät erilaisia suodatinpankkeja. Syy tälle arkkitehtuurille on kaksijakoinen. Ensinnäkin, taulukkotiedoissa, kuten kuvissa, paikalliset arvoryhmät korreloivat usein voimakkaasti ja muodostavat erottuvia paikallisia motiiveja, jotka on helppo havaita. Toiseksi, kuvien ja muiden signaalien paikalliset tilastot vaihtelevat sijainnin mukaan. Toisin sanoen, jos motiivi voi esiintyä yhdessä kuvan osassa, se voi esiintyä missä tahansa, mistä johtuu ajatus eri paikoissa olevista yksiköistä, jotka jakavat saman painoarvon ja havaitsevat saman kuvion taulukon eri osissa. Matemaattisesti piirrekartan suorittama suodatustoiminto on diskreetti konvoluutio, mistä konvoluutioneuroverkon nimi johtuu. (LeCun, Bengio ja Hinton 2015)

Kuvio 6 kuvaa tyypillisen konvoluutioverkon jokaisen kerroksen (vaakasuunnassa) tulosteita sovellettuna samojedikoiran kuvaan (vasen alakulma, ja RGB-syötteet (engl. red, green, blue) alhaalla oikealla). Jokainen suorakulmainen kuva on piirrekartta (engl. feature map), joka vastaa yhden opitun ominaisuuden tulostetta, joka on havaittu kussakin kuvan sijainnissa. Tieto virtaa alhaalta ylöspäin, ja alemman tason ominaisuudet toimivat suunnattuina reunatunnistimina, ja jokaiselle tulostettavaksi tulevalle kuvaluokalle lasketaan pisteet. (LeCun, Bengio ja Hinton 2015)



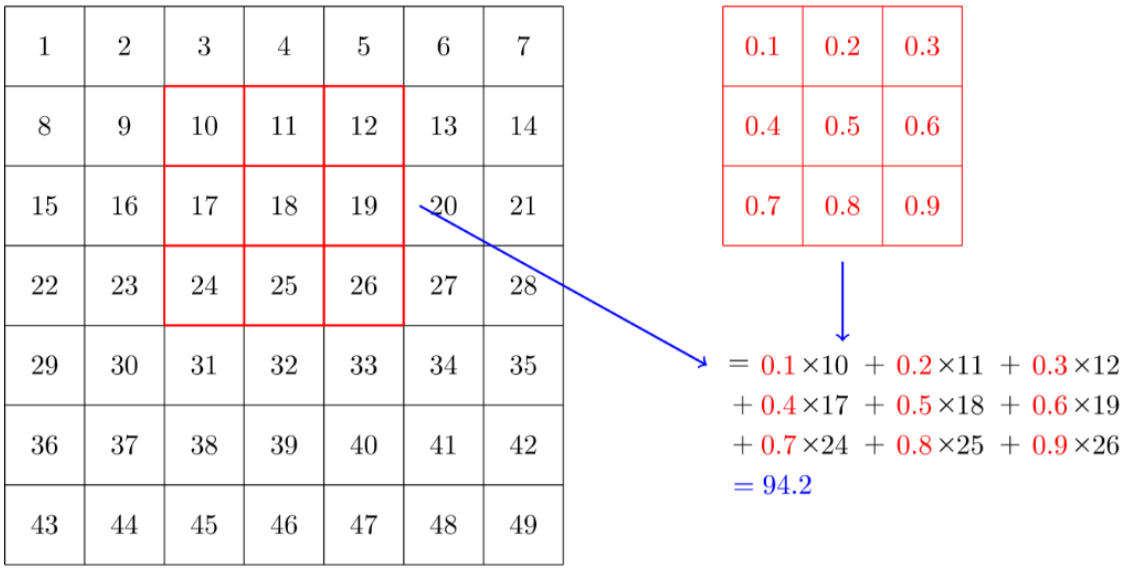
Kuvio 6. Konvoluutioneuroverkon toiminta (LeCun, Bengio ja Hinton 2015)

Vaikka konvoluutiokerroksen tehtävänä on havaita edellisen kerroksen piirteiden paikalli-

set konjunktiot, poolauskerroksen tehtävänä on yhdistää semanttisesti samanlaiset piirteet yhdeksi. Koska motiivin muodostavien piirteiden suhteelliset paikat voivat vaihdella jonkin verran, motiivin luotettava havaitseminen voidaan tehdä karkearakeisella kunkin piirteen sijainnilla. Tyypillinen poolausyksikkö laskee paikallisen yksikkökorjauksen maksimin yhdessä karttakohdekartassa (tai muutamassa karttakohdekartassa). Viereiset poolausyksiköt ottavat syötteen patcheista, joita on siirretty useammalla kuin yhdellä rivillä tai sarakkeella, mikä vähentää esityksen ulottuvuutta ja luo invarianssin pienille siirtymille ja vääristymille. Kaksi tai kolme konvoluutio-, epälineaarisuus- ja poolausvaihetta pinotaan, minkä jälkeen tulevat konvoluutioisemmat ja täysin yhdistetyt kerrokset. (LeCun, Bengio ja Hinton 2015)

Syvät neuroverkot hyödyntävät sitä ominaisuutta, että monet luonnolliset signaalit ovat koostumushierarkioita, joissa korkeamman tason piirteitä saadaan muodostamalla alemman tason piirteistä. Esimerkiksi kuvissa paikalliset reunojen yhdistelmät muodostavat motiiveja, motiivit kootaan osiin ja osat muodostavat objekteja. Samanlaisia hierarkioita esiintyy puheessa ja tekstissä äänistä puhelimiin, foneemiin, tavuihin, sanoihin ja lauseisiin. Poolaus sallii representaation vaihdella hyvin vähän, kun edellisen kerroksen elementit vaihtelevat sijainniltaan ja ulkonäöltään. (LeCun, Bengio ja Hinton 2015)

Konvoluutiokerroksen tavoitteena on oppia syötteiden piirreesitykset. Konvoluutiokerros koostuu useista konvoluutiokerneleistä, joita käytetään erilaisten piirrekarttojen laskemiseen. (Gu ym. 2018) Kerneli on pieni matriisi, kuten 3×3 -matriisi kuvan 7 oikeassa yläkulmassa. Konvoluutio ei vaadi muuta kuin kertomista ja yhteenlaskua – kaksi operaatiota, jotka vastaavat suurimmasta osasta työstä. (Howard ja Gugger 2020, 404)



Kuvio 7. Kernelin asettaminen yhteen paikkaan (Howard ja Gugger 2020, 404)

Uusi piirrekartta voidaan saada konvoloimalla ensin syöte opitun kernelin kanssa ja sitten soveltamalla elementtikohtaista epälineaarista aktivointifunktiota konvoloituneisiin tuloksiin (Gu ym. 2018). Gu ym. (2018) painottaa, että jokaisen ominaisuuskartan luomiseksi kerneli jaetaan kaikkien syötteen spatiaalisten sijaintien kesken. Valmiit ominaisuuskartat saadaan käyttämällä useita eri kerneleitä. Matemaattisesti piirrearvo sijainnissa (i, j) l:nnen kerroksen k:nnessa piirrekartassa, $z^l_{i,j,k}$, lasketaan seuraavasti:

$$z^l_{i,j,k} = \mathbf{w}^{l\,T}_k \mathbf{x}^l_{i,j} + b^l_k,$$

missä $\mathbf{w}^l_k$ ja b^l_k ovat vastaavasti l:nnen kerroksen k:nnen suodattimen painovektori ja biastermi, ja $\mathbf{x}^l_{i,j}$ on syöttöpaikka, joka on keskitetty l:nnen kerroksen sijaintiin (i, j). Huomattavaa on, että ominaisuuskartan luova kerneli $\mathbf{w}^l_k$ on jaettu. Tällaisella painonjakomekanismilla on useita etuja, se esimerkiksi voi vähentää mallin monimutkaisuutta ja helpottaa verkon kouluttamista. (Gu ym. 2018)

Aktivointifunktio tuo konvoluutioneuroverkolle epälineaarisuutta, joka on toivottavaa monikerroksisissa verkoissa epälineaaristen piirteiden havaitsemiseksi. Olkoon a(·) epälineaarinen aktivointifunktio. Konvoluutioominaisuuden $z^l_{i,j,k}$ aktivointiarvo $a^l_{i,j,k}$ voidaan laskea

seuraavasti:

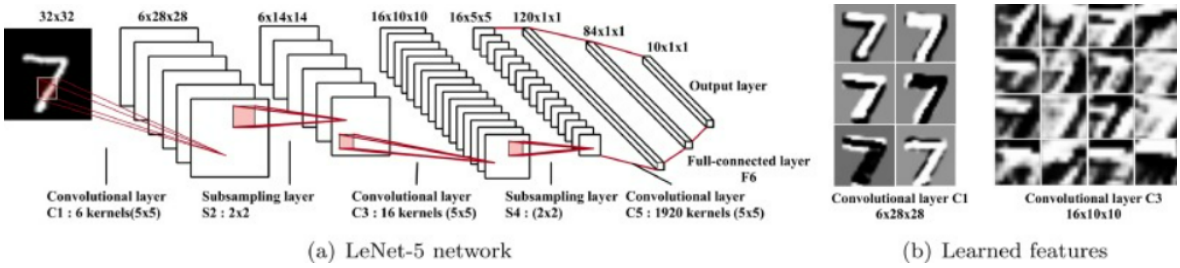
$$a^l_{i,j,k} = a\left(z^l_{i,j,k}\right).$$

Tyypillisiä aktivointifunktioita ovat sigmoid, tanh ja ReLU. Poolauskerroksen tavoitteena on saavuttaa muutosinvarianssi vähentämällä piirrekarttojen resoluutiota. Se sijoitetaan yleensä kahden konvoluutiokerroksen väliin. Kukin poolauskerroksen piirrekartta on yhdistetty sitä vastaavaan edellisen konvoluutiokerroksen piirrekarttaan. Merkitsemällä poolausfunktion pool(·) ominaisuuskartalle, meillä on:

$$y^l_{i,j,k} = pool\left(a^l_{m,n,k}\right), \forall(m,n) \in \Re_{ij},$$

missä $\Re_{ij}$ on paikallinen naapurusto sijainnin (i, j) lähellä. Tyypillisiä poolaustoimintoja ovat average pooling ja max pooling. (Gu ym. 2018)

Kuvassa 8 (a) on havainnollistettu LeNet-5-verkon arkkitehtuuri, joka toimii hyvin numeroluokittelutehtävässä. Kuvassa 8 (b) on puolestaan LeNet-5-verkon ominaisuuksien visualisointi, jossa kunkin tason piirrekartat näytetään eri lohkossa. Ensimmäisen konvoluutiokerroksen kernelit on suunniteltu havaitsemaan matalan tason piirteitä, kuten reunoja ja käyriä, kun taas korkeampien kerrosten kernelit on opetettu koodaamaan abstrakteja piirteitä. Pinoamalla useita konvoluutio- ja poolauskerroksia voisimme vähitellen poimia korkeamman tason ominaisuusrepresentaatioita. (Gu ym. 2018)



(a) LeNet-5 network (b) Learned features

Kuvio 8. LeNet-5 -verkko (a) ja ominaisuuskartat numerosta 7 (b), jotka on opittu kahdella ensimmäisellä konvoluutiokerroksella. (Gu ym. 2018)

Useiden konvoluutio- ja poolauskerrosten jälkeen voi olla yksi tai useampi täysin yhdistetty kerros, joka pyrkii suorittamaan korkean tason päättelyn. Täysin yhdistetyt kerrokset ottavat kaikki edellisen kerroksen neuronit ja yhdistävät ne nykyisen kerroksen jokaiseen yksittäiseen neuroniin globaalin semanttisen tiedon tuottamiseksi. (Gu ym. 2018)

Viimeinen konvoluutioneuroverkon kerros on output-kerros. Luokittelutehtävissä käytetään yleisesti softmax-operaattoria. Olkoon θ kaikki konvoluutioneuroverkon parametrit, esimerkiksi painovektorit ja bias-termit. Tietyn tehtävän optimaaliset parametrit voidaan saada minimoimalla kyseiselle tehtävälle määritetty sopiva häviöfunktio. Oletetaan, että meillä on N haluttua tulo-lähtösuhdetta (engl. input-output relations) $\{ (x^{(n)}, y^{(n)}) n \in [1,...,N] \}$. CNN:n häviö voidaan laskea seuraavasti:

$$\mathscr{L} = \frac{1}{N} \sum_{n=1}^{N} l(\theta; y^{(n)}, o^{(n)}),$$

missä $x^{(n)}$ on n:s syöttötieto, $y^{(n)}$ on sitä vastaava kohdenimike (engl. target label) ja $o^{(n)}$ on konvoluutioneuroverkon tuloste. (Gu ym. 2018)

Konvoluutioneuroverkon kouluttaminen on globaalin optimoinnin ongelma. Minimoimalla häviöfunktion voimme löytää parhaiten sopivat parametrit. Stokastinen gradienttilasku on yleinen ratkaisu CNN-verkon optimointiin. (Gu ym. 2018)

4 Aiempi tutkimus

Tämän luvun tarkoituksena on kertoa aiemmasta tutkimuksesta, joka toimii gradun pääläh-teenä ja jonka datasettiä olisi tarkoitus testata muita koneoppimismalleja käyttäen. Gradun pohjana toimii Kanadan New Brunswick yliopiston tutkijoiden Habibi Lashkarin, Kaurin ja Rahalin (2020) tekemä artikkeli *DIDarknet: A Contemporary Approach to Detect and Characterize the Darknet Traffic using Deep Image Learning*. Gradun päälähteen tukena toimii myös tutkijoiden GitHub-repositorio, joka sisältää heidän tutkimukseen liittyvää koodia (Warren ym. 2022).

4.1 Tutkimuksen taustaa ja motivointia

Habibi Lashkarin ym. (2020) mukaan vaikka darknet-liikenteen luokitteluun on tehty merkittäviä yrityksiä, jotka pohjautuvat voimakkaasti jo olemassa oleviin datasetteihin ja koneoppimisluokittelijoihin (engl. machine learning classifiers), on erittäin vähän yrityksiä havaita ja luokitella darknet-liikennettä syväoppimisen avulla. Tutkijoiden julkaisu ehdottaa uudenlaista lähestymistapaa nimeltä DeepImage, joka käyttää feature selection -tekniikkaa valitakseen datasetin tärkeimmät ominaisuudet harmaakuvan luomiseksi. Tämän jälkeen syntynyt kuva syötetään kaksiulotteiseen konvoluutioneuroverkkoon havaitakseen ja karakterisoidakseen darknet-liikennettä.

Darknet-liikenteen analysointi auttaa haittaohjelmien varhaisessa seurannassa ennen hyökkäystä ja haitallisen toiminnan havaitsemisessa hyökkäyksen jälkeen. Seuraavat motivaatiot johtivat Habibi Lashkarin ym. (2020) tutkimukseen:

- Salatun liikenteen havaitsemiseksi on tehty merkittäviä yrityksiä, jotka kattaa joko VPN- tai Tor-liikenteen erikseen. Yksikään tutkimus ei ole yrittänyt yhdistää VPN- ja Tor-liikenteen yhdeksi datasetiksi, joka kattaisi laajan valikoiman kaapattuja sovelluksia ja darknetin tarjoamia piilopalveluja.
- Darknet-liikenteen tunnistaminen yleisesti, ja erityisesti piilopalveluiden tunnistaminen, on välttämätöntä lainvastaisen toiminnan torjumiseksi ennen kuin se tapahtuu.

Seuraavat ovat Habibi Lashkarin ym. (2020) julkaisun tärkeimmät kontribuutiot:

1. Suurin osa olemassa olevista tekniikoista keskittyy darknet-liikenteen luokitteluun ja vain harvat niistä korostavat anonymisoitua VPN- tai Tor-liikennettä erikseen. Habibi Lashkarin ym. (2020) julkaisu esittää uuden tekniikan VPN- ja Tor-sovelluksien havaitsemiseksi ja karakterisoimiseksi yhdessä todellisena darknet-liikenteen edustajana.
2. Yhdistetään kaksi julkista datasettiä täydellisen darknet-datasetin luomiseksi, joka kattaa VPN- ja Tor-liikenteen.
3. Kaksiulotteisen konvoluutioneuroverkon tehokkuuden osoittaminen darknet-liikenteen havaitsemisessa ja karakterioimisessa suurella havaitsemisnopeudella, joka puolestaan kannustaa tunnistamaan monipuoliset ja epäilyttävät piilopalvelut.

4.2 Saatavilla olevien darknet-liikenteen datasettien analysointi

Yksi tärkeimmistä mallin kouluttamisen vaatimuksista on edustavan ja kattavan julkisen datasetin saatavuus. Habibi Lashkari ym. (2020) esittelevät julkaisussaan julkisesti saatavilla olevia datasettjä luokitteluineen.

DARPA - MIT Lincoln Laboratory (1998-99): Tämä on yksi tavanomaisista dataseteistä, joka kattaa 27 hyökkäysluokkaa ja normaalin taustadatan (engl. backround data), joita on kerätty seitsemän viikon aikana. Datasetti kaappasi muun muassa FTP-, telnet-, SNMP- ja selaustoimintoja sekä Nmap-, syn flood-, buffer overflow-, ja denial of service-hyökkäyksiä. Se sisältää verkkoliikenteen ja tarkastuslokit (engl. audit logs), jotka on kerätty simuloidussa verkossa ja näin ollen siitä puuttuu reaaliaikainen hyökkäysliikenne. Datasettiä arvioidaan sekä verkossa (Air Force Research Lab) että offline-tilassa (simuloitu verkko).

CTU-13 - Czech Technical University (2011): Valjastettiin keräämään reaaliaikaista botnet-liikennettä sekä tausta- ja normaaliliikennettä. Tämä datasetti sisältää 13 haittaohjelmaliikenneskenaariota vastaten eri botnet-näytteitä. Haittaohjelmaliikenne koostuu kaksisuuntaisista virroista, jotka on kaapattu suorittamalla tietty haittaohjelma Windows-virtuaalikoneessa ja tallentamalla palvelimella tuotettu verkkoliikenne. Netflow-tiedostot tallentavat jakelun erilaiset botnet-virrat, C&C-virrat, taustavirrat ja normaalit virrat.

Malware Capture Facility Project - Czech Technical University (2013): Tämän datasetin luomisen tavoitteena oli kaapata, analysoida ja julkaista todellista haittaohjelmien verkkoliikennettä joissakin tapauksissa useita kuukausia. Windows-virtuaalikoneita isännöidään Linux-koneilla suorittamaan haittaohjelmia DDoS:n ja roskapostin estämiseksi. Liikenne on merkitty (engl. labelled) käytön helpottamiseksi.

Anon17 - NIMS Lab (2014-17): Tämä datasetti koostuu kolmesta anonymiteettityökalusta: Tor, I2P ja JonDonym. Todellisessa verkkoympäristössä kaapattu datasetti on merkitty valitun anonyymiteettipalvelun saatavilla olevien tietojen perusteella. Se sisältää Tor-, TorApp-, TorPT-, I2PApp80BW-, I2PApp0BW-, I2PUsers-, I2PApp- ja JonDonym-dataa.

ISCXVPN2016 - ISCX (2016): Kaappasi reaaliaikaista VPN-liikennettä eri sovelluksille, kuten selaus-, chat-, tiedostonsiirto-, sähköposti-, suoratoisto-, VOIP- ja P2P-sovelluksille Wireshark:n ja TCPdump:n avulla. Salatun liikenteen luomiseen, joka oli merkitty käytön helpottamiseksi, käytettiin ulkoista VPN-palveluntarjoajaa.

ISCXTor2017 - ISCX (2017): Kaappasi reaaliaikaista Tor-liikennettä eri sovelluksille, kuten selaus-, chat-, FTP-, sähköposti-, VOIP- ja P2P-sovelluksille sekä ääni- ja videosuoratoistosovelluksille Wireshark:n ja TCPdump:n avulla. Datasetti on merkitty käytön helpottamiseksi.

Darknet Usage TextAddress (DUTA)-10K - GVIS Lab (2019): Sisältää 25 kategoriaa laillista ja laitonta toimintaa yli 10 367 manuaalisesti merkittylle onion domainille. Tor-verkon uusimpien piilotettujen toimintojen kanssa se esitteli CryptoLocker-lunnasohjelman, joka oli levinnyt laajasti WannaCry lunnasohjelman jälkeen.

4.2.1 Datasetin arviointikriteerit

Habibi Lashkari ym. (2020) määrittelevät kuusinkertaiset arviointikriteerit julkisesti saatavilla olevien salatun liikenteen tai darknet-liikenteen datasettien vertailua varten:

1. **Covering Different Connections (CDC)**: Ensimmäinen ja tärkein kriteeri on, sisältääkö datasetti Tor-liikennettä (T), VPN-liikennettä (V) tai Tor over VPN -liikennettä (TV).

2. **Complete Traffic (CT)**: Se sisältää erilaisia protokollia (diversity of protocols, DP) ja erilaisia sovelluksia (diversity of applications, DA), joita käytetään täydellisen verkkoliikenteen kaappaamiseen.
3. **Complete Interaction (CI)**: Se kattaa täydellisen vuorovaikutuksen eri protokollilla erilaisen datan lähettämistä ja vastaanottamista varten, kuten audio (A), video (V), tiedostonsiirto (FT), teksti/chat (T), sähköposti (E), VoIP (Vo), nettiselailu (B) ja P2P (P2).
4. **Complete Capture (CC)**: Kaappaamalla headerin (H) ja salatun tietosisällön (engl. encrypted payload) (P) ilman anonymisointia (A) taataan, että datasetti pysyy tutkijoille läpinäkyvänä paljastamalla kaikki kaapattu tieto.
5. **Feature set (FS)**: Se edustaa datasettiin tallennettuja ominaisuuksia. Habibi Lashkari ym. (2020) luokittelevat tämän kriteerin header- ja payload-ominaisuuksiksi synkronoidaakseen yllä mainitun täydellisen datan kaappauksen kanssa.
6. **Metadata (M)**: Se tarkoittaa, että datasetin tiedot, kuten kaapattu verkkoliikenne, hyökkäysskenaario, protokollien tyyppi jne. ovat saatavilla.

Näiden kriteerien perusteella tutkijat esittävät seitsemän datasetin yksityiskohtaisen vertailun kuviossa 9. Suurin osa aineistoista kattoi protokollien ja sovelluksien monimuotoisuuden eikä suosinut kaapatun liikenteen anonymisointia. Vaikka datasetit sisältää salattua tai anonymisoitua liikennettä, mikään edellä mainituista dataseteistä ei ole täydellinen tavalla tai toisella darknet-liikenteen tunnistukseen ja karakterisointiin, paitsi ISCXVPN2016 ja ISCX-Tor2017, jotka täyttävät useimmat määritellyt kriteerit.

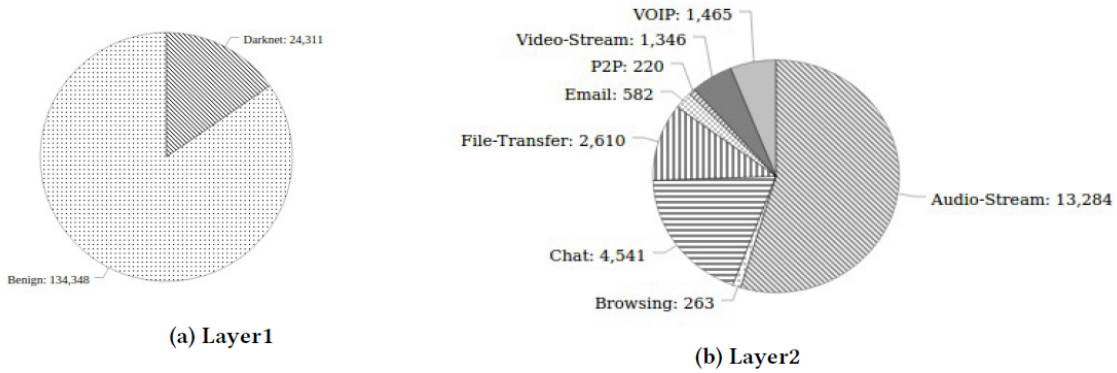
Dataset	CDC			CT		CI								CC			FS		M
	Tor	VPN	TV	DP	DA	A	V	FT	T	E	Vo	B	P2	H	P	A	H	P	
DARPA	N	N	N	Y	N	N	N	N	N	Y	N	Y	N	Y	N	N	Y	N	Y
CTU-13	N	N	N	N	N	N	N	N	Y	Y	N	N	Y	Y	N	N	Y	N	Y
MCFP	N	N	N	N	N	N	N	N	Y	Y	N	N	Y	Y	N	N	Y	N	Y
Anon17	Y	N	N	Y	Y	N	N	N	Y	N	N	N	N	Y	Y	N	Y	Y	Y
ISCXVPN2016	N	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
ISCXTor2017	Y	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
DUTA-10K	Y	N	N	Y	Y	N	N	N	N	N	N	Y	N	N	-	-	-	-	N

Kuvio 9. Datasettien arviointi (Habibi Lashkari, Kaur ja Rahali 2020, 6)

4.3 Valittu datasetti

Datasettien arviointikriteerien tulosten perusteella Habibi Lashkari ym. (2020) valitsivat datasetit ISCXVPN2016 ja ISCXTor2017. Molemmat datasetit kaappasivat tavallista verkkoliikennettä sekä VPN- ja Tor-liikennettä seitsemää erilaista kategoriaa varten vastaavissa sovelluksissa: nettiselaaminen (Firefox ja Chrome), chatti (ICQ, AIM, Skype, Fcebook ja Hangouts), sähköposti (SMTPS, POP3S ja IMAPS), tiedostonsiirto (Skype, FTP over SSH (SFTP) ja FTP over SSL (FTPS) käyttänen Filezilla-ohjelmistoa ja ulkopuolista palvelua), suoratoisto (Vimeo ja Youtube), VoIP (Facebook, Skype ja Hangouts äänipuhelut) ja P2P (uTorrent ja Transmission (BitTorrent)).

Valitut datasetit yhdistettiin luodaakseen uuden kaksikerroksisen verkkoliikennedatasetin, joka nimettiin darknet-datasetiksi. Darknet-datasetin ensimmäinen kerros on merkattu hyvänlaatuiseksi edustamaan tavallista liikennettä. Toinen kerros on merkattu darknetiksi edustamaan anonymisoitua (Tor tai VPN) liikennettä liittyen darknetin tarjoamiin piilopalveluihin. Suoratoistoliikenne erotettiin ääneen ja videoon toisen kerroksen merkkausta varten tehdäkseen kahdeksan kategoriaa. Kuvio 10 esittää eri protokollaliikennettä sekä tietueiden lukumäärää darknet-datasetissä. Darknet-datasetti koostuu yhteensä 158 659 tietueesta. Siinä on 134 348 hyvänlaatuista näytettä ja 24 311 darknet-näytettä. Ääni-suoratoistolla on eniten näytteitä, 13 284, kun taas vähiten näytteitä on kaapattu P2P-protokollalle.



Kuvio 10. Piilopalvelupohjaisen toiminnan määrä (Habibi Lashkari, Kaur ja Rahali 2020, 6)

Datasettiin tallennettujen toimintojen analysoimiseksi tutkijat suunnittelivat viestintäkaavion mallin yksilöllisten IP-osoitteiden lähde- ja kohdeparien välillä Gephillä, kuten kuviossa 11 näkyy. Tämän suunnatun graafin intensiteetti sallii äärimmäisen viestinnän, joka tapahtui eri hostien välillä datasettiä luotaessa. Graafi korostaa vain 10 parasta dataa lähettävää lähdekonetta kohdekoneiden enimmäismäärään. Väitetään, että suurimmalla osalla koneista on yksityinen IP-osoite, kun taas vain yksi julkinen IP-osoitepalvelin (engl. address host) (131.202.240.150) on mukana. Tutkimalla syvemmälle nähdään, että jotkut yksityiset palvelimet käyttivät osoitetta 131.202.240.150 yhtenä välikoneista siirrettäessään tietoja toiselle yksityiselle palvelimelle. (Habibi Lashkari, Kaur ja Rahali 2020, 5)

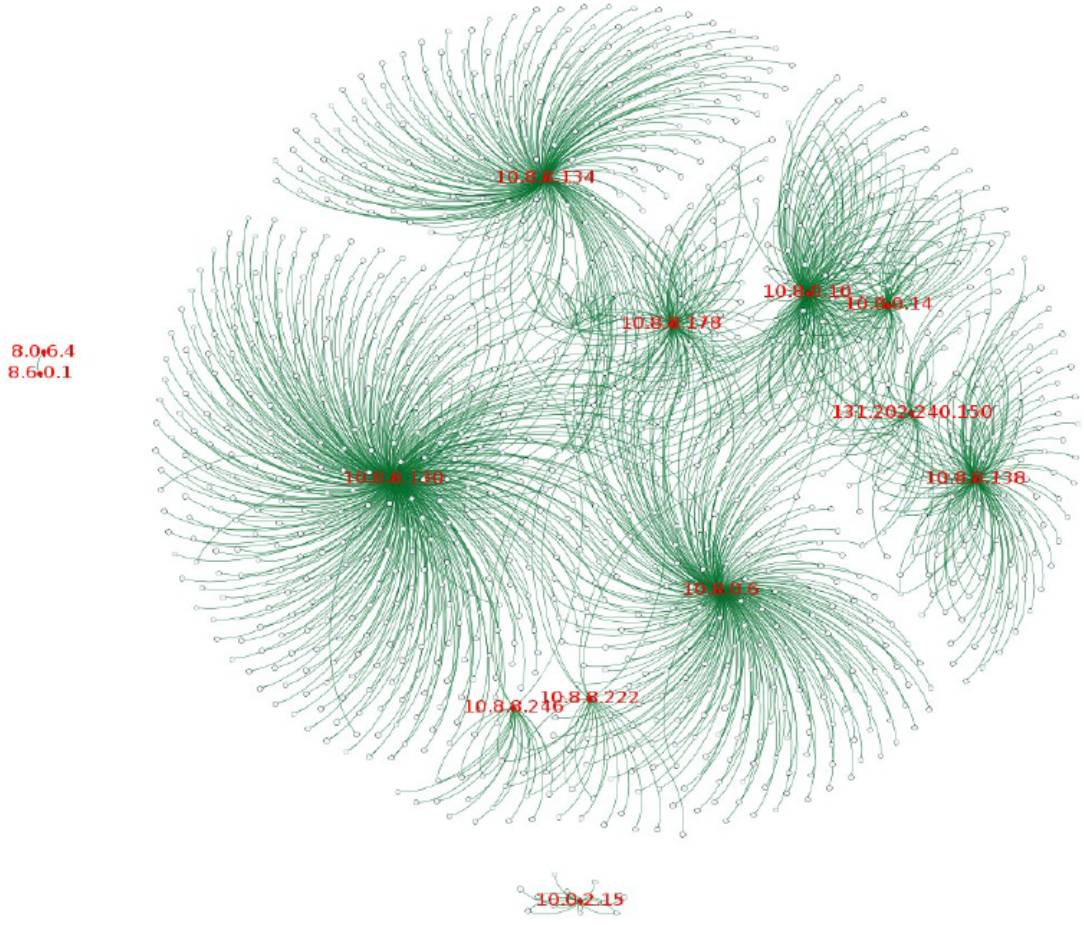
4.4 Konvoluutioneuroverkot

Konvoluutioneuroverkko on syväoppimisalgoritmi, joka ottaa syötteeksi harmaan tai värillisen kuvan, määrittää painot ja painottaa kuvan objekteja, suorittaa useita konvoluutioita pienentääkseen kuvan kokoa säilyttäen samalla tärkeät ominaisuudet ja luokittelee syötekuvan. CNN:n perustana on kolmen tyyppisiä kerroksia: konvoluutio-, poolaus- ja tiheyskerros (engl. dense layer). Konvoluutiokerros on lineaarinen operaatio, joka käyttää syötekuvaa ja valittua painomatriisia, jota kutsutaan suodattimeksi tai kerneliksi konvoloidun kuvan tuottamiseksi. Kerneli kulkee toistuvasti syöttökuvan yli laskeakseen valittujen elementtien tulon summan ja sijoittaa tuloksen konvoloituun matriisiin. Useita konvoluutioita käytetään poimiakseen matalan tason piirteitä ensimmäisen konvoluutiokerroksen jälkeen ja vastaavasti

korkean tason ominaisuuksia viimeisen konvoluutiokerroksen jälkeen. Matemaattisesti konvoluutio määritellään seuraavasti:

$$f[n,n,n_c]_s * g[f,f,n_c] = h[c,c,n_c],$$

missä f on nxn-syötekuva ja g on valittu fxf-suodatin askeleella s, joka tuottaa cxc-konvoloidun matriisin h. Syöttökuvalla, suodatinmatriisilla ja konvoloituneella kuvalla on n_c-kanavia, joita on kolme RGB:tä varten (värikuva) ja yksi harmaasävykuvalle. (Habibi Lashkari, Kaur ja Rahali 2020, 6)



Kuvio 11. Viestintä ainutlaatuisten lähde- ja kohdeparien välillä ja kohde-IP -osoitteet (Habibi Lashkari, Kaur ja Rahali 2020, 6)

Samoin kuin konvoluutiokerros, poolauskerros vähentää piirrevektorin (engl. feature vector) ulottuvuuden datan käsittelyyn tarvittavan laskennallisen tehon alentamiseksi. On olemassa kahdentyyppisiä poolauksia: max pooling ja average pooling. Max pooling ottaa maksimielementin syöttökuvan valitusta osasta ennalta määritetyllä suodatinkoolla ja askelarvolla, kun taas average pooling menee muodostamaan poolin valittujen elementtien keskiarvolla. Suodattimen kokoa ja askelarvoa (engl. stride) kutsutaan poolauksen hyperparametreiksi. Poolaaminen auttaa myös hallitsemaan ylisovitusta (engl. overfitting) mallia kouluttaessa. Matemaattisesti yhdistäminen esitetään seuraavasti:

$$n = ((n - F)/S) + 1,$$

missä n on kuvan koko, F on suodattimenkoko ja S askelarvo (engl. stride value). (Habibi Lashkari, Kaur ja Rahali 2020, 6)

Tiheyskerrosta kutsutaan myös täysin yhdistetyksi kerrokseksi (engl. fully-connected layer), jossa jokainen nykyisen tiheyskerroksen solmu on kytketty jokaiseen toiseen solmuun edellisessä kerroksessa. Tiheyskerros esittää epälineaarisen operaation, joka muuntaa syötekuvan monitasoiseksi perceptroniksi (engl. multi-level perceptron). Tiheyskerroksen tavoitteena on virittää painoparametrit kunkin luokan stokastisen todennäköisyysesityksen luomiseksi. (Habibi Lashkari, Kaur ja Rahali 2020, 6)

Seuraavat funktiot ovat käytössä konvoluutioneuroverkossa:

1. **Activation**: Se päättää, laukaiseeko neuroni vai ei. On olemassa erityyppisiä aktivointifunktioita, mutta Rectified Linear Unit (ReLU) tuottaa parhaita tuloksia.
2. **Dropout**: Se on säännöstelytekniikka, joka toimii pudottamalla tietyn prosenttiosuuden datasta mallin treenaamisen aikana vähentääkseen ylisovitussongelmaa.
3. **Flatten**: Piirrekartta litistetään sarakkeeksi syöttääksen nämä tiedot keinotekoiselle neuroverkolle (ANN) myöhemmin.
4. **One Hot Encoder**: Se on prosessi, jolla kategorisia muuttujia muunnetaan muotoon, joka voidaan syöttää oppimisalgoritmeille, jotta niiden ennustamistyö parantuu.

4.5 Ehdotettu malli

Habibi Lashkarin ym. (2020) ehdottama malli, DeepImage, perustuu kahteen pääkomponenttiin: piirteiden poimintaan (engl. feature extraction) valitakseen parhaat ominaisuudet ja mallin kerrostettuun näkymään (engl. layered view). Piirteiden poiminta on keskeisessä asemassa parhaan ominaisuusjoukon (engl. feature set) valinnassa darknet-liikenteen tunnistamiseksi ja karakterisoimiseksi. CICFlowMeteriä käytetään datan esikäsittelyvaiheessa 80 verkkoliikenneominaisuuden poimimiseen datasetistä ja kohdetunnisteet (engl. target labels) on määritelty kaikelle toimintaliikenteelle, mikä on tallennettu datasettiin piirrevektorin luomiseksi. 80 poimitusta ominaisuudesta, 61 ominaisuutta on valittu käyttämällä feature ranking -tekniikkaa, kuten kuviossa 12 näkyy. (Habibi Lashkari, Kaur ja Rahali 2020, 7)

Algorithm 1 Feature Selection

```
1: procedure COMPUTE_FEATURE_IMPORTANCE(Feature_Vector, Target_Labels)
2:     Build a forest of trees with input (Feature_Vector,Target_Labels)
3:     for each node in forest do
4:         Compute standarddeviation of node as array elements
5:         Sort the node in descending order to get indices and importance values for most important features
6:         if importance > 0.001 then
7:             Rank indices and importance values of best features
8:         end if
9:     end for
10: end procedure
```

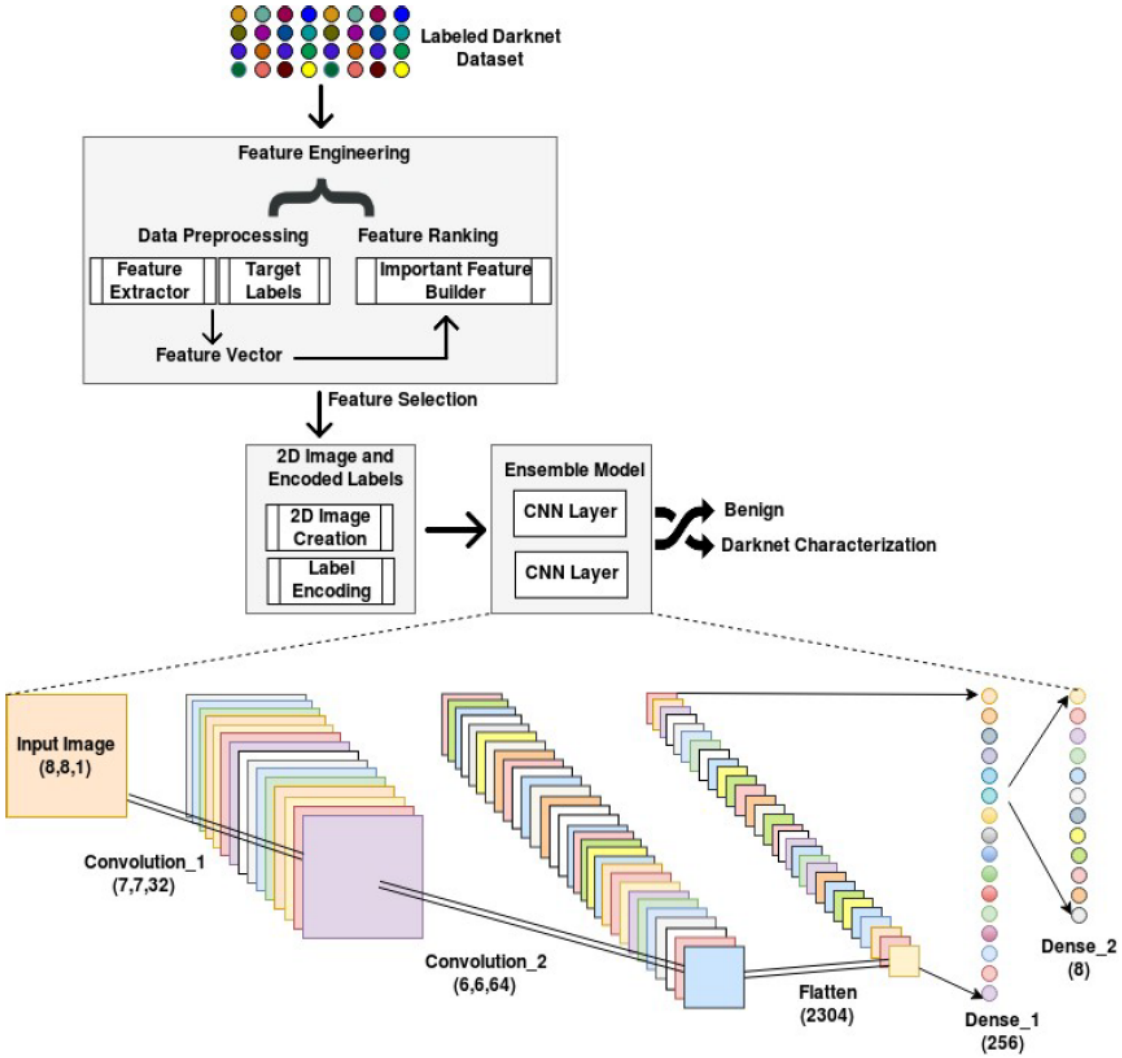
Kuvio 12. Ominaisuuksien valinta (Habibi Lashkari, Kaur ja Rahali 2020, 7)

Seuraavassa vaiheessa valittuja ominaisuuksia käytetään kaksiulotteisen kuvavektorin muodostamiseen ja koodataan numeerisesti kohdetunnisteet ennen kokonaisuusmalliin (engl. ensemble model) siirtymistä. Ehdotettu konvoluutioneuroverkkomalli muodostaa useita kerroksia datasetin luokittelemiseksi hyvänlaatuiseksi tai anonymisoiduksi ensimmäisessä kerroksessa sekä luonnehtii darknet-liikenteen kahdeksaan eri kategoriaan, kuten kuviossa 13 näkyy.

Kaksiulotteinen harmaa kuva, joka on luotu piirteiden poimimisen tuloksena, otetaan syötteeksi muodostamaan (8,8,1) kuvavektori, missä 8x8-vektoria käytetään tallentamaan 61

luettelossa olevaa ominaisuutta ja kolmas parametri ’1’ esittää harmaasävykuvaa ehdotetussa mallissa. Se näytetään flash-näkymänä kokonaisuusmallilaatikossa kuviossa 13. Ehdotetun konvoluutioneuroverkon syötekerros on peräkkäinen ja sitä seuraa ensimmäinen kaksiulotteinen (2D) konvoluutiokerros muodolla (7,7,32). Se käyttää 32 3x3-suodatinta 3x3-pikselin alialueiden poimimiseen aktivointifunktiolla ReLu. Toinen konvoluutio 64 3x3-suodattimella (6,6,64) sovelletaan malliin. Tätä seuraa tasoitus ja kaksi tiheää kerrosta. Tasoituskerros (Flatten kuvassa) muotoilee tensorin uudelleen kahdeksaan neuroniin, jotka edustavat kahdeksaa darknet-liikenteen kategoriaa datasetissä.

Poolauskerrokset on eliminoitu ehdotetun mallin kerrosnäkymästä. Ensimmäisen poolauskerroksen lisääminen ensimmäisen konvoluutiokerroksen jälkeen pienentää piirrevektorin arvoon (3,3,64). Siksi on vähemmän mahdollisuuksia lisätä enemmän konvoluutioita, sillä tuloksena on jyrkkä pudotus mallin kouluttamisessa ja sen tarkkuuden testaamisessa. Lisäksi, vain 80 ominaisuutta poimitaan alussa, jotka vähenee 61 ominaisuussijoituksen jälkeen, ja tarvitaan vain 8x8 syötekoko näiden ominaisuuksien sijoittamiseksi.



Kuvio 13. Metodologian arkkitehtuuri (Habibi Lashkari, Kaur ja Rahali 2020, 7)

4.6 Testausympäristö ja -parametrit

Ehdotettu järjestelmä on toteutettu Python-ohjelmointikielellä Kerasin ja TensorFlow:n avulla käyttämällä Scikit-Learn:ia. Kokeet suoritettiin Ubuntu-palvelimella, jossa oli 50 CPU:ta ja 500 Gt RAM-muistia. Lopuksi tietojoukko jaettiin mallille syötettävään koulutussarjaan (80 %) ja testaussarjaan (20 %). Kuvio 14 näyttää parametrit viritetyillä arvoilla maksimaalisen tarkkuuden ja pienimmän lokihäviön saavuttamiseksi.

Parameter	Value
Activation Function (Hidden layers)	RELU
Activation Function (Output layer)	Softmax
Loss Function	sparse_categorical_crossentropy
Optimizer	adam
Epoch	1500
Batch Size	32
Estimators	250
Maximum Depth	16
Early Stopping Monitor	patience = 3

Kuvio 14. Parametrit (Habibi Lashkari, Kaur ja Rahali 2020, 8)

4.7 Testaus, analyysi ja pohdinta

Ensinnäkin tutkijat tuovat esille parhaan ominaisuusjoukon darknet-liikenteen havaitsemiseen ja jatkaa darknet-liikenteen karakterisointia löytääkseen yhteisen toimtamallin darknet-liikenteessä.

4.7.1 Paras ominaisuussarja

Selvittääkseen CICFlowMeter:n kautta erotettujen verkkoominaisuuksien tärkeyden, tutkijat eliminoivat virtauksen omianisuusmerkinnät (engl. flow label features), mukaan lukien virtauksen id:n (flow ID), aikaleiman (timestamp), lähde- ja kohde-IP:n ja sitten laskivat tärkeysarvot kaikille ominaisuuksille (kuvio 12) osana piirteiden poimintaprosessia. Kuviossa 15 olevista suositeltujen ominaisuuksien tärkeysprosenttiluettelosta on selvää, että suurin tyhjäkäyntiarvo (engl. maximum idle value) on tärkein ominaisuus pimeänverkkoliikenteen havaitsemiseksi kerroksessa 1. Sitä seuraa eteenpäin minimisegmenttikoko (engl. minimum forward segment size) ja takaperin paketin vähimmäispituus (engl. minimum backward packet length). Kerroksessa 2 eteenpäin lähetettävät paketit per sekunti ovat tärkein ominaisuus darknet-liikenteen karakterisoinnissa. Sitä seuraavat taaksepäin tulevat paketit per sekunti ja maksimi tyhjäkäyntiarvo. Ominaisuuden tärkeysarvojen vertailu molemmissa tasoissa paljastaa seuraavat yhtäläisyydet:

- Kaikki suositellut ominaisuudet edistävät lähes yhtä paljon darknet-liikenteen detektorin koulutusta.
- 22 luetteloon valitusta ominaisuudesta 15 löytyy molemmista kerroksista, mikä osoittaa, että nämä ominaisuudet ovat välttämättömiä darknet-liikenteen tunnistamisessa hyvänlaatuisen liikenteen seasta kerroksessa 1 ja anonymisoidun darknet-liikenteen karakterisoinnissa kerroksessa 2.

Layer1

Rank	Index	Feature Name	Percent
1	F74	Idle Max	0.078017
2	F67	Fwd Seg Size Min	0.075886
3	F12	Bwd Pkt Len Min	0.072589
4	F1	Protocol	0.051608
5	F72	Idle Mean	0.048613
6	F64	Fwd Init Win Bytes	0.042459
7	F42	FIN Flag Count	0.042023
8	F63	Subflow Bwd Bytes	0.039559
9	F40	Packet Length Std	0.036495
10	F11	Bwd Pkt Len Max	0.035476
11	F13	Bwd Pkt Len Mean	0.035112
12	F75	Idle Min	0.034859
13	F53	Bwd Seg Size Avg	0.033452
14	F60	Subflow Fwd Packets	0.032802
15	F65	Bwd Init Win Bytes	0.030974
16	F51	Average Packet Size	0.029259
17	F38	Packet Length Max	0.020524
18	F0	Destination Port	0.018926
19	F39	Packet Length Mean	0.018509
20	F6	Total Len of Bwd Pkt	0.016423
21	F33	Fwd Header Length	0.014669
22	F36	Bwd Packets/s	0.014571

Layer2

Rank	Index	Feature Name	Percent
1	F35	Fwd Packets/s	0.075397
2	F36	Bwd Packets/s	0.062840
3	F74	Idle Max	0.04995
4	F2	Flow Duration	0.04119
5	F15	Flow IAT Mean	0.03928
6	F18	Flow IAT Min	0.03891
7	F17	Flow IAT Max	0.03522
8	F72	Idle Mean	0.03198
9	F75	Idle Min	0.02882
10	F12	Bwd Pkt Len Min	0.02755
11	F63	Subflow Bwd Bytes	0.026458
12	F13	Bwd Pkt Len Mean	0.026138
13	F0	Destination Port	0.025542
14	F40	Packet Length Std	0.024493
15	F53	Bwd Seg Size Avg	0.023945
16	F39	Packet Length Mean	0.023916
17	F51	Average Packet Size	0.023769
18	F11	Bwd Pkt Length Max	0.020243
19	F38	Packet Length Max	0.020210
20	F41	Pkt Len Variance	0.018736
21	F60	Subflow Fwd Packets	0.016200
22	F52	Fwd Seg Size Avg	0.015822

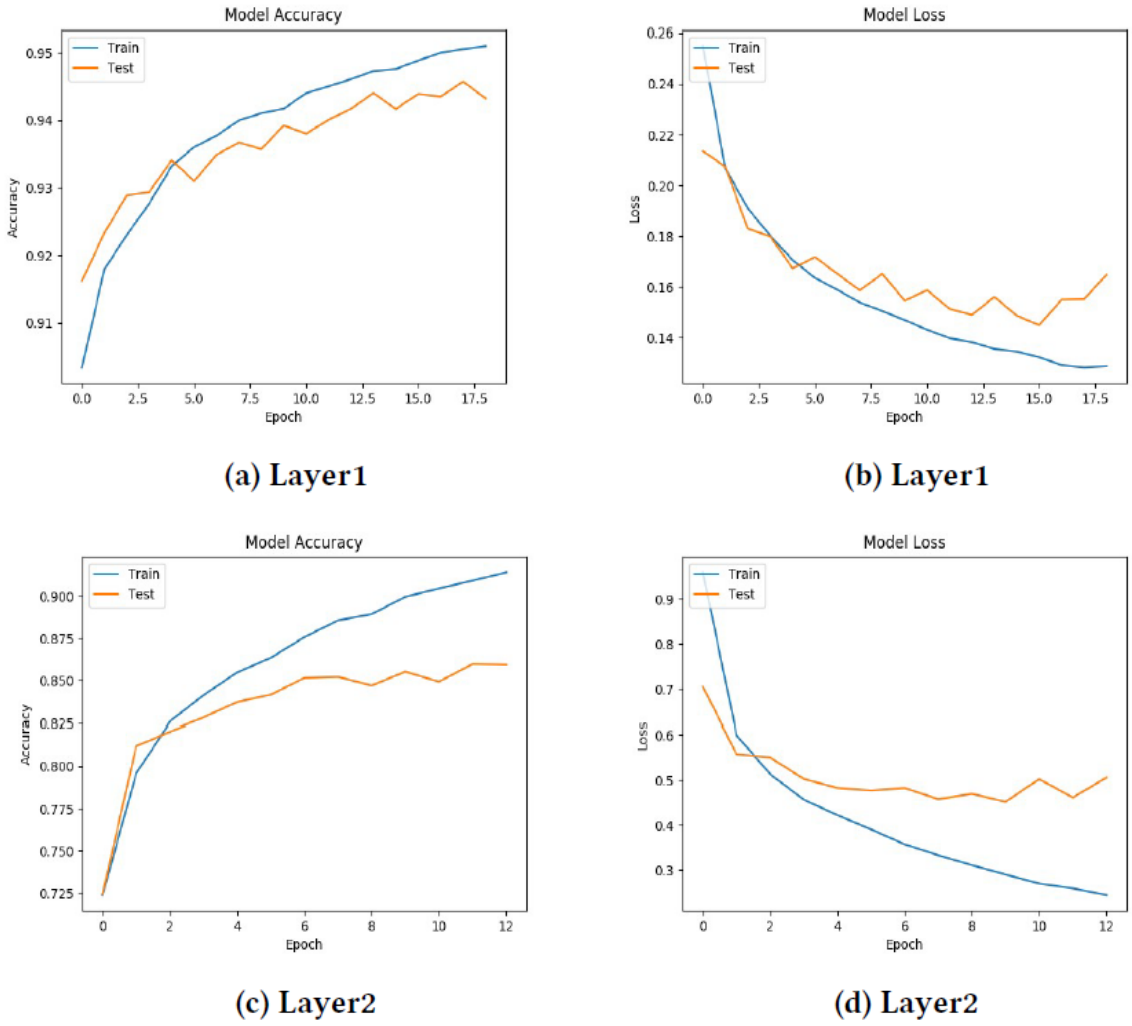
Kuvio 15. Parhaat ominaisuudet, jotka on valittu kaikista poimituista ominaisuuksista (Habibi Lashkari, Kaur ja Rahali 2020, 9)

4.7.2 DeepImage:n tarkkuus ja lokihäviö

Edellisessä vaiheessa valittuja parhaita ominaisuuksia käytetään kaksiulotteisen harmaakuvan luomiseen, joka syötetään DeepImage:lle suorittamista varten. DeepImage:n suorituskyvyn seuraamista varten tutkijat tekivät tarkkuus ja logaritmisen häviön koulutus- ja testauskäyrien eri epookkiarvoille kerrokselle 1 ja kerrokselle 2, kuten kuvassa 16 näkyy. Seuraavassa on tärkeimmät havainnot, jotka on johdettu tarkkuus- ja häviökäyristä, jotka kuvaavat DeepImage:n potentiaalia havaita ja karakterisoida darknet-liikennettä:

- Tarkkuuskäyrät molemmissa kerroksissa osoittavat nousevaa trendiä, mikä edellyttää, että epookkien arvojen kasvaessa koulutus- ja testaussettien tarkkuus myös kasvaa. Ylisovituksesta ei ole merkkiä kummassakaan käyrässä.

- Häviökäyrät molemmissa kerroksissa osoittavat laskevaa trendiä, mikä osoittaa, että epookkien arvojen kasvaessa koulutus- ja testaussettien lokihäviön määrä vähenee, mikä on toivottavaa. Häviökäyristä, jotka ennustivat todennäköisyyttä, voidaan tulkita, että se ei poikkea datasetin todellisesta luokittelusta.
- Kerroksessa 1 mallin tarkkuus luokittelijan kouluttamiseen on 95% ja testaus on 94%. Harjoittelu- ja testauskäyrien lokihäviö laskee arvoihin 0,13 ja 0,17.
- Kerroksessa 2 mallin tarkkuus luokittelijan kouluttamiseen on 92% ja testaus on 86%. Lokihäviö sekä harjoittelu- että testauskäyrille laskee arvoihin 0,2 ja 0,5.



Kuvio 16. DeepImage: tarkkuus ja häviö (Habibi Lashkari, Kaur ja Rahali 2020, 9)

4.7.3 Kilpaileva DL-algoritmi

Deep Packet, kilpailukykyinen syväoppimisen lähestymistapa, joka käytti ISCXVPN2016-datasettiä VPN-liikenteen luokittelemiseksi 1D CNN:n ja SAE-algoritmin avulla, ylitti kaikki koneoppimisluokittelut, 2D CNN ja SAE-algoritmit. Se saavutti 93%:n tarkkuuden VPN-liikenteen ja 35%:n tarkkuuden Tor-liikenteen karakterisoinnissa. Tutkijat arvioivat kuitenkin myös 1D CNN:n käyttämällä uutta darknet-datasettiä, joka sisältää sekä salatun VPN- että Tor-liikenteen. On havaittu, että 1D CNN:n tarkkuus osoittautui alun perin 63 prosentiksi ja se parani 73 prosenttiin piirteiden poiminnan ja hyperparametrien virityksen jälkeen. Tulokset kohteelle 1D CNN esitetään kuvassa 17 vertaamalla niitä DeepImage-malliin.

Category	Precision	Recall	F1-Score	Accuracy
1D CNN	0.74	0.73	0.73	0.73
DeepImage	**0.86**	**0.86**	**0.86**	**0.86**

Kuvio 17. Vertailu muihin DL-luokittelijoihin (Habibi Lashkari, Kaur ja Rahali 2020, 9)

4.7.4 Darknet liikenteen karakterisointi

Tutkijat suorittivat moniluokkaisen luokituksen ja arvioimme DeepImage:n laskemalla tarkkuus (engl. precision), palautus (engl. recall), f1-pisteet (engl. f1-score) ja tarkkuuden (engl. accuracy) kerrokselle 2 (layer2). Kuviosta 18 on havaittavissa, että 2635 äänen suoratoistonäytteestä 2423 (92 %) tunnistetaan oikein. Vastaavasti 38 40:stä P2P-näytteestä havaitaan P2P:ksi, jolloin sen palautusarvo on 0,98. Päinvastoin, selaamisen (engl. browsing) tarkkuus on 47%, mikä on alhaisin kaikista kategorioista. DeepImage karakterisoi kaikki sille annetut darknet-näytteet 86 %:n kokonaistarkkuudella useimmissa kategorioissa.

Vertailimme karakterisointituloksia 1D CNN:ään todistaaksemme DeepImagen suorituskyvyn tehokkuuden (kuvio 17). DeepImage suoriutui paljon paremmin kuin 1D CNN, mikä tekee selväksi, että 2D CNN:n suorituskykyä darknet-datasetissä ei voida korreloida salatun liikenteen karakterisoinnin tuloksiin, jotka on saatu aikaisemmissa tutkimuksissa, kuten Deep Packet -tutkimuksessa, joka käyttää vain salattua liikennettä VPN:n muodossa ja Non-VPN-muodossa tai Tor- ja Non-Tor-liikenteen muodossa.

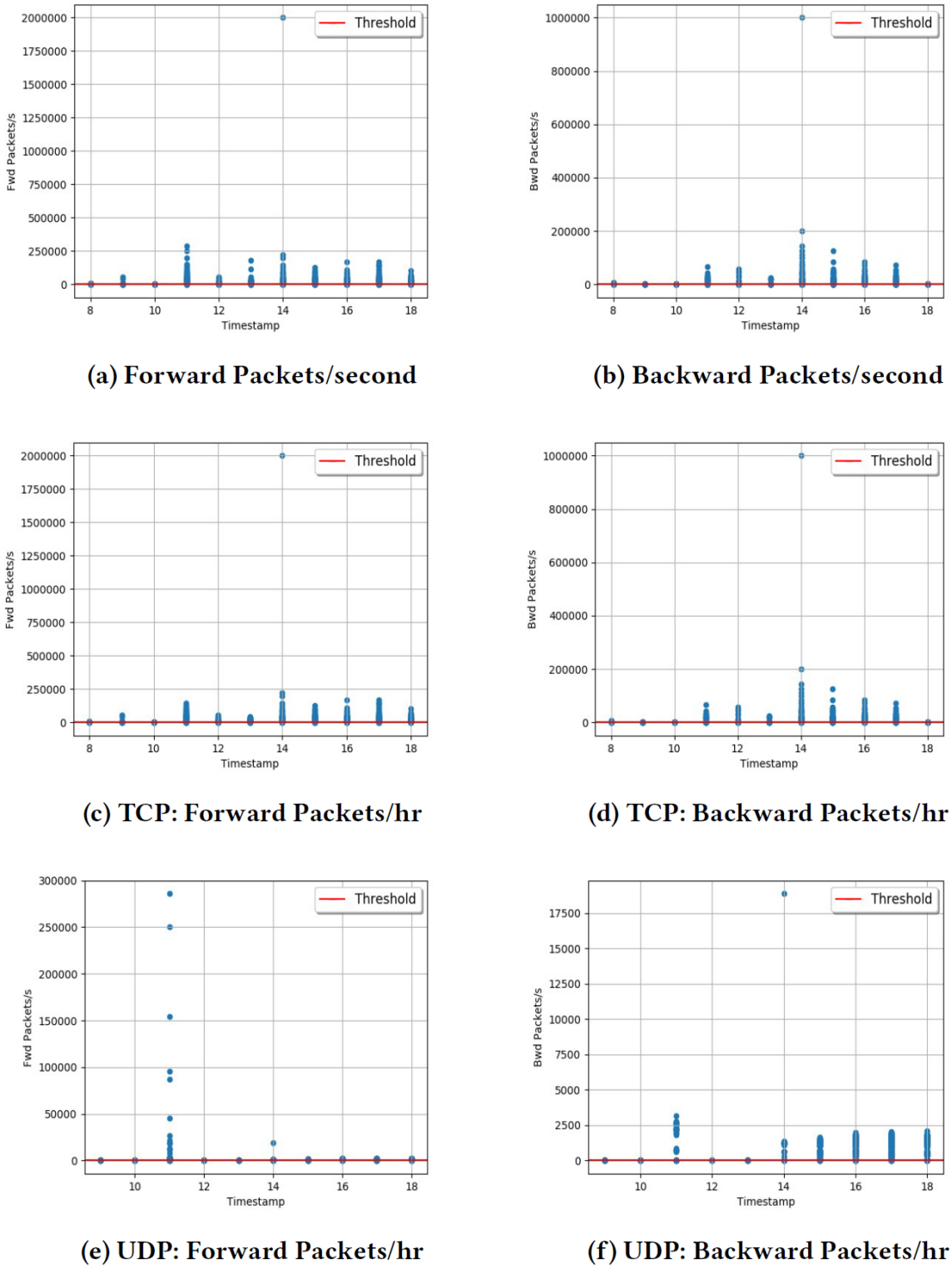
Category	Precision	Recall	F1-Score	Accuracy	FN Rate	#Testing Instances	#Training Instances
Audio-Streaming	0.92	0.92	0.92	0.92	0.8	2635	10649
Browsing	0.55	0.47	0.51	0.47	0.53	59	204
Chat	0.90	0.86	0.88	0.86	0.14	919	3622
Email	0.66	0.67	0.67	0.67	0.33	124	458
File-Transfer	0.74	0.75	0.75	0.75	0.25	521	2089
P2P	0.90	0.95	0.93	0.95	0.05	40	180
Video-Streaming	0.82	0.88	0.85	0.88	0.12	283	1063
VOIP	0.58	0.61	0.59	0.61	0.39	282	1183

Kuvio 18. Karakterisointi (Habibi Lashkari, Kaur ja Rahali 2020, 10)

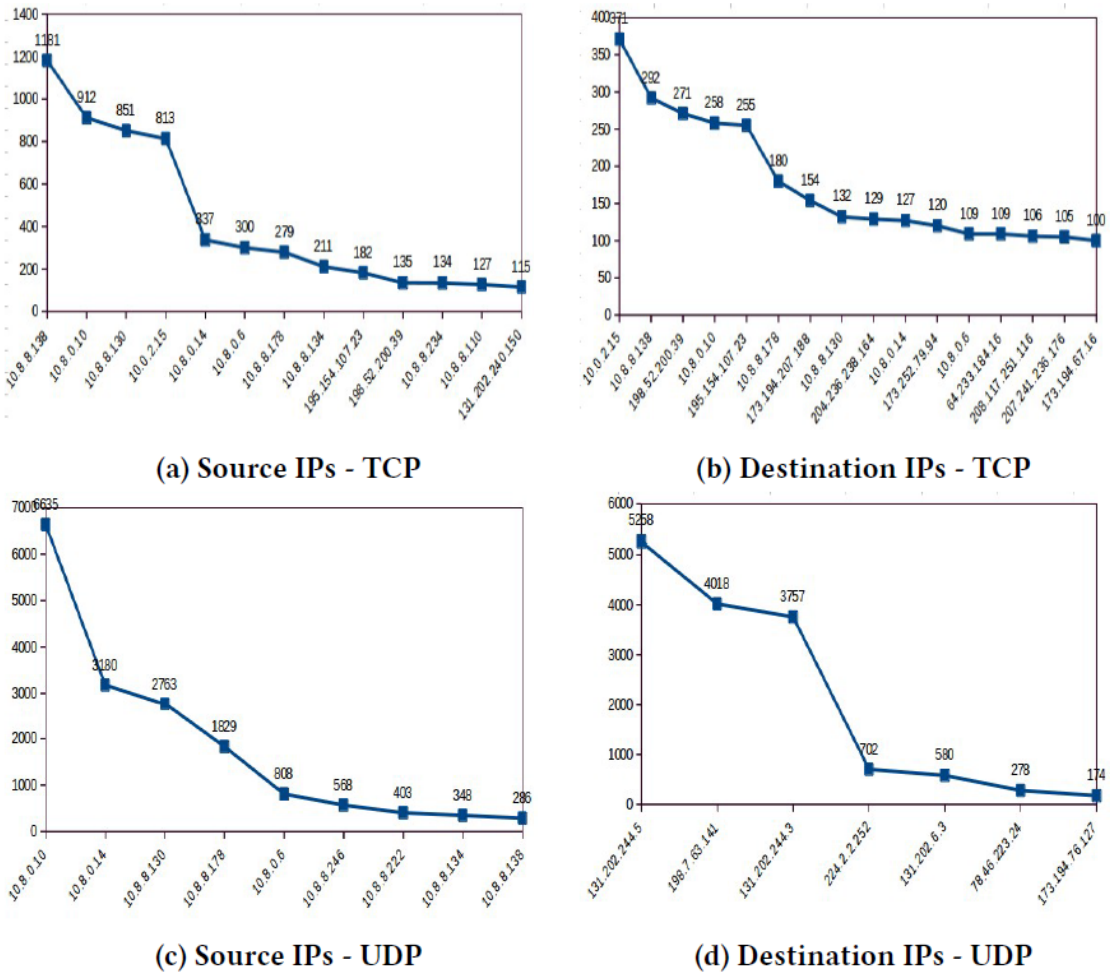
4.7.5 Darknet-liikenteen analyysi

Kun darknet-liikenne on havaittu ja karakterisoitu onnistuneesti edellisissä kohdissa, on syytä analysoida syvällisesti darknet-liikennettä protokollakohtaisen viestinnän trendin löytämiseksi. Kuten kuviossa 15 on lueteltu, eteenpäin lähtevät paketit per sekunti ja taaksepäin lähtevät paketit per sekunti ovat kolmen tärkeimmän ominaisuuden joukossa, jotka karakterisoivat darknet-liikennettä darknet-datasetissä. Siksi tutkijat piirsivät koko tunnin liikenteen ja TCP/UDP-liikenteen ajan suhteen kuvassa 19. Analysoimalla edelleenlähetyksen ja taaksepäin tulevien pakettien määrää per sekunti trendiä paljastuu, että suurimman osan ajasta darknet vastaanottaa alle 250 000 eteenpäin suunnattua pakettia ja alle 200 000 taaksepäin suunnattua pakettia. Eteenpäin suuntautuvien pakettien enimmäismäärä sekunnissa saavuttaa 2 000 000, kun taas taaksepäin tulevien pakettien enimmäismäärä sekunnissa on 1 000 000. Lisäksi TCP-pohjaiset eteenpäin- ja taaksepäin-paketit sekunnissa kantavat täsmälleen saman trendin kuin yleiset tuntikohtaiset eteenpäin- ja taaksepäin-paketit.

Siitä huolimatta UDP-pohjaiset eteenpäin suuntaavat paketit sekunnissa noudattavat ainutlaatuista trendiä, jossa erikokoiset paketit lähetetään klo 11.00 AST. Kaiken kaikkiaan, datasetti sisältää pääasiassa enemmän TCP-liikennettä kuin UDP-liikennettä. Lisäksi suoritettiin lähde- ja kohde-IP-osoitepohjaisen TCP- ja UDP-liikenneanalyysin analysoidakseen tärkeimmät yksityiset ja julkiset IP-osoitteet, joita käytetään selkeästi viestinnässä, kuten kuvassa 20 näytetään. On havaittavissa, että suurin osa TCP- ja UDP-viestinnässä käytetyistä lähde-IP-osoitteista on yksityisiä, kun taas suurin osa molempien protokollien näkyvistä IP-osoitteista on julkisia.



Kuvio 19. TCP- ja UDP-toimintaliikenteen analyysi (Habibi Lashkari, Kaur ja Rahali 2020, 10)



Kuvio 20. Erillisen IP-pohjaisen TCP/UDP-liikenteen analyysi (Habibi Lashkari, Kaur ja Rahali 2020, 11)

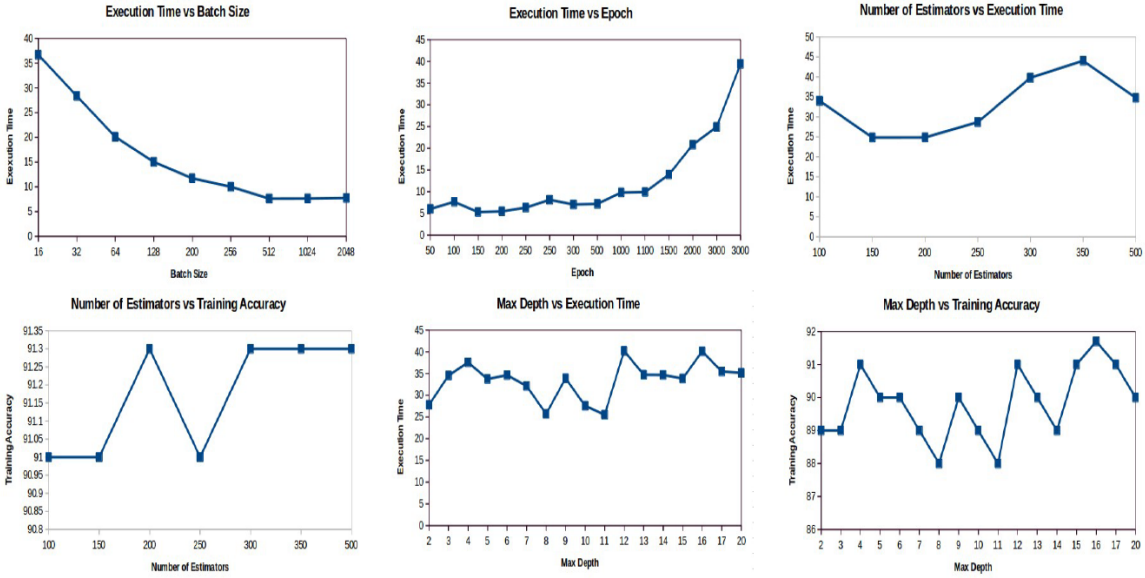
4.7.6 Hyperparametrien viritys

Hyperparametrit on viritetty parantamaan mallin suorituskykyä. Tutkijat testasivat seuraavat hyperparametrit:

- Toteutusaika: Kuvio 21 (a ja b) osoittaa selvästi, että suoritusaika lyhenee eksponentiaalisesti erän koon (engl. batch size) ja stabiilien (engl. stables) kasvaessa lopussa. Päinvastoin se pysyy lähes samana epookkiarvon kasvaessa 1100 asti, mutta kasvaa eksponentiaalisesti myöhemmin.
- Estimaattorien määrä Extra Trees -luokittimessa: Kuvio 21 (c ja d) osoittaa, että suo-

ritusaika lyhenee aluksi estimaattien määrän kasvaessa ja muuttuu sitten kiinteäksi ja myöhemmin kasvaa nopeasti ja lopulta vähenee. Päinvastoin harjoittelun tarkkuus pysyy tasaisena noin 91 % koko ajan.

- Extra Trees -luokittelijan maksimisyvyys: Kuvio 21 (e ja f) viittaa siihen, että toteutuksessa on vaihtelua metsän puiden enimmäissyvyyden kasvaessa. Suoritusajalla ei havaita tasaista mallia. Harjoittelutarkkuus vaihtelee kuitenkin 88 % ja 91,7 % välillä maksimisyvyysarvojen kasvaessa.



Kuvio 21. Hyperparametrivirityksen vaikutus (Habibi Lashkari, Kaur ja Rahali 2020, 11)

5 Datasetti ja testausympäristö

Tämä luku käsittelee Darknet 2020 -datasetin luontia ja sisältöä sekä lokaalin testausympäristön parametreja. Darknet 2020 -datasetti toimii kolmen koneoppimismallin datalähteenä ja se on alun perin kehitetty Habibi Lashkarin ym. (2020) tutkimusta varten. Merkittävin ero muihin olemassa oleviin darknet-tietoliikennedatasetteihin, joita käsitellään enemmän luvussa 4.2, on se, että Darknet 2020 sisältää sekä Tor- että VPN-liikennettä, mikä tekee siitä paljon luotettavamman muihin datasetteihin verrattuna.

5.1 Darknet 2020 -datasetti

Kuten on jo aiemmassa luvussa mainittu, tutkijat Habibi Lashkari ym. (2020) muodostivat DeepImage:n darknet-datasetin kahdesta jo olemassa olevasta datasetistä: ISCXVPN2016 ja ISCXTor2016. Datasetti ISCXVPN2016 on tuotettu luomalla kahdelle käyttäjälle, Alicelle ja Bobille, omat tilit, jotta ne voisivat käyttää erilaisia palveluita, kuten Skypeä tai Facebookkia. Tutkijat kaappasivat sekä tavallisen istunnon (engl. session) että VPN:n kautta toteutetun istunnon, joten heillä on yhteensä 14 liikenneluokkaa: VOIP, VPN-VOIP, P2P, VPN-P2P jne. Liikenne kaapattiin Wiresharkin ja tcpdumpin avulla, mikä tuotti yhteensä 28GB dataa. VPN:lle käytettiin ulkoista VPN-palveluntarjoajaa ja yhdistettiin siihen OpenVPN:llä (UDP mode). SFTP- ja FTPS-liikenteen tuottamiseen käytettiin myös ulkopuolista palveluntarjoajaa ja Filezillaa asiakkaana. (University of New Brunswick: Canadian Institute for Cybersecurity 2016b)

Datasetti ISCXTor2016 tuotettiin luomalla kolme käyttäjää selainliikenteen keräämiseen ja kaksi käyttäjää kommunikaatiota varten, kuten chat, sähköposti, FTP jne. Non-Tor-liikenteeseen käytettiin aiempaa hyvänlaatuista liikennettä VPN-projektista. Liikenne kaapattiin Wiresharkin ja tcpdumpin avulla, mikä tuotti yhteensä 22GB dataa. Merkintäprosessin (engl. labeling process) helpottamiseksi tallennettiin lähtevän liikenteen työasemalta ja gateway:sta samanaikaisesti ja kerättiin joukko .pcap-tiedostopareja: yksi tavallisen liikenteen pcap (workstation) -tiedosto ja yksi Tor-liikenteen pcap (gateway) -tiedosto. Myöhemmin tunnistettiin kaapattu liikenne kahdessa vaiheessa. Ensin käsiteltiin työasemalla kaapatut .pcap-tiedostot:

poimittiin virrat (engl. flow) ja vahvistettiin, että suurin osa liikennevirroista oli sovelluksen X (Skype, ftps jne.) tuottamia, liikenteen kaappauksen kohteena. Tämän jälkeen merkattiin kaikki Tor .pcap -tiedoston virtaukset X:ksi. (University of New Brunswick: Canadian Institute for Cybersecurity 2016a)

5.2 Testausympäristö

Valitut algoritmit testattiin gradun tekijän omalla koneella, jonka tarkemmat tekniset tiedot näkyvät taulukossa 1.

Koneen tiedot	
Käyttöjärjestelmä	Windows 10
Prosessori	Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz 2.81 GHz
Näytönohjain	NVIDIA GeForce GTX 1050 Ti
RAM	16,0 GB
Järjestelmän tyyppi	64-bittinen käyttöjärjestelmä, x64-pohjainen prosessori

Taulukko 1. Yhteenveto koneen teknisistä tiedoista

5.3 Testausalgoritmit

5.3.1 Random Forest -algoritmi

Vuonna 1994 Berkeleyn professori Leo Breiman julkaisi teknisen raportin nimeltä *Bagging Predictors*, joka osoittautui yhdeksi nykyaikaisen koneoppimisen vaikutusvaltaisimmista ideoista. Breimanin ehdottama menettely sisälsi seuraavat askeleet:

1. Valitaan satunnaisesti tietojen rivien osajoukko (eli "oppimisjoukon käynnistyskopiot").
2. Koulutetaan malli käyttämällä tätä osajoukkoa.
3. Tallennetaan malli ja palataan sitten vaiheeseen 1 muutaman kerran.
4. Tämä antaa meille useita koulutettuja malleja. Tehdääkseen ennusteen, ennustetaan käyttämällä kaikkia malleja ja ottamalla sitten kunkin mallin ennusteen keskiarvo.

(Howard ja Gugger 2020, 298)

Tämä menettely tunnetaan nimellä *bagging* (lyhenne sanoista Bootstrap Aggregation) (Zhang ja Ma 2012, 12). Se perustuu syvään ja tärkeään näkemykseen: vaikka jokainen datan osajoukolle koulutettu malli tekee enemmän virheitä kuin koko tietojoukolle koulutettu malli, nämä virheet eivät korreloi keskenään. Eri mallit tekevät erilaisia virheitä ja näiden virheiden keskiarvo on siis nolla. Joten jos otamme kaikkien mallien ennusteiden keskiarvon, meidän pitäisi päätyä ennusteeseen, joka tulee lähemmäksi oikeaa vastausta, mitä enemmän malleja meillä on. Se tarkoittaa, että voimme parantaa lähes minkä tahansa koneoppimisalgoritmin tarkkuutta harjoittelemalla sitä useita kertoja, joka kerta eri satunnaisessa datan osajoukossa ja laskemalla sen ennusteiden keskiarvon. (Howard ja Gugger 2020, 298) Algoritmi on yksi varhaisimmista ja yksinkertaisimmista, mutta tehokkaimmista kokonaisuuspohjaisista (engl. ensemble-based) algoritmeista (Zhang ja Ma 2012, 12).

Random Forestit ovat Breimanin bagging-idean jatke, ja ne kehitettiin tehostamisen (engl. boosting) kilpailijaksi (Zhang ja Ma 2012, 157). Nykyään se on ehkä laajimmin käytetty ja käytännössä yksi tärkeimmistä koneoppimismenetelmistä. Pohjimmiltaan random forest on malli, joka laskee keskiarvon päätöspuiden ennusteista, jotka generoidaan satunnaisesti vaihtelemalla erilaisia parametreja, jotka määrittelevät, mitä dataa käytetään puun ja muiden puuparametrien kouluttamiseen. Bagging on erityinen lähestymistapa kokoonpanoon (engl. ensembling), useiden mallien tulosten yhdistämiseen. (Howard ja Gugger 2020, 298)

Random forest -algoritmia voidaan käyttää joko kategoriselle vastemuuttujalle, jota kutsutaan luokitukseksi (engl. classification), tai jatkuvalle vastaukselle, jota kutsutaan regressioksi (engl. regression). Vastaavasti ennustajamuuttujat voivat olla joko kategorisia tai jatkuvia. Laskennallisesta näkökulmasta random forestit ovat houkuttelevia, koska ne käsittelee luonnollisesti sekä regressiota että (moniluokka) luokittelua; ovat suhteellisen nopeasti koulutettavissa ja ennustettavissa; riippuvat vain yhdestä tai kahdesta viritysparametristä; niillä on sisäänrakennettu arvio yleistysvirheestä; voidaan käyttää suoraan korkean ulottuvuuden ongelmiin ja niitä voidaan helposti toteuttaa rinnakkain. (Zhang ja Ma 2012, 157)

Kuten nimestä voi päätellä, random forest on puupohjainen kokoonpano, jossa jokainen puu riippuu satunnaismuuttujien kokoelmasta. p-ulotteiselle satunnaisvektorille $X = (X_1, ..., X_p)^T$,

joka edustaa reaaliarvoisia tulo- tai ennustajamuuttujia ja satunnaismuuttuja Y, joka edustaa reaaliarvoista vastetta, oletamme tuntemattoman yhteisjakauman $P_{XY}(X,Y)$. Tavoitteena on löytää ennustefunktio $f(X)$ Y:n ennustamiseksi. Ennustefunktion määrittää häviöfunktio $L(Y, f(X))$ ja määritetty minimoimaan häviön odotettu arvo

$$E_{XY}(L(Y, f(X))),$$

missä alaindeksit tarkoittavat odotusta X:n ja Y:n yhteisjakauman suhteen. (Zhang ja Ma 2012, 158)

Intuitiivisesti $L(Y, f(X))$ on mitta siitä, kuinka lähellä $f(X)$ on Y:tä; se rankaisee niitä $f(X)$ arvoja, jotka ovat Y:stä kaukana. Tyypillisiä L:n valintoja ovat neliöity virhehäviö $L(Y, f(X)) = (Y - f(X))^2$ regressiolle ja 0-1 häviö luokittelulle:

$$L(Y, f(X)) = I((Y) \neq f(X)) = \begin{cases} 0 & \text{jos } Y = f(X) \\ 1 & \text{muuten} \end{cases}$$

Osoittautuu, että $E_{XY}(L(Y, f(X)))$ minimoiminen neliön virhehäviölle antaa ehdollisen odotuksen

$$f(x) = E(Y|X = x)$$

joka tunnetaan regressiofunktiona. Luokitustilanteessa, jos Y:n mahdollisten arvojen joukkoa merkitään $\mathscr{Y}$:llä, minimoidaan $E_{XY}(L(Y, f(X)))$ 0-1 häviöön antaa

$$f(x) = \arg\max_{y \varepsilon \mathscr{Y}} P(Y = y|X = x),$$

joka tunnetaan Bayesin sääntönä. (Zhang ja Ma 2012, 158)

Kokonaisuudet (engl. ensembles) rakentaa f:n niin sanottujen perusoppijoiden (engl. base learners) kokoelmana $h_1(x), ..., h_J(x)$ ja nämä perusoppijat yhdistetään antamaan kokonaisuusennustaja (engl. ensemble predictor) $f(x)$. Regressiossa perusoppijoista lasketaan keskiarvo

$$f(x) = \frac{1}{J} \sum_{j=1}^{J} h_j(x),$$

kun taas luokituksessa $f(x)$ on useimmin ennustettu luokka

$$f(x) = \arg\max_{y \varepsilon \mathscr{Y}} \sum_{j=1}^{J} I(y = h_j(x)).$$

Random foresteissa j:nnes perusoppija on puu, joka on merkitty $h_j(X, \Theta_j)$, missä Θ_j on kokoelma satunnaismuuttujia ja Θ_j:t ovat riippumattomia $j = 1, ..., J$. (Zhang ja Ma 2012, 159)

5.3.2 Gradient Boosting -algoritmi

On toinenkin tärkeä lähestymistapa kokoonpanoon (engl. ensembling), jota kutsutaan tehostukseksi (engl. boosting), jolloin lisätään mallien lukumäärää niiden keskiarvoistamisen sijaan (Howard ja Gugger 2020, 323). Yleisimmät ensemble-tekniikat, kuten random forestit, perustuvat yksinkertaiseen mallien keskiarvoon kokoonpanossa. Boosting-menetelmien perhe perustuu erilaiseen, rakentavaan kokonaisuuden muodostusstrategiaan. (Natekin ja Knoll 2013, 1) Näin tehostaminen toimii:

1. Treenataan pieni malli, joka ei sovi meidän datasettiin.
2. Lasketaan tämän mallin harjoitusjoukon ennusteet.
3. Vähennetään ennusteet tavoitteista; näitä kutsutaan "jäännöksiksi"ja ne edustavat virhettä harjoitusjoukon jokaiselle pisteelle.
4. Palataan vaiheeseen 1, mutta sen sijaan, että käytettäisiin alkuperäisiä tavoitteita, käytetään jäännöksiä harjoituksen tavoitteina.
5. Jatketaan tätä, kunnes saavutetaam jonkin pysäytyskriteerin, kuten puiden enimmäismäärän, tai huomataan vahvistusjoukon virheen pahenevan.

(Howard ja Gugger 2020, 323)

Tätä lähestymistapaa käyttämällä jokainen uusi puu yrittää sovittaa kaikkien aiempien puiden yhteiseen virheeseen. Koska luomme jatkuvasti uusia residuaaleja, vähentämällä kunkin uuden puun ennusteet edellisen puun jäännöksistä, jäännökset pienenevät ja pienenevät. Ennusteiden tekemiseksi tehostettujen puiden joukolla laskemme ennusteet jokaisesta puusta ja lisäämme ne sitten yhteen. On olemassa monia malleja, jotka noudattavat tätä peruslähestymistapaa, ja samoille malleille on monia nimiä. Gradienttitehostinkoneet (engl. gradient boosting machines, GBM) ja gradienttitehostetut päätöspuut (engl. gradient boosted decision trees, GBDTs) ovat termejä, joihin törmätään todennäköisimmin. Kirjoittamishetkellä XGBoost on suosituin. (Howard ja Gugger 2020, 323–324)

Jerome H. Friedmanin kehittämä gradient boosting -algoritmin perusideana on rakentaa uudet perusoppijat korreloimaan mahdollisimman paljon häviöfunktion negatiivisen gradientin kanssa, joka liittyy koko kokoonpanoon (Friedman 2001; Natekin ja Knoll 2013). Sekä häviöfunktio että perusoppijamallit voidaan määrittää mielivaltaisesti tarpeen mukaan. Kun otetaan huomioon jokin spesifinen häviöfunktio $\Psi(y, f)$ ja/tai mukautettu perusoppija $h(x, \theta)$, parametrien arvioiden ratkaisu voi olla vaikea saada. Tämän ratkaisemiseksi ehdotettiin, että funktio $h(x, \theta)_t$ on rinnakkaisin negatiivisen gradientin $\{g_t(x_i)\}_{i=1}^N$ kanssa havaittujen tietojen mukaan:

$$g_t(x) = E_y \left[\frac{\partial \Psi(y.f(x))}{\partial f(x)} | x \right]_{f(x)=\hat{f}^{t-1}(x)}.$$

Tietyn GBM:n suunnittelemiseksi tiettyä tehtävää varten on annettava funktionaalisten parametrien $\Psi(y, f)$ ja $h(x, \theta)$ valinnat. Toisin sanoen, on määriteltävä, mitä todella aiotaan optimoida, ja sen jälkeen valita funktion muoto, jota käytetään ratkaisun rakentamisessa. On selvää, että nämä valinnat vaikuttavat suuresti GBM-mallin ominaisuuksiin. (Natekin ja Knoll 2013, 4) Tälle luokalle ja kaikille gradienttitehostetuille puumenetelmille on monia säädettäviä hyperparametreja. Toisin kuin random forestit, gradienttitehostetut puut ovat erittäin herkkiä näiden hyperparametrien valinnalle. Käytännössä useimmat ihmiset käyttävät silmukkaa, joka yrittää useita erilaisia hyperparametreja löytääkseen ne, jotka toimivat parhaiten. (Howard ja Gugger 2020, 324)

5.3.3 Logistic regression -algoritmi

Logistisesta regressiosta on tullut yksi tilastotieteilijöiden ja tutkijoiden eniten käyttämistä tilastomenetelmistä binääri- ja suhteellisten vastetietojen analysointiin (Hilbe 2009, 1). Logistista regressiota pidetään tällä hetkellä parhaana käytäntönä käsiteltäessä tuloksia, jotka ovat kaksijakoisia tai kategorisia (Osborne 2015). Regressio on tilastollinen menetelmä, jolla yksi muuttuja selitetään tai ymmärretään yhden tai useamman muun muuttujan perusteella. Selitettävää muuttujaa kutsutaan riippuvaksi tai vastemuuttujaksi; muita muuttujia, joita käytetään selittämään tai ennustamaan vastausta, kutsutaan riippumattomiksi muuttujiksi. Lineaarinen regressio on standardi tai perusregressiomalli, jossa vasteen keskiarvo ennustetaan tai selitetään yhden ennustajan perusteella. Perusmalli on helposti laajennettavissa siten, että

siitä tulee monimuuttuja lineaarinen malli, eli lineaarinen regressio, jossa on useampi kuin yksi ennustaja. (Hilbe 2009, 1)

Logistisen regressiomallin erottaa lineaarisesta regressiomallista se, että logistisen regression tulosmuuttuja on binaarinen tai dikotomisinen. Tämä ero heijastuu sekä mallin muotoon että sen oletuksiin. (Hosmer Jr, Lemeshow ja Sturdivant 2013, 1) Tarkastellaan kokoelmaa p riippumattomia muuttujia, jotka on merkitty vektorilla $x' = (x_1, x_2, ..., x_p)$. Oletetaan, että jokainen näistä muuttujista on vähintään intervalliskaalattu. Merkitään ehdollinen todennäköisyys, että lopputulos on olemassa, $Pr(Y = 1|x) = \pi(x)$. Monilogistisen regressiomallin logit saadaan yhtälöstä

$$g(x) = ln\Big(\frac{\pi(x)}{1-\pi(x)}\Big) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p,$$

jossa usean logistisen regression mallille pätee

$$\pi(x) = \frac{e^{g(x)}}{1 + e^{g(x)}}.$$

(Hosmer Jr, Lemeshow ja Sturdivant 2013, 35–36)

Ensisijainen käsitteellinen ero lineaarisen tai moninkertaisen regression ja logistisen regression välillä on, että logistisessa regressiossa esittämiemme kysymystyyppeihin liittyy kaksijakoinen (tai kategorinen) riippuvainen muuttuja (engl. dependent variable, DV), kuten onko opiskelija saanut tutkinnon vai ei. Lineaarisessa tai moninkertaisessa regressiossa riippuvien muuttujien oletetaan olevan luonteeltaan jatkuvia. Logistinen regressio on luonteeltaan epälineaarinen; tapa, jolla logistinen regressio muuntaa dikotomisen tai kategorisen muuttujan riippuvaiseksi muuttujaksi, joka voidaan ennustaa muista binäärisistä, kategorisista tai jatkuvista muuttujista, sisältää epälineaarisen muunnoksen. (Osborne 2015)

6 Tulokset ja johtopäätökset

Tämä luku esittelee kunkin koneoppimisalgoritmin tulokset. Kullakin algoritmilla suoritettiin testi lokaalissa testausympäristössä käyttäen Python-ohjelmointikieltä sekä scikit-learn:in moduuleja. Kunkin algoritmin tapauksessa datasetti jaettiin aluksi harjoitus-settiin (80%) ja testisettiin (20%), joilla malllit siis koulutettiin ja sen jälkeen testattiin. Koko Darknet 2020 -datasetti ja kolmen mallin koodi on nähtävissä osoitteessa https://github.com/anarikai/Masters-thesis.

6.1 Random Forest -algoritmin tulokset

Testikerralla luotiin 900:n puun metsä asettamalla `n_estimators`-parametriksi 900, jossa jokaisella puulla on myös 10 kerrosta asettamalla `max_depth`-parametriksi 10. Jotta testit voidaan tarvittaessa ajaa uudelleen, on parempi tehdä tuloksesta toistettavan asettamalla `random_state`-parametri SEED:iksi.

Taulukossa 2, joka kuvaa random forest -algoritmin luokitteluraporttia, voidaan nähdä, miten algoritmi suoriutui kunkin luokan tietoliikenteen luokittelusta. Luokitteluraportti näyttää mallin tarkkuus- (precision), palautus- (recall), F1- (f1-score) ja tukipisteet (support). Tarkkuus määritellään todellisten positiivisten (engl. true positives, TP) ja mallin ennustaman positiivisten kokonaismäärän suhteeksi. Kunkin luokan tarkkuus voidaan sitten määritellä todellisten positiivisten ja väärien positiivisten (engl. false positives, FP) avulla seuraavasti: tarkkuus = TP/(TP+FP). (Sammut ja Webb 2011, 780) Recall tai herkkyys (engl. sensitivity), on mallin oikein ennustamien positiivisten esimerkkien murto-osa (Sammut ja Webb 2011, 901). Herkkyys voidaan määrittää seuraavasti: recall = TP/(TP+FN). F_1 voidaan määrittää seuraavasti: $F_1 = 2 \times precision \times recall / (precision + recall)$. (Sammut ja Webb 2011, 292) Tuki, support, tarkoittaa luokan todellisten esiintymien määrää tietojoukossa (Leung 2022). Esimerkiksi support määrä 18653 luokalle 0 tarkoittaa, että on 18653 havaintoa, jolla on todellinen non-Tor-merkintä (engl. label).

Accuracy, mikä voidaan myös suomentaa tarkkuutena, viittaa siihen, missä määrin mallin ennusteet vastaavat mallinnettavaa todellisuutta, ja termiä käytetäänkin usein luokittelumal-

lien yhteydessä. Jotta accuracy- ja precision-termit ei mene sekaisin, käytetään tässä osassa termien englanninkielistä muotoa. Tässä yhteydessä accuracy = $P(\lambda(X) = Y)$, missä XY on yhteisjakauma (engl. joint distribution), ja luokitusmalli λ on funktio $X \rightarrow Y$. Joskus tämä määrä ilmaistaan etusijalla eikä arvona välillä 0,0 - 1,0. Testidatan luokittelijan accuracy-arvo voidaan laskea oikein luokiteltujen kohteiden lukumäärä jaettuna objektien kokonaismäärällä. (Sammut ja Webb 2011, 9)

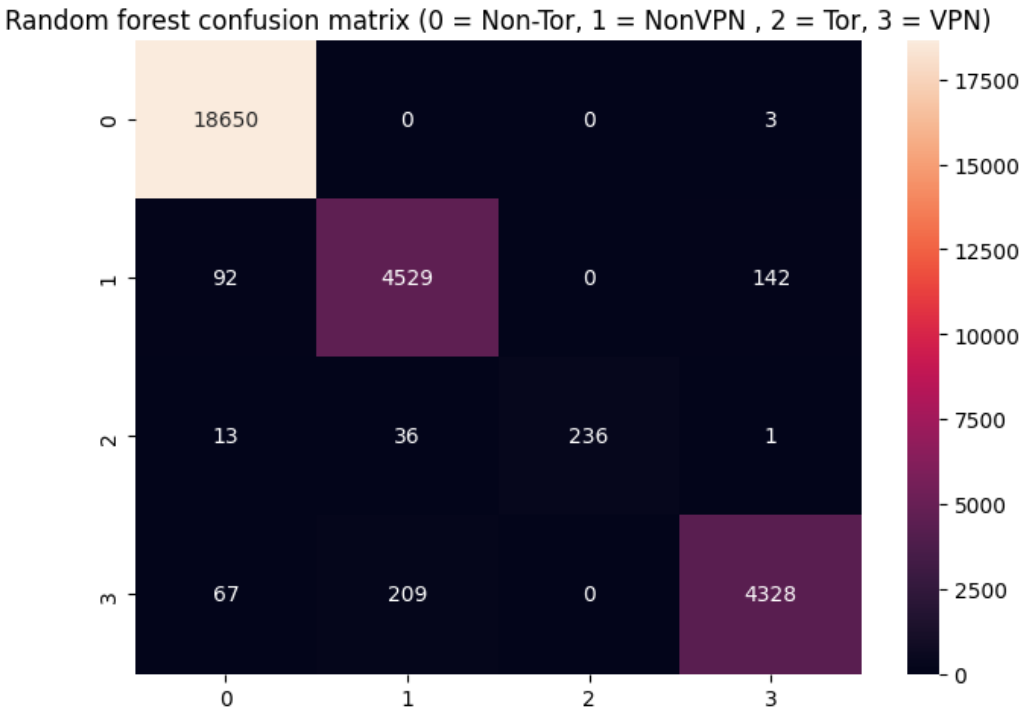
Makrokeskiarvoinen F1-pistemäärä (tai makro-F1-pistemäärä), macro avg luokitteluraportissa, lasketaan käyttämällä kaikkien luokkakohtaisten F1-pisteiden aritmeettista keskiarvoa (painottamatonta keskiarvoa). Tämä menetelmä kohtelee kaikkia luokkia tasavertaisesti niiden tukiarvoista riippumatta. F1-pisteiden painotettu keskiarvo, weighted avg, lasketaan ottamalla kaikkien luokkakohtaisten F1-pisteiden keskiarvo ottaen huomioon kunkin luokan support-määrä. (Leung 2022)

Mallin tulokset havainnollistetaan hämmennysmatriisina (engl. confusion matrix), joka esittää yhteenvedon luokittelijan luokittelusuorituskyvystä testidatan suhteen. Luokat (engl. class) 0-3 kuvaavat datasetin neljää tietoliikennetyyppiä: non-Tor, non-VPN, Tor ja VPN. Pystyakselilla on näkyvissä todellinen (engl. actual) luokka ja vaaka-akselilla ennustettu (engl. predicted) luokka. Diagonaaliset elementit ovat oikein luokiteltuja soluja ja diagonaalista poikkeavat solut edustavat elementtejä, jotka luokiteltiin väärin. Hämmennysmatriisi näyttää, missä tarkalleen virheet tapahtuvat (Howard ja Gugger 2020, 76).

Hämmennysmatriisista 22 voidaan nähdä, kuinka hyvin random forest -algoritmi pärjäsi darknet-liikenteen luokittelussa kunkin luokan suhteen. Parhaiten malli pystyi luokittelemaan non-Tor-liikenteen luokassa 0. Luokan kaikista 18653 alkioista 18650 luokiteltiin oikein. Myöskin luokitteluraportti 2 osoittaa, että luokassa 0 precision, recall ja F1-pisteet ovat muita luokkia korkeammalla tuloksin 0,99, 1,00 ja 1,00. Tämä puolestaan viittaa siihen, että malli on oppinut tunnistamaan luokan 0 esiintymät erittäin hyvin. Myös VPN-liikenteen luokassa 3 on korkeat precision-, recall- ja F1-arvot luvuin 0,97, 0,94 ja 0,95.

Luokalla 1 on hieman alempi F1-pistemäärä 0,95, mikä osoittaa, että mallilla on joitain vaikeuksia tunnistaa tämän luokan esiintymiä. Luokan 1 precision- ja recall-arvot ovat kuitenkin edelleen kohtuullisen korkeat. Luokalla 2 on alhaisin F1-pistemäärä, 0,90, mikä viittaa

siihen, että mallilla on myöskin hankaluuksia tämän luokan esiintymien tunnistamisessa. Luokan 2 precision-arvo on kuitenkin täydellinen, mikä tarkoittaa, että kun malli ennustaa esiintymän luokassa 2, se on melkein aina oikein. Alempi recall-arvo viittaa siihen, että mallista saattaa puuttua joitakin luokan 2 esiintymiä. Myöskin ottaen huomioon muiden luokkien support-määrät, luokalla 2 on kaikista vähiten esiintymiä testidatasetissä. Yhteenvetona voidaan todeta, että malli näyttää toimivan suhteellisen hyvin yleisellä accuracy-arvolla 0.98 ja hyvällä suorituskyvyllä useimmissa luokissa. Luokkien 1 ja 2 tapausten tunnistamisessa on kuitenkin parantamisen varaa.



Kuvio 22. Random forest -mallin hämmennysmatriisi

Class	Precision	Recall	F1-score	Support
0	0.99	1.00	1.00	18653
1	0.95	0.95	0.95	4763
2	1.00	0.83	0.90	286
3	0.97	0.94	0.95	4604
accuracy			0.98	28306
macro avg	0.98	0.93	0.95	28306
weighted avg	0.98	0.98	0.98	28306

Taulukko 2. Random Forest -mallin luokitusraportti

6.2 Gradient Boosting -algoritmin tulokset

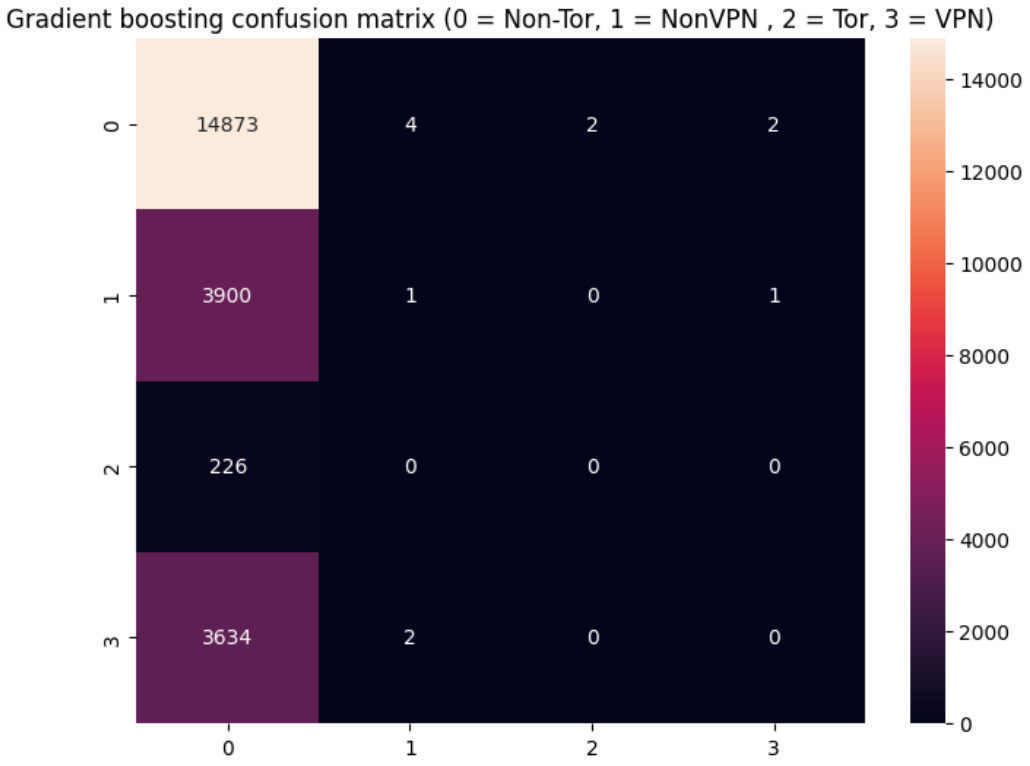
Gradient boosting -algoritmin luokitusraportista 3 voidaan nähdä, että se suoriutui luokittelutehtävästä huomattavasti random forest -algoritmia heikommin. Mallin kokonaistarkkuus, accuracy, on 0,66, mikä on suhteellisen alhainen luku. Tämä viittaa siihen, että malli tekee vääriä ennusteita datan suurimmalle osalle. Tarkastaltaessa luokkakohtaisia pisteitä voidaan huomata, että luokassa 0 ovat muita luokkia korkeammat precision- ja recall-arvot. Täydellinen recall-arvo (1.00) tarkoittaa, että malli tunnistaa oikein kaikki luokan 0 esiintymät, mutta precision-arvo ei ole kovin korkea, mikä puolestaan osoittaa, että malli saattaa ennustaa joitain esiintymiä luokkaan 0, jotka eivät edes itse asiassa kuulu kyseiseen luokkaan.

Kaikkien luokkien 1, 2 ja 3 precision-arvot ovat 0,00, mikä tarkoittaa, että malli ei tunnista oikein näiden luokkien esiintymiä ollenkaan. Näiden luokkien recall-arvo on myös 0,00, mikä osoittaa, että malli ei tunnista mitään todellista positiivista näille luokille. Myöskin kyseisten luokkien F1-arvot ovat kaikki 0,00, mikä viittaa siihen, että malli toimii erittäin huonosti näillä luokilla.

Tarkastaltaessa gradient boosting -luokittelijan hämmennysmatriisia 23 voidaan huomata, että malli luokitteli oikein kaikki luokan 0 todelliset esiintymät, joita oli 14873 kappaletta. Malli luokitteli virheellisesti luokan 0 esiintymiä neljä kappaletta luokasta 1 ja kaksi kappaletta sekä luokasta 2 että 3. Luokasta 1 malli luokitteli oikein vain yhden esiintymän. Suu-

rin osa luokan 1 esiintymistä luokiteltiin virheellisesti luokkaan 1 kuuluvaksi. Myöskin yksi kappale luokan 1 esiintymistä malli luokitteli virheellisesti kuuluvan luokkaan 3. Luokista 2 ja 3 malli ei osannut luokitella oikein mitään esiintymää. Luokille 1, 2 ja 3 yhtenäistä näyttäisi olevan se, että melkein kaikki luokkien esiintymät luokitellaan virheellisesti kuuluvan luokkaan 0.

Yhteenvetona voidaan todeta, että malli toimii suhteellisen hyvin luokalle 0, mutta vastaavasti ei toimi luokille 1, 2 ja 3. Näiden luokkien alhaiset precision- ja recall-arvot osoittavat, että malli ei tunnista oikein näiden luokkien esiintymiä käytännössä lainkaan. Mallin kokonaissuorituskyky on suhteellisen heikko, mikä viittaa siihen, että se ei ehkä sovellu annettuun luokitustehtävään. Gradient Boosting voi olla erittäin tehokas algoritmi, mutta sen täyden potentiaalin saavuttaminen vaatii korkean tason asiantuntemusta ja taitoa.



Kuvio 23. Gradient boosting -mallin hämmennysmatriisi

Class	Precision	Recall	F1-score	Support
0	0.66	1.00	0.79	14881
1	0.14	0.00	0.00	3902
2	0.00	0.00	0.00	226
3	0.00	0.00	0.00	3636
accuracy			0.66	22645
macro avg	0.20	0.25	0.20	22645
weighted avg	0.46	0.66	0.52	22645

Taulukko 3. Gradient boosting -mallin luokitusraportti

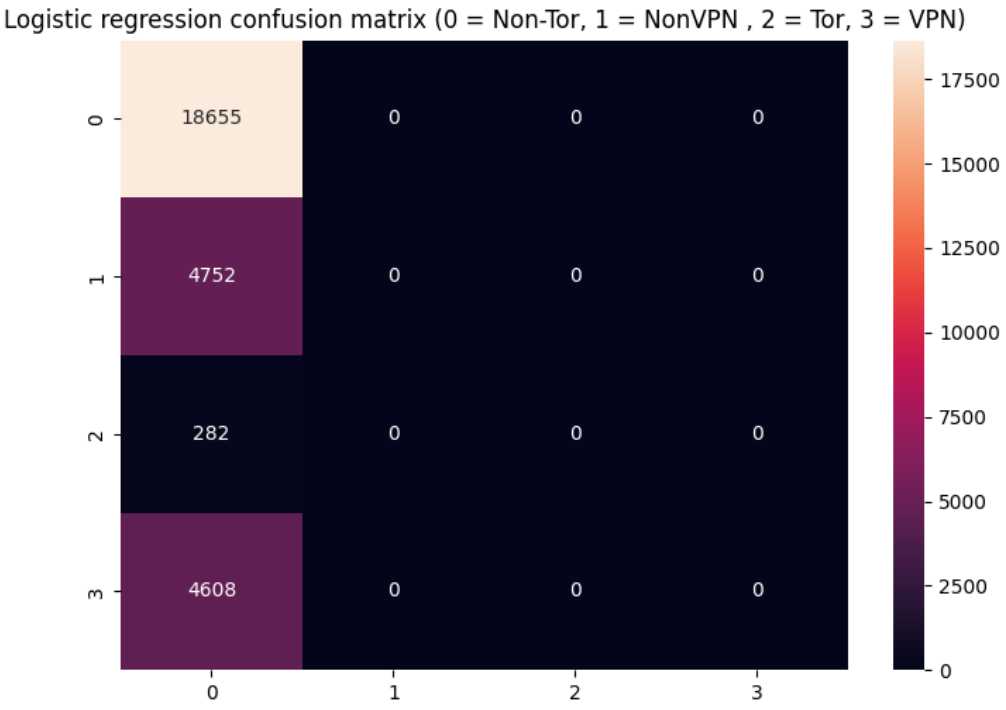
6.3 Logistic Regression -algoritmin tulokset

Kuten luvussa 6.2 esitetty gradient boosting -algoritmin tulokset, myöskin logistic regression -algoritmin mallin kokonaistarkkuus, accuracy, on vain 0,66, mikä on edelleen verrattain heikko tulos. Tämä puolestaan viittaa siihen, että malli tekee vääriä ennusteita merkittävälle osalle testidatasta. Kun tarkastellaan yksittäisten luokkien pisteitä, voidaan havaita, että luokkien 1, 2 ja 3 precision-, recall ja F1-arvot ovat kaikki 0,00, mikä tarkoittaa, että malli ei tunnista oikein näiden luokkien esiintymiä lainkaan. Luokan 0 recall-arvo on täydellinen arvolla 1,00, mikä tarkoittaa, että malli tunnistaa oikein kaikki luokan 0 esiintymät. Luokan 0 precision-arvo on kuitenkin suhteellisen alhainen, 0,66, mikä osoittaa, että malli saattaa ennustaa joitain tapauksia luokkana 0, jotka eivät itse asiassa kuulu kyseiseen luokkaan.

Logistic regression -mallin hämmennysmatriisista kuvassa 24 voidaan huomata, että malli luokitteli oikein ainoastaan kaikki luokan 0 esiintymät, mutta ei muiden luokkien esiintymiä. On myös olennaista mainita, että logistic regression -malli luokittelee kaikki luokkien 1, 2 ja 3 esiintymät virheellisesti kuuluvaksi luokkaan 0, kuten myös luvussa 6.2 esitetty gradient boosting -algoritmilla toimiva malli.

Macro avg- ja weighted avg-F1-arvot ovat 0,20 ja 0,52, mikä viittaa siihen, että yleisesti ottaen malli toimii huonosti. Yhteenvetona voidaan todeta, että malli ei toimi hyvin millekkään luokalle. Kaikkien luokkien alhaiset precision- ja recall-arvot osoittavat, että malli ei tunnista

oikein minkään luokan esiintymiä. Mallin kokonaissuorituskyky on suhteellisen heikko, mikä viittaa siihen, että se ei ehkä sovellu annettuun luokitustehtävään tai se vähintääkin vaatii parametrien virittämistä.



Kuvio 24. Logistic regression -algoritmin hämmennysmatriisi

Class	Precision	Recall	F1-score	Support
0	0.66	1.00	0.79	18655
1	0.00	0.00	0.00	4752
2	0.00	0.00	0.00	282
3	0.00	0.00	0.00	4608
accuracy			0.66	28297
macro avg	0.16	0.25	0.20	28297
weighted avg	0.43	0.66	0.52	28297

Taulukko 4. Logistic regression -mallin luokitusraportti

7 Yhteenveto

Darkneteistä on tulossa yhä suositumpia laittomien toimien, kuten huumekaupan, aseiden myynnin ja rahanpesun, harjoittamisessa. Tästä huolimatta darknet-verkkoja voidaan käyttää myös turvallisen ja yksityisen viestinnän tarjoamiseen toimittajille, ilmiantajille ja aktivisteille, joiden on suojeltava henkilöllisyytensä ja vältettävä valvontaa. Jotkin organisaatiot käyttävät darknettejä suojattuun tiedostojen jakamiseen tai tietyissä maissa sensuroitujen tai estettyjen verkkosivustojen isännöintiin. On kuitenkin tärkeää huomata, että darknettien käyttö lailliseen toimintaan voi silti sisältää samalla tavalla riskejä ja haasteita kuin sen käyttö laittomaan toimintaankin. Darknettejä ei säännellä tai valvota samalla tavalla kuin tavallista Internetiä, ja käyttäjät voivat silti olla alttiita hyökkäyksille ja huijauksille. Darknet-liikenteen analysointi voi tarjota olennaista tietoa rikollisverkostoista ja auttaa lainvalvontaviranomaisia ehkäisemään ja torjumaan rikollisuutta verkossa.

Tämän pro gradun teoriaosuudessa keskityttiin tekoälyyn ja sen osa-alueisiin sekä erityisesti Habibi Lashkari, Kaur ja Rahali (2020) tutkimukseen, joka toimi tutkielman aiheen inspiraationa. Tutkimuksen empiirinen osuus käsitteli Darknet 2020 -datasetin testaamista kolmella erilaisella koneoppimisalgoritmilla: random forest, gradient boosting ja logistic regression. Algoritmeista luokittelutehtävässä ylivoimaisesti parhaiten pärjäsi random forest yleisellä accuracy-arvollaan 0,98. Gradient boosting- ja logistic regression-algoritmeilla oli vaikeuksia tunnistaa luokkien 1 (NonVPN), 2 (Tor) ja 3 (VPN) esiintymät, mutta molemman osasivat kuitenkin tunnistaa luokan 0 (Non-Tor) esiintymät suhteellisen hyvin.

Darknet-liikenteen analysointi on monimutkainen tehtävä, joka vaatii edistyneitä teknisiä taitoja ja osaamista esimerkiksi data-analyysistä, tietoliikenteestä, verkkoarkkitehtuurista sekä -turvallisuudesta. Darknet-tietoliikenteen analyysin ala on suhteellisen uusi, ja siinä on vielä paljon tutkittavaa. Tästä syystä Habibi Lashkari, Kaur ja Rahali (2020) kehittämä malli, DIDarknet, onkin ensimmäisiä laatuaan sillä se hyödyntää syväoppimista konvoluutioneuroverkon muodossa tietoliikenteen luokitteluun. DIDarknet-mallin tulokset olivat huomattavasti tarkempia ja parempia kuin tässä tutkimuksessa esiintyvien koneoppimisalgoritmien tulokset.

Syväoppimisen hyödyntämisen kannalta samankaltaisissa darknet-tietoliikenteen luokittelemisen jatkotutkimuksissa olisi tärkeää ottaa huomion, miksi syväoppimisen menetelmää on käytetty niin vähän. Osaltaan tähän ongelmaan vaikuttaa vanhat ja puutteelliset datasetit, jotka ei imitoi oikeaa darknet-liikennettä tarpeeksi hyvin. Tämän vuoksi on vaikea saada uskottavia ja päteviä tuloksia, jos lähtödata on puutteellista. Toistaiseksi darknettien "sinkomaisen"luonteen vuoksi niiden reaaliaikaista tietoliikennettä on vaikea kaapata. Jotta jatkotutkimusten tulokset ovat mahdollisimman todenmukaisia ja uskottavia, on tärkeää panostaa uusien datasettien laatuun sekä nykyisen tutkimuksen laajentamiseen esimerkiksi Tor over VPN -liikenteen luokitteluun, kuten myös Habibi Lashkari ym. (2020, 12) tutkimuksessaan toteavat.

Lähteet

Bergman, Michael K. 2001. “White paper: the deep web: surfacing hidden value”. *Journal of electronic publishing* 7 (1).

Chertoff, Michael, ja Tobby Simon. 2015. “The impact of the dark web on internet goverannce and cyber security”.

Friedman, Jerome H. 2001. “Greedy function approximation: a gradient boosting machine”. *Annals of statistics,* 1189–1232.

Gayard, Laurent. 2018. *Darknet: Geopolitics and Uses.* John Wiley & Sons.

Goodfellow, Ian, Yoshua Bengio ja Aaron Courville. 2016. *Deep learning.* MIT press.

Gu, Jiuxiang, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai ym. 2018. “Recent advances in convolutional neural networks”. *Pattern recognition* 77:354–377.

Guyon, Isabelle, Steve Gunn, Masoud Nikravesh ja Lofti A Zadeh. 2008. *Feature extraction: foundations and applications.* Nide 207. Springer.

Habibi Lashkari, Arash, Gurdip Kaur ja Abir Rahali. 2020. “DIDarknet: a contemporary approach to detect and characterize the darknet traffic using deep image learning”. Teoksessa *2020 the 10th International Conference on Communication and Network Security,* 1–13.

Hilbe, Joseph M. 2009. *Logistic regression models.* CRC press.

Hosmer Jr, David W, Stanley Lemeshow ja Rodney X Sturdivant. 2013. *Applied logistic regression.* Nide 398. John Wiley & Sons.

Howard, Jeremy, ja Sylvain Gugger. 2020. *Deep Learning for Coders with fastai and PyTorch.* O’Reilly Media.

Jardine, Eric. 2015. “The Dark Web dilemma: Tor, anonymity and online policing”. *Global Commission on Internet Governance Paper Series,* numero 21.

Kaur, Shubhdeep, ja Sukhchandan Randhawa. 2020. “Dark web: a web of crimes”. *Wireless Personal Communications* 112 (4): 2131–2158.

Kumar, Gaurav, ja Pradeep Kumar Bhatia. 2014. “A detailed review of feature extraction in image processing systems”. Teoksessa *2014 Fourth international conference on advanced computing & communication technologies,* 5–12. IEEE.

LeCun, Yann, Yoshua Bengio ja Geoffrey Hinton. 2015. “Deep learning”. *nature* 521 (7553): 436–444.

Leung, Kenneth. 2022. “Micro, Macro & Weighted Averages of F1 Score, Clearly Explained”. Viitattu 25. huhtikuuta 2023. https://towardsdatascience.com/micro-macro-weighted-averages-of-f1-score-clearly-explained-b603420b292f#989c.

Mahesh, Batta. 2020. “Machine learning algorithms-a review”. *International Journal of Science and Research (IJSR).[Internet]* 9:381–386.

Mirea, Mihnea, Victoria Wang ja Jeyong Jung. 2019. “The not so dark side of the darknet: a qualitative study”. *Security Journal* 32 (2): 102–118.

Moore, Daniel, ja Thomas Rid. 2016. “Cryptopolitik and the Darknet”. *Survival* 58 (1): 7–38.

Natekin, Alexey, ja Alois Knoll. 2013. “Gradient boosting machines, a tutorial”. *Frontiers in neurorobotics* 7:21.

Nixon, Mark, ja Alberto Aguado. 2002. *Feature Extraction and Image Processing.* Newnes.

O’Shea, Keiron, ja Ryan Nash. 2015. “An introduction to convolutional neural networks”. *arXiv preprint arXiv:1511.08458.*

Osborne, Jason W. 2015. *Best practices in logistic regression.* Sage Publications. https://doi.org/https://doi.org/10.4135/9781483399041.

Patterson, Josh, ja Adam Gibson. 2017. *Deep learning: A practitioner’s approach.* "O’Reilly Media, Inc.".

Rudesill, Dakota S, James Caverlee ja Daniel Sui. 2015. “The deep web and the darknet: A look inside the internet’s massive black box”. *Woodrow Wilson International Center for Scholars, STIP* 3.

Sammut, Claude, ja Geoffrey I Webb. 2011. *Encyclopedia of machine learning.* Springer Science & Business Media.

Soska, Kyle, ja Nicolas Christin. 2015. “Measuring the longitudinal evolution of the online anonymous marketplace ecosystem”. Teoksessa *24th USENIX security symposium (USENIX security 15),* 33–48.

University of New Brunswick: Canadian Institute for Cybersecurity. 2016a. “Tor-nonTor dataset (ISCXTor2016)”. Viitattu 17. tammikuuta 2023. https://www.unb.ca/cic/datasets/tor.html.

———. 2016b. “VPN-nonVPN dataset (ISCXVPN2016)”. Viitattu 17. tammikuuta 2023. https://www.unb.ca/cic/datasets/vpn.html.

Warren, Jackson, Karson Fye, Drake Cullen, Halladay James E, Nathan Briner, Ram Basnet, Jeremy Bergen ja Tenzin Doleck. 2022. *CMUDarknet.* https://github.com/Karson-Fye/CMUDarknet.

Weimann, Gabriel. 2016. “Terrorist migration to the dark web”. *Perspectives on Terrorism* 10 (3): 40–44.

Zhang, Cha, ja Yunqian Ma. 2012. *Ensemble machine learning: methods and applications.* Springer.

Zhou, Zhi-Hua. 2021. *Machine learning.* Springer Nature.