Jani Ranta

Sähköverkon vikailmoitusten luokittelu koneoppimisen avulla

Opinnäytetyö
Kevät 2020
SeAMK Tekniikka
Tietotekniikka

SeAMK
SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Koulutusyksikkö: SeAMK Tekniikka

Tutkinto-ohjelma: Tietotekniikka

Suuntautumisvaihtoehto: Ohjelmointi

Tekijä: Ranta Jani

Työn nimi: Sähköverkon vikailmoitusten luokittelu koneoppimisen avulla

Ohjaaja: Juha Hirvonen

Vuosi: 2020 Sivumäärä: 34 Liitteiden lukumäärä: 0

Tässä työssä luotiin koneopetettuja neuroverkkoja sähköverkon vikailmoitusten automaattiseen luokitteluun. Tämän on tarkoitus säästää työntekijöiden aikaa.

Teoriaosuudessa perehdytään tekoälyyn ja sen eri kategorioihin. Eniten käsitellään koneoppimista, joka on tekoälyn alaluokka, sillä työssä käytetyt neuroverkot ovat yksi koneoppimisen malli. Teoriaosuuden lopuksi tutustutaan myös tekoälyn filosofiaan.

Työ toteutettiin käyttämällä TensorFlow-käyttöliittymää ja Keras-rajapintaa. Nämä työkalut esitellään omassa luvussaan.

Lopputuloksena luotiin tavoitteiden mukaiset neuroverkot, jotka oppivat luokittelemaan vikailmoitusten kuvia sekä kuvauksia. Työssä rakennettuja neuroverkkoja on tarkoitus tulevaisuudessa käyttää vikailmoitusten priorisointiin.

Avainsanat: tekoäly, koneoppiminen, TensorFlow, Keras, luokittelu, neuroverkko

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Faculty: SeAMK Technology

Degree programme: Information technology

Specialisation: Programming

Author/s: Ranta Jani

Title of thesis: Classification of electrical grid fault notifications using machine learning

Supervisor(s): Juha Hirvonen

Year: 2020 Number of pages: 34 Number of appendices: 0

In this thesis I created machine taught neural networks for automatic classification of electrical grid fault notifications. This is supposed to save workers time.

The theory section will familiarise itself with artificial intelligence and its various categories. Most of it is about machine learning, which is a subcategory of artificial intelligence, as neural networks used in this work are one model of machine learning. End of the theory section will also explore the philosophy of artificial intelligence.

This work was accomplished using the TensorFlow interface and the Keras interface. These tools are presented in their own chapter.

The end result created target-compliant neural networks that learn how to classify images of fault notifications as well as descriptions. Neural networks created in this thesis are to be used in future to prioritize fault notifications.

Keywords: artificial intelligence, machine learning, TensorFlow, Keras, classification, neural network

SISÄLTÖ

Opinnäytetyön tiivistelmä 1
Thesis abstract 2
SISÄLTÖ 3
Kuva-, kuvio- ja taulukkoluettelo 5
Käytetyt termit ja lyhenteet 6
1 Johdanto 7
1.1 Työn tausta 7
1.2 Työn tavoite 7
1.3 Työn rakenne 8
1.4 Yritysesittely 8
1.4.1 Järvi-Suomen Energia 8
1.4.2 Solteq Oyj 8
2 Mitä tekoäly on? 9
2.1 Tekoäly 9
2.1.1 Tekoälyn kategoriat 9
2.2 Koneoppiminen 10
2.2.1 Ohjattu opetus 11
2.2.2 Ohjaamaton opetus 12
2.2.3 Vahvistettu opetus 13
2.2.4 Syväoppiminen ja neuroverkot 13
2.3 Tekoälyn filosofiaa 14
2.3.1 Turingin testi 14
2.3.2 Kiinalainen huone -argumentti 15
3 TensorFlow ja Keras-rajapinta 16
3.1 TensorFlow 16
3.2 Keras 16
3.2.1 Mallin luominen, konfigurointi ja koulutus 16
3.2.2 Keras-mallin kerroksia 17
4 Käytännön osuus 19
4.1 Koulutusdata 19

4.1.1 Kuvausdata 19
4.1.2 Kuvadata 21
4.2 Datan täydennys 22
4.2.1 Kuvausdatan täydennys 22
4.2.2 Kuvadatan täydennys 23
4.3 Käytetyt neuroverkkomallit 24
4.4 Koulutuksen tulokset 26
4.4.1 Kuvauksien tulokset 26
4.4.2 Kuvien tulokset 28
4.5 Kuvausten yhteenveto 29
4.6 Kuvien yhteenveto 29
5 Johtopäätökset 30
5.1 Kuvaukset 30
5.2 Kuvat 30
5.3 Prioriteettityökalu 31
LÄHTEET 32

Kuva-, kuvio- ja taulukkoluettelo

Kuva 1. Esimerkkejä täydennetyistä kuvista. 24

Kuvio 1. Kuvauksien neuroverkko malli. 25

Kuvio 2. Kuvien neuroverkko malli. 26

Kuvio 3. Kuvauksien validoinnin tappiofunktio (x-akseli: epookki, y-akseli: tappiofunktion tulos) 27

Kuvio 4. Kuvauksien validoinnin tarkkuus (x-akseli: epookki, y-akseli: tarkkuus) .. 27

Kuvio 5. Kuvien validoinnin tappiofunktio (x-akseli: epookki, y-akseli: tappiofunktion tulos) 28

Kuvio 6. Kuvien validoinnin tarkkuus (x-akseli: epookki, y-akseli: tarkkuus) 28

Taulukko 1. Kuvauksien luokkanumerot. 19

Taulukko 2. Esimerkkejä kuvausdatasta. 20

Taulukko 3. Kuva datan keskijännite luokat. 21

Taulukko 4. Kuvadatan pienjänniteluokat. 22

Käytetyt termit ja lyhenteet

CNTK	Avoimen lähdekoodin kirjasto neuroverkkojen rakentamiseen
Epookki	Ajankohta, jolloin opetuksessa koulutusdata on käyty kokonaan läpi.
Gradient descent	Optimointialgoritmi, jota käytetään funktion minimoimiseen siirtymällä iteratiivisesti jyrkimmän laskeutumisen suuntaan.
Keras	Korkea tason ohjelmointirajapinta neuroverkkojen rakentamiseen ja opetukseen.
Python	Korkeatasoinen, tulkittu ja yleiskäyttöinen ohjelmointikieli.
Rajapinta	Määritelmä, jonka mukaan ohjelmat voivat tehdä pyyntöjä ja vaihtaa tietoja.
Sigmoid-funktio	Logistinen käyrä, jonka kuvaaja on S-muotoinen ja jolla voidaan mallintaa jonkin populaation kasvua. Aluksi kuvaajan kasvu on lähes eksponentiaalista, mutta kasvu hidastuu, kun ympäristön rajat tulevat vastaan.
Softmax-funktio	Normalisoi syöttöarvon vektoriksi, joka seuraa todennäköisyysjakaumaa, jonka kokonaissumma on 1.
TensorFlow	Avoimen lähdekoodin käyttöliittymä koneoppimiselle.
Theano	Ohjelmointikirjasto Pythonille matemaattisten lausekkeiden ilmaisuun ja optimointiin.

1 Johdanto

1.1 Työn tausta

Järvi-Suomen Energialla on käytössä sähköverkon vikailmoituspalvelu, johon palvelun käyttäjät voivat lähettää sähköverkon vikailmoituksia. Vikailmoituksessa on mahdollista kuvailla havaintoja tekstikenttään sekä lisätä vikaan liittyvä kuva. Vikailmoitukseen lisätään myös vian sijainti. Vikailmoituksen sijainti saadaan joko puhelimen GPS-tiedoista tai käyttäjä voi itse valita kartasta sijainnin.

Vikailmoitukset käsitellään manuaalisesti ylläpitäjien toimesta. Vikailmoitusten manuaalisessa käsittelyssä on ongelmana se, että vikailmoituksia ei ole järjestelty mitenkään. Manuaalinen käsittely vie paljon aikaa, sekä oikeasti tärkeiden vikailmoitusten löytäminen on haastavaa, jonka seurauksena vikojen korjaaminen voi viivästyä.

Järvi-Suomen Energian käyttöinsinööri Tomi Öster on kerännyt ja luokitellut käsiteltyjä vikailmoituksia, joiden avulla voidaan kouluttaa neuroverkkoa luokittelemaan uusia vikailmoituksia.

1.2 Työn tavoite

Tavoitteena on selvittää, onko mahdollista kouluttaa koneopetettua neuroverkkoa luokittelemaan sähköverkon vikailmoituksia.

Luokittelun avulla voidaan järjestää vikailmoitukset prioriteettijärjestykseen. Prioriteettijärjestyksessä olevat vikailmoitukset hyödyttävät sekä asiakkaita että sähköverkon ylläpitäjiä, koska tärkeiden ja kriittisten vikailmoituksen löytäminen nopeutuu, mikä puolestaan nopeuttaa kyseisten vikojen korjaamista.

Työssä myös selvitetään, onko kerätyn koulutusdatan määrä riittävä, jotta sitä voidaan käyttää neuroverkon opetukseen.

1.3 Työn rakenne

Luvussa 1 käydään läpi työn taustaa ja tavoitteita, sekä esitellään yhteistyötahot. Luvussa 2 käsitellään, mitä tekoäly on, ja miten se voidaan määritellä. Luvussa myös käydään myös läpi tekoälyn alaluokkia ja filosofiaa. Tämän jälkeen luvussa 3 esitellään TensorFlow-käyttöliittymää sekä Keras-rajapintaa ja sen käyttämistä.

Luvussa 4 käydään läpi luokittimien koulutusdataa ja niiden täydennysmenetelmiä. Luvussa myös esitetään työn tulokset. Viimeisessä luvussa, luvussa 5, pohditaan työn tuloksia, sekä esitellään mahdollisia jatkotoimenpiteitä.

1.4 Yritysesittely

Työ toteutettiin Solteqille, jonka asiakkaana toimi Järvi-Suomen Energia.

1.4.1 Järvi-Suomen Energia

Järvi-Suomen Energia kuuluu Suur-Savon Sähkö -konserniin ja jakaa sähköenergiaa Suur-Savon Sähkö -konsernin alueella yli 100 000 asiakkaalle. Heidän asiantuntijansa työskentelevät noin 27 000 kilometrin mittaisen sähköverkoston suunnittelun, rakennuttamisen, käytön ja kunnossapidon parissa. Järvi-Suomen Energialla on käytössä vikailmoituspalvelu, jonka avulla he ovat keränneet työssä käytetyn opetusdatan. (Suur-Savon Sähkö [Viitattu 4.6.2020].)

1.4.2 Solteq Oyj

Solteq Oyj on pohjoismainen ohjelmistoratkaisujen tarjoaja, joka on erikoistunut liiketoiminnan digitalisaatioon sekä toimialakohtaisiin ohjelmistoihin. Työntekijöitä Solteqilla on n. 600 ja toimipisteitä on Suomen lisäksi Ruotsissa, Norjassa, Puolassa, Tanskassa ja Iso-Britanniassa. Solteqin asiakkaiden toimialoihin kuuluu muun muassa kauppa, teollisuus, energia ja palvelut. (Solteq 2019.)

2 Mitä tekoäly on?

Tässä luvussa käsitellään, mitä tekoäly on, ja käydään läpi koneoppimisen teoriaa. Luvussa käsitellään myös koneälyn filosofisia ajatuksia.

2.1 Tekoäly

Nykyisin on lähes mahdotonta olla törmäämättä tekoälyä koskeviin aiheisiin, mutta sen määrittely on hankalaa, koska eri ihmisille se voi merkitä erilaisia asioita. Joidenkin mielestä se on keinotekoinen elämänmuoto, joka ylittää ihmisen älykkyyden, kun taas toisille tekoäly on melkein minkä tahansa datan käsittelyä. (Reaktor [Viitattu 7.4.2020].)

Tekoälyn määrittelyn haastavuutta hankaloittaa se, että tekoäly-nimitystä on alettu käyttämään määrittelemään melkein mitä tahansa tilastotieteestä käsinkoodattuihin jos-niin-lausekkeisiin. (Reaktor [Viitattu 7.4.2020]). Esimerkiksi Cambridgen yliopiston sanakirjassa tekoäly määritellään seuraavasti:

> Tutkimus siitä, miten tuottaa koneita, joilla on joitakin ominaisuuksia, joita ihmismieli on, kuten kyky ymmärtää kieltä, tunnistaa kuvia, ratkaista ongelmia ja oppia. (Cambridge Dictionary [Viitattu 7.4.2020]).

Tekoälyn keskeisiä aihepiirejä ovat muun muassa:

- Koneoppiminen
- Syväoppiminen
- Datatiede
- Robotiikka. (Reaktor [Viitattu 15.4.2020])

2.1.1 Tekoälyn kategoriat

Integratiivisen biologian, tietojenkäsittelytieteen ja tekniikan apulaisprofessori Arend Hintze luokitteli tekoälyn neljään kategoriaan. Hänen kategorioissaan on nykyisin nähtäviä tekoälyn tekniikoita sekä teoreettisia tekoälytekniikoita, joita ei ole vielä olemassa. (Rouse 2020a.)

Tyyppi 1 on reaktiiviset koneet. Näillä tekoälyjärjestelmillä ei ole muistia ja ne ovat tehtäväkohtaisia. Esimerkkinä tästä on Deep Blue, IBM:n shakkia pelaava koneäly. Deep Blue voitti Garry Kasparovin shakkiottelussa 1990-luvulla. Deep Blue pystyy tunnistamaan shakkilaudan nappuloita ja tekemään ennusteita sen avulla, mutta koska sillä ei ole muistia, se ei voi hyödyntää menneitä kokemuksia. (Rouse 2020a.)

Tyyppin 2 tekoälyllä on rajoitettu muisti. Tämän tyypin tekoälyt pystyvät hyödyntämään aikaisempia kokemuksiaan päätöksenteossa. Muun muassa itsestään ajavat autot hyödyntävät tämänkaltaista tekoälyä. (Rouse 2020a.)

Tyyppi 3 on mielen teoria, joka on psykologian termi. Tekoälyyn sovellettuna se tarkoittaa, että järjestelmä ymmärtäisi tunteita. Tällaista tekoälyä ei ole vielä olemassa, mutta tämäntyyppinen tekoäly pystyy käsittelemään aikeita ja ennustamaan käyttäytymistä. (Rouse 2020a.)

Tyyppi 4 on itsetietoisuus. Tämän kategorian tekoälyllä on itsetunto, joka antaa sille tietoisuuden. Itsetietoisuuden omaavat koneet ymmärtävät oman nykytilansa. Tällaista tekoälyä ei ole vielä olemassa. (Rouse 2020a.)

2.2 Koneoppiminen

Koneoppiminen on tekoälyn osa-alue, jonka juuret ovat tilastotieteessä. Koneoppimisen voidaan ajatella olevan tiedon eristämistä datasta. (Reaktor [Viitattu 14.4.2020].)

Koneoppiminen voidaan jakaa osa-alueisiin niillä ratkaistavien ongelmien luonteiden mukaan. Karkeasti jaettuna osa-alueita ovat ohjattu oppiminen, ohjaamaton oppiminen sekä vahvistusoppiminen (Luvuissa 2.2.1, 2.2.2 ja 2.2.3 käydään osa-alueet tarkemmin). (Reaktor [Viitattu 14.4.2020].)

Koneoppimisen avulla voidaan luoda ohjelmia, jotka pystyvät opetella ennustamaan tuloksia ilman, että sitä on nimenomaisesti ohjelmoitu tekemään niin. Koneoppimisen algoritmit käyttävät historiatietoja syötteenä uusien tulosarvojen ennustamiseen. (Rouse 2020b.)

Koneoppimisen käyttökohteita ovat muun muassa suositusmoottorit, petosten havaitseminen, roskapostisuodatus, liiketoimintaprosessien automaatio ja ennakoiva ylläpito. Parhaiten tunnetuin käyttökohde on Facebookin suositusjärjestelmä, joka räätälöi käyttäjän uutissyötteen. Esimerkiksi, jos käyttäjä usein pysähtyy lukemaan tietyn ryhmän postauksia, suositusjärjestelmä alkaa näyttämään enemmän kyseisen ryhmän toimintaa syötteessä. (Rouse 2020b.)

2.2.1 Ohjattu opetus

Ohjatut koneoppimisalgoritmit on suunniteltu oppimaan esimerkkien avulla. Ohjatun oppimisalgoritmin koulutusdata koostuu pareista, jotka sisältävät syötettävän datan sekä oletetun tuloksen. Koulutuksen aikana algoritmi etsii datasta kuvioita, jotka korreloivat haluttujen tulosten kanssa. Ohjatun oppimismallin tavoite on ennustaa oikea luokka syötetylle datalle. Perusmuodossaan ohjatun oppimisalgoritmin voi kirjoittaa seuraavasti:

$$Y = f(x) \tag{1}$$

missä

Y	on ennustettu lopputulos
f	on kartoitusfunktio, joka luodaan opetuksen aikana
x	on syöttöarvo kartoitusfunktiolle (Wilson 29.9.2019.)

Ohjattu opetus voidaan jakaa kahteen kategoriaan, jotka ovat **luokittelu** ja **regressio** (Wilson 29.9.2019).

Luokittelualgoritmille annetaan koulutuksen aikana datapisteitä, joilla on määritetty luokka. Algoritmin tehtävänä on ottaa syöttöarvo ja määritellä sille luokka annettujen koulutustietojen perusteella. (Wilson 29.9.2019.)

Yleisin esimerkki luokittelusta on sähköpostien määrittely roskapostiksi. Algoritmille syötetään koulutusdatana sähköposteja, joista osa on roskapostia ja osa ei. Kun valittavana on kaksi luokkaa, niin luokittelu on binääristä. Malli löytää datasta ominaisuuksia, jotka korreloivat kumpaankin luokkaan ja luo kaavan (1) mukaisen

kartoitusfunktion. Tämän jälkeen malli käyttää funktiota roskapostien tunnistamiseen. (Wilson 29.9.2019.)

Luokittelussa käytetään lukuisia algoritmeja. Käytettävä algoritmi valitaan datan sekä tilanteen mukaan. Suosittuja luokittelualgoritmeja ovat:

- lineaariset luokittimet
- päätöspuut
- satunnaiset metsät
- tukivektorikoneet. (Wilson 29.9.2019.)

Regressioalgoritmin tavoitteena on ennustaa jatkuvaa määrää, kuten myynti, tulot ja testitulokset. Regressioprosessissa malli yrittää löytää tärkeän suhteen riippuvaisten ja riippumattomien muuttujien välillä. (Wilson 29.9.2019.)

2.2.2 Ohjaamaton opetus

Ohjaamaton opetus toimii ilman valmiita esimerkkejä. Tämän osa-alueen malleissa hankaluutena on oikeiden vastausten puute, koska oikeita luokkia ei ole, ja ei voida rakentaa mallia, joka tulostaa oikean vastauksen syötteen saadessaan. Mallin toimivuuden arviointi on hankalaa, koska ei voida tarkistaa, miten malli ennustaa testidatan oikeat vastaukset. (Reaktor [Viitattu 14.4.2020].)

Mallille syötetään luokittelematonta dataa, ja tavoitteena on löytää siitä jonkinlainen rakenne (Khatun 10.7.2018). Rakenteella voidaan esimerkiksi tarkoittaa **visualisointia**, jossa samankaltaiset syötteet sijoitetaan lähelle toisiaan ja erilaiset kauas toisistaan. Tällä voidaan myös tarkoittaa **ryvästämistä**. Siinä tavoite on ryhmien muodostus, joissa esimerkit ovat samankaltaisia keskenään, mutta erilaisia muihin ryhmiin verrattuna. (Reaktor [Viitattu 14.4.2020].)

Muutaman vuoden aikana on myös noussut tärkeään asemaan **generatiivinen oppiminen**, joka on ohjaamatonta opetusta. Siinä generatiivinen malli voi luoda alkuperäisestä datasta lisää samankaltaista dataa. Esimerkiksi voidaan tehdä aidon näköinen kuva täysin tietokoneen luoman ihmisen kasvoista. (Reaktor [Viitattu 14.4.2020].)

2.2.3 Vahvistettu opetus

Vahvistusoppimista käytetään silloin, kun operoidaan tekoälyagenttia monimutkaisissa tilanteissa. Tekoälyagentti voi olla muun muassa itse ajava auto tai robotti. Vahvistusopetusta käytetään myös tilanteissa, joissa oikea ja väärä ratkaisu ei ole välittömästi selvä. (Reaktor [Viitattu 14.4.2020].)

Vahvistusopetuksessa agentti voi oppia interaktiivisessa ympäristössä omien tekojensa ja kokemustensa avulla. Opetus tapahtuu palkintojen avulla, joita käytetään positiivisen ja negatiivisen käyttäytymisen signaaleina. Agentin tavoitteena on löytää toimintamalli, joka maksimoi agentin kokonaiskumulatiivisen palkinnon. (Bhatt 2018.)

2.2.4 Syväoppiminen ja neuroverkot

Syväoppiminen on koneoppimismenetelmä, jossa muodostetaan verkosto prosessointiyksiköiden (neuroneiden) kerroksista. Järjestelmän prosessoima tieto kuljetetaan kerrosten läpi vuoron perään. Verkostoa kutsutaan **neuroverkoksi**, koska sen toiminta on samanlainen kuin biologisissa neuroverkoissa. (Reaktor [Viitattu 24.4.2020].)

Neuroverkkojen uskotaan olevan paras lähestymistapa tekoälyn kehittämiseen. Simuloimalla alisymbolista tiedonkäsittelyä neuroverkkojen tasolla tuloksena saadaan älykkyyttä. Simuloinnissa ei huomioida oikeiden hermosolujen sisäisiä toimintamekanismeja, koska tavoitteena ei ole biologisten järjestelmien simulointi. (Reaktor [Viitattu 24.4.2020].)

Neuroverkko koostuu syöttö-, piilo- ja ulostulokerroksista. **Syöttökerros** vastaanottaa numeerisen esityksen datasta. **Ulostulokerros** palauttaa ennustuksen ja **piilotetut kerrokset** hoitavat suurimman osan laskennasta. (Liang 20.10.2018.)

Jokaisella neuronilla on aktivointifunktio, joka päättää neuronin aktivoitumisesta sen syöttöarvojen painotetun summan perusteella (Sharma 30.3.2017).

Neuroverkon opetusprosessissa tärkeässä osassa on painoarvot. Sen jälkeen, kun järjestelmä on ennustanut tuloksen, se arvioi, kuinka hyvä ennustus oli suhteessa odotettuun tulokseen tappiofunktion avulla. Tappiofunktion arvo on sitä suurempi, mitä suurempi ennustuksen ja odotetun tuloksen välinen ero on. Käyttäen taaksepäin etenemistä ja **gradient descent** -menetelmää, järjestelmä päivittää jokaisen kerroksen neuronien painoarvot tappiofunktion vastaiseen suuntaan. Tämän seurauksena seuraavalla iteraatiolla tappiofunktion tuloksen pitäisi olla pienempi. (Liang 20.10.2018.)

Neuroverkon tarkkuus saadaan laskemalla, kuinka monta kertaa neuroverkko on ennustanut oikean luokan, joka jaetaan tehtyjen ennustusten kokonaismäärällä (TensorFlow 6.4.2020).

Neuroverkkoa kouluttaessa data jaetaan kahteen osaan, nämä osat ovat koulutusdata ja validointidata. Koulutusdataa käytetään neuroverkon painoarvojen opetuksessa. Validointidatan tarkoituksena on testata neuroverkon tarkkuutta datalla, jota se ei ole aikaisemmin nähnyt. (Brownlee 6.1.2017.)

2.3 Tekoälyn filosofiaa

Filosofia on myös osa tekoälyä, koska tekoälyn olemus herättää filosofisia kysymyksiä. Esimerkiksi voidaan pohtia älykkään toiminnan edellytyksiä, kuten mielen olemassaoloa. (Reaktor [Viitattu 8.4.2020].)

Käytännössä tekoälyn filosofialla ei ole suurta vaikutusta sen harjoittamiseen, samaan tapaan kuin tieteen filosofialla ei ole suurta vaikutusta tieteen harjoittamiseen. Käytännön ongelmia ratkoessa ei mietitä liikaa, onko koneäly oikeasti älykäs vai vaikuttaako se vain älykkäältä. (Reaktor [Viitattu 8.4.2020].)

2.3.1 Turingin testi

Tietojenkäsittelytieteen isänä pidetty Alan Turing (1912–1954) oli englantilainen matemaatikko ja loogikko. Turingin tunnetuin kontribuutio tekoälyn alalla on imitaatiopeli. (Reaktor [Viitattu 8.4.2020].)

Ilmaisua ”Turingin testi” käytetään yleisemmin viittaamaan käyttäytymiskokeisiin, jotka koskevat mielen, ajatuksen tai älykkyyden läsnäoloa oletetusti ajattelevissa entiteeteissä (Stanford 2016).

Turingin testissä ihmishaastattelija on vuorovaikutuksessa kahden pelaajan kanssa vaihtamalla kirjoitettuja viestejä. Laite läpäisee testin, jos haastattelija ei pysty erottamaan, kumpi pelaajista on laite ja kumpi ihminen. Läpäisemisen perusteluna on se, että jos haastattelija ei pysty tunnistamaan laitetta vapaassa luonnollisella kielellä käydyssä keskustelussa, on laitteen täytynyt saavuttaa älykkyys, joka vastaa ihmisen tasoa. (Reaktor [Viitattu 8.4.2020].)

2.3.2 Kiinalainen huone -argumentti

Argumentti ja ajatuskokeilu kiinalainen huone on amerikkalaisen filosofin John Searlen (1932-) julkaisema väitös, joka ollessaan oikeassa, kumoaisi Turingin testin. Siitä on tullut yksi tunnetuimmista perusteluista viimeaikaisessa filosofiassa. (Stanford 2020.)

Perustelussa Searle kuvittelee olevansa yksin huoneessa, jossa on ohjeistukset kiinalaisiin kirjoitusmerkkeihin. Huoneeseen sujautetaan kiinalaisia merkkejä oven alta. Searle ei ymmärrä mitään kiinalaisista merkeistä, mutta silti hän pystyy tulkitsemaan merkkejä ohjeistusten avulla ja onnistuu lähettämään oikeanlaiset merkit takaisin oven alitse, jonka seurauksena ulkopuoliset virheellisesti olettavat, että huoneen sisällä oleva henkilö ymmärtää kiinalaisia merkkejä. (Stanford 2020.)

Väitöksen suppea johtopäätös on, että tietokone voidaan saada näyttämään kieltä ymmärtävänä, mutta todellisuudessa se ei voi tuottaa todellista ymmärrystä. Searle väittää, että ajatuskokeilu korostaa sitä, että tietokoneet eivät ymmärrä merkitystä tai semantiikkaa. Hänen mielestään tietokoneet käyttävät syntaktisia sääntöjä merkkijonon manipulointiin. (Stanford 2020.)

Väitöksen laajempi päätelmä on teoria siitä, että ihmisen mielet eivät ole tietokoneen kaltaisia laskennallisia koneita. Sen sijaan Searlen mukaan mielen on oltava seurausta biologisista prosesseista ja tietokoneet voivat parhaimmillaan vain simuloida näitä prosesseja. (Stanford 2020.)

3 TensorFlow ja Keras-rajapinta

Tässä luvussa käydään läpi TensorFlow'ta ja Keras-rajapintaa.

3.1 TensorFlow

TensorFlow on käyttöliittymä koneoppimisen algoritmien ilmaisemiseen ja toteutus tällaisten algoritmien toteuttamiseksi. Käyttöliittymän avulla ilmaistu laskenta voidaan suorittaa vähäisillä muutoksilla eri laitteissa, jotka vaihtelevat mobiililaitteista tuhansien GPU-korttien järjestelmiin. Järjestelmä on joustava ja sitä voidaan käyttää ilmaisemaan monenlaisia algoritmeja, kuten koulutus- ja päättelyalgoritmeja sekä syvän neuroverkon malleja. TensorFlow'ta käytetään myös tutkimusten tekemiseen ja koneoppimisjärjestelmien tuotantoon viemiseen monilla aloilla, joiden käyttökohteisiin kuuluu puheentunnistus, konenäkö, robotiikka, tiedonhaku, luonnollinen kielten käsittely ja maantieteellisten tietojen keruu. (Abadi ym. 2015, 1.)

3.2 Keras

Keras on korkean tason rajapinta neuroverkoille, ~~ja~~ se on kirjoitettu Pythonilla. Sitä voidaan käyttää TensorFlow'n, CNTK:n tai Theanon päällä. Rajapinnan kehityksessä keskityttiin nopean kokeilun mahdollistamiseen.

Keras-rajapinnan tavoitteita ovat muun muassa

- käyttäjäystävällisyys
- modulaarisuus
- skaalautuvuus. (Chollet 2015d.)

3.2.1 Mallin luominen, konfigurointi ja koulutus

Keras-rajapinnan avulla neuroverkkoja voidaan luoda syöttämällä lista kerrosinstansseja Sequential-mallin rakentajaan. Sequential-malli on lineaarinen

pino kerroksia, ja sisältää siis tiedot syöttö- ja ulostulokerrosten ko'oista sekä piilokerrosten määrästä ja ko'oista. Mallille voi myös syöttää kerroksia add-metodilla. Mallin ensimmäiselle kerrokselle eli syöttökerrokselle täytyy määritellä syöttödatan eli käsiteltävän datan muoto, loput kerrokset päättelevät niille tulevan datan muodon automaattisesti. (Chollet 2015c.)

Ennen koulutusta täytyy konfiguroida mallin koulutusprosessi, joka suoritetaan compile-metodilla. Compile-metodille annetaan kolme parametria, jotka ovat optimoija, tappiofunktio sekä lista mitattavia arvoja. (Chollet 2015c.)

Mallin koulutus tapahtuu fit-funktiolla, jolle annetaan syöttödata, epookit ja validointidata tai desimaaliarvo, joka määrittää, kuinka suuri osa syöttödatasta käytetään validointiin (Chollet 2015g).

3.2.2 Keras-mallin kerroksia

Keras-rajapinta tarjoaa suuren määrän neuroverkkokerroksia sekä mahdollistaa myös omien kerrosten luonnin.

Dense-kerros on tyypillinen tiiviisti liitetty neuroverkkokerros. (Chollet 2015a).

2D-konvoluutiokerros luo konvoluutiokernelin, joka on kierretty kerroksen tulon kanssa tuottamaan ulostulon tensori (Chollet 2015b). Konvoluutiokerroksen avulla voidaan esimerkiksi tuoda esille kuvista piirteitä (Brownlee 17.4.2019).

Pooling-kerroksen tavoite on vähentää laskennan monimutkaisuutta pakkaamalla syöttöarvon koko pienemmäksi. Kerroksen tyyppinä voi olla maksimi-, minimi- tai keskiarvo. (Chollet 2015f.)

Flatten-kerros muuntaa syöttöarvon tasaiseksi. Esimerkiksi, jos 64 neuronia syöttää tasoituskerrokselle matriisia, jonka koko on 32x32, niin kerroksen ulostulona tulee olemaan vektori, jonka koko on 65536. (Chollet 2015a.)

Batch normalization -kerroksen avulla normalisoidaan aikaisemman kerroksen aktivoinnit erissä (Chollet 2015e).

Embedding-kerros muuntaa positiiviset kokonaisluvut (indeksit) kiinteän kokoisiksi vektoreiksi (Chollet 2015a).

Dropout-kerroksessa asetetaan koulutuksen aikana murto-osa syöttöarvoista arvoon 0. Toisin sanottuna valittujen neuronien syöttöarvot jätetään huomioimatta. Tämän avulla estetään neuroverkon ylikouluttamista. (Chollet 2015a.)

Kerroksille annetaan myös aktivointifunktio, joka päättää neuronin aktivoitumisesta sen syöttöarvojen painotetun summan perusteella. Oletuksena käytetään lineaarista aktivointifunktiota. (Chollet 2015a.) Yleisimmin käytetty aktivointifunktio on **oikaistu lineaarinen yksikkö** (eng. rectified linear unit), Keras-rajapinnassa tätä kutsutaan nimellä relu. Oikaistu lineaarinen yksikkö -funktio palauttaa tuloarvon suoraan, jos se on positiivinen, muulloin se palauttaa arvon 0. (Brownlee 6.1.2019.)

4 Käytännön osuus

Tässä luvussa käsitellään kuvaus- ja kuvaluokittimien koulutusdataa, jota käytettiin opettamaan neuroverkkoja luokittelemaan sähköverkon vikailmoituksia. Lisäksi käydään läpi datan täydennysmenetelmiä, joilla koulutusdatan määrää saatiin kasvatettua. Luku myös esittelee luokittelua varten kehitetyt neuroverkkomallit ja niillä saadut tulokset.

4.1 Koulutusdata

Neuroverkkojen koulutuksessa käytettiin Järvi-Suomen Energian vikailmoituspalvelulla kerättyä dataa, joka oli luokiteltu heidän toimestaan. Molempien neuroverkkojen koulutusdatasta 30 % käytettiin validointiin.

4.1.1 Kuvausdata

Kuvausdata sisältää vapaamuotoisen tekstin, jossa kuvaillaan vian havaintoja, sekä kuvauksen luokan numero. Taulukko 1 esittelee kuvauksien luokkanumerot, ja taulukko 2 näyttää esimerkkejä kuvausdatasta.

Taulukko 1. Kuvauksien luokkanumerot.

Luokan nimi	Luokan numero	Kuvauksien määrä
Keskijännite	0	112
Nollavika	1	67
Pienjännite	2	669
Jokin muu	3	854

Kuten taulukossa 1 näkyy, keskijännite- ja nollavikaluokissa kuvauksien määrä on huomattavasti pienempi kuin muissa luokissa, minkä vuoksi niiden määrää on kasvatettu datan täydennysmenetelmillä. Luvussa 4.2.1 käydään tarkemmin läpi käytettyjä menetelmiä.

Taulukko 2. Esimerkkejä kuvausdatasta.

Kuvaus	**Luokan numero**
Viimeisen kuuden tunnin aikana, useita katkoja. Täällä päin ainakin tuulee reilusti. Muuntajalle ja siitä pienjännitelinja talolle kulkee tienviertä suurimman osan matkasta. Tie on aurattu aaton Pyryn jälkeen, jolloin huomasin ja ilmoitin vikakeskukseen, että 20 kV:n linjasta on yksi kaapeli poikki. Puissa on mäellä edelleen runsaasti lunta ja latvakatkoja on jonkin verran seudun metsissä.	0
Linjalla valokaari kun sähköä yritetään kytkeä päälle.	0
Kaatunut puu nojaa 20kV -ilmakaapeliin.	0
Osassa taloa valot palavat normaalisti ja osassa himmeästi. Olisiko yhdessä vaiheessa jotain vialla?	1
Sähköt meni muutama minuutti sitten. Valot sammui, mutta tiskikone ja hella toimi vähän aikaa sen jälkeen. Sammutin ne. Lieneekö taas nollavika? Osa valoista palaa himmeänä, osa sammui kokonaan.	1
Puu kaatunut sähköjohdon päälle. Sähköt ei ole poikki.	2
Kuusen latva roikkuu sähköjohdon päällä.	2
Sähköt ollut poissa jo useamman tunnin	3
Sähköt kokonaan pois.	3

4.1.2 Kuvadata

Kuvadatalla oli kaksi pääluokkaa: keskijännite ja pienjännite. Keskijänniteluokka koostuu taulukossa 3 mainituista alaluokista, ja pienjänniteluokka koostuu taulukon 4 alaluokista.

Koulutuksessa käytetyt kuvat ovat peräisin vikailmoituspalvelusta sekä verkkoasentajien ottamista kuvista. Keskijänniteluokkaan Järvi-Suomen Energia toimitti koulutuksen aikana lisäkuvia 298 kappaletta, joita ei luokiteltu alaluokkiin.

Taulukko 3. Kuvadatan keskijänniteluokat.

Luokka	**Kuvien määrä**
Iso puu ilmajohdolla	133
Johtimet maassa	54
Pylväs poikki	16
Jokin muu	22
Tiealue	14
Vähäinen vaurio	159
Lisäkuvia	298

Pienjänniteluokkaan toimitettiin myös 648 kappaletta lisäkuvia koulutuksen aikana, niitä ei myöskään luokiteltu alaluokkiin.

Taulukko 4. Kuvadatan pienjänniteluokat.

Pienjännite	Kuvien määrä
AMKA maassa	120
AMKA poikki	25
Iso puu AMKA:lla	362
Pj-pylväs poikki	28
Tiealue	74
Vähäinen vaurio	341
Lisäkuvia	648

4.2 Datan täydennys

Molempien datasettien laajentamiseen käytettiin datan täydennysmenetelmiä. Datan täydennyksellä tavoiteltiin parempaa ennustustarkkuutta luokkiin, joissa datan määrä oli vähäinen.

Datan täydennysmenetelmien avulla luotiin kopiodataa, joka on hieman erilaista kuin alkuperäinen. Näiden menetelmien avulla saatiin luotua enemmän luokiteltua dataa, jota hyödynnettiin neuroverkon koulutuksessa.

4.2.1 Kuvausdatan täydennys

Datan täydennyksellä luotiin uusia kuvaksia, joissa sanat ovat eri järjestyksessä ja lisättiin kyseisen luokan avainsanoja kuvaukseen.

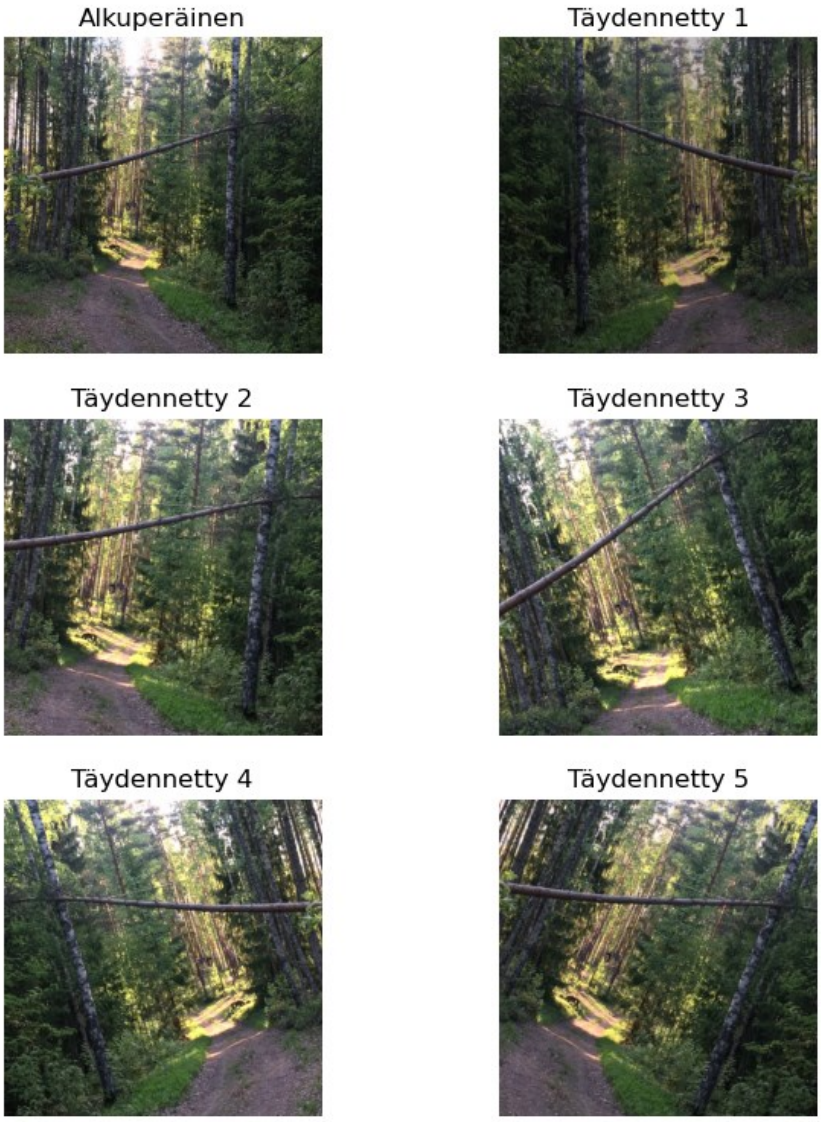
Avainsanoja lisättiin keskijännite- ja nollavikaluokkiin. Keskijänniteluokan avainsanoja on muun muassa valokaari, surisee, välähdys, voimalinja, 20 kV, suurjännite ja monta muuta keskijännitevikaan viittaavaa. Nollavian avainsanoihin kuuluu himmeä, sähköisku, alijännite, vajaa virta, nollajohto sekä monia muita nollavikaa kuvailevia sanoja.

4.2.2 Kuvadatan täydennys

Datan täydennyksellä luotiin 5 kopiota jokaisesta kuvasta, niitä muokattiin seuraavasti:

- Kuvan kirkkautta muunneltiin välillä 20 – 100 % (imitoidaan erilaista valaistusta)
- Zoomausta 0 – 50 %
- Kuvaa kierrettiin enintään 20 astetta (imitoidaan erilaista kuvauskulmaa)
- Kuvasta tehtiin horisontaalinen käännös (peilikuva)

Kuva 1 esittelee esimerkkejä kuvadatan täydennyksestä.



Kuva 1. Esimerkkejä täydennetyistä kuvista.

4.3 Käytetyt neuroverkkomallit

Neuroverkot luotiin luvussa 3.2 esitetyn Keras-rajapinnan avulla, hyödyntäen luvussa 3.2.2 mainittuja neuroverkkokerroksia.

Neuroverkkojen parametriarvot valittiin koulutuksessa saatujen tulosten ja koulutusnopeuden perusteella. Käytetyt kerrokset valittiin, koska ne soveltuvat kyseessä olevien luokitteluongelmien ratkomiseen.

```
model = tf.keras.Sequential([
        tf.keras.layers.Embedding(vocab_size, 32),
        tf.keras.layers.GlobalAveragePooling1D(),

        tf.keras.layers.Dense(128, use_bias=False, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.Activation('relu'),

        tf.keras.layers.Dense(128, use_bias=False, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.Activation('relu'),

        tf.keras.layers.Dense(128, activation='relu'),

        tf.keras.layers.Dropout(0.3),
        tf.keras.layers.Dense(4, activation='softmax')
])
```

Kuvio 1. Kuvauksien neuroverkkomalli.

Kuvion 1 neuroverkossa luodaan Embedding-kerros, johon syötetään kuvauksia, joiden sanat on muunnettu numeraalisiksi arvoiksi. Embedding-kerros muuntaa numeraaliset arvot kiinteiksi vektoreiksi ja syöttää ne GlobalAveragePooling-kerrokselle.

Seuraavaksi luodaan kolme Dense-kerrosta, joissa on 128 neuronia, ja ne aktivoidaan luvussa 3.2.2 mainitulla relu-funktiolla. Kahdessa ensimmäisessä kerroksessa käytetään myös BatchNormalization-menetelmää, jonka vuoksi nämä kerrokset aktivoidaan BatchNormalization-menetelmän jälkeen.

Ennen ulostulokerrosta määritellään Dropout-kerros arvolla 0,3, joka ”unohtaa” 30 % opetetuista arvoista. Tämän tavoitteena on opettaa neuroverkkoa yleistämään opittuja piirteitä.

Ulostulokerroksessa on neljä neuronia, jotka aktivoidaan softmax-funktiolla. Neuronien aktivointifunktio määrittää todennäköisyyden kullekin neljälle luokalle.

```
model = Sequential([
    Conv2D(64, 3, padding='same', activation='relu', input_shape=(100, 100 ,3)),
    MaxPooling2D(),
    Conv2D(64, 3, padding='same', activation='relu'),
    MaxPooling2D(),
    Conv2D(64, 3, padding='same', activation='relu'),
    MaxPooling2D(),
    Flatten(),
    Dense(512, activation='relu'),
    Dense(1, activation='sigmoid')
])
```

Kuvio 2. Kuvien neuroverkkomalli.

Kuvion 2 neuroverkon konvoluutiokerroksissa on 64 neuronia, ja kerroksien neuronit aktivoidaan luvussa 3.2.2 mainitulla relu-funktiolla. Jokaisen konvoluutiokerroksen jälkeen on MaxPooling-kerros, jonka tavoitteena on vähentää laskennan monimutkaisuutta.

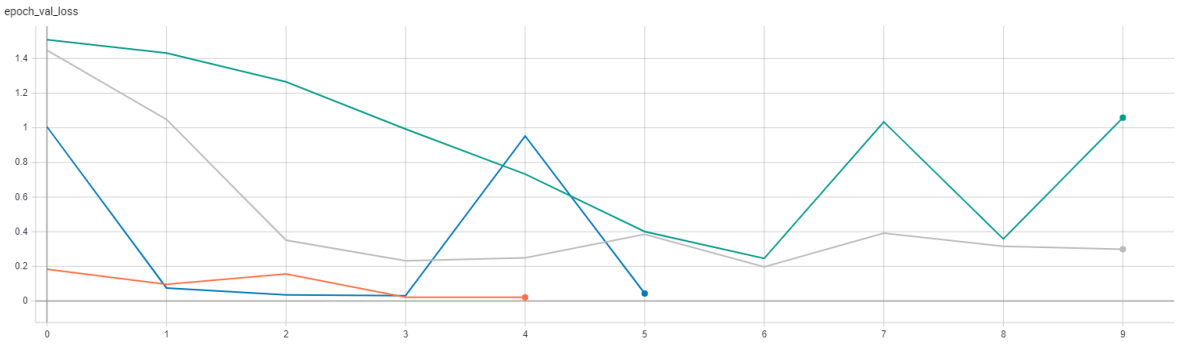
Pooling-kerroksen jälkeen Flatten-kerros tasaa syöttöarvon ja syöttää sen Dense-kerrokselle, joka sisältää 512 neuronia ja aktivoidaan relu-funktiolla. Sen jälkeen siirrytään ulostulokerrokseen, jonka neuronin aktivointi sigmoid-funktion avulla määrittelee luokkien todennäköisyydet.

4.4 Koulutuksen tulokset

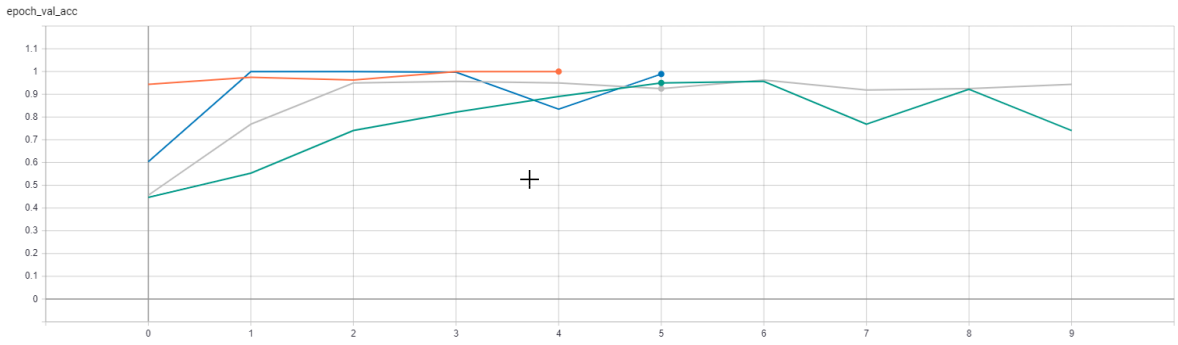
Tässä osiossa kerrotaan tarkemmin neuroverkkomallien koulutustuloksista.

4.4.1 Kuvauksien tulokset

Kuvio 3 esittelee neuroverkon tappiofunktion validointituloksia jokaisen epookin jälkeen. Kuvio 4 esittää neuroverkon validointitarkkuutta jokaisen epookin jälkeen.



Kuvio 3. Kuvauksien validoinnin tappiofunktio (x-akseli: epookki, y-akseli: tappiofunktion tulos)



Kuvio 4. Kuvauksien validoinnin tarkkuus (x-akseli: epookki, y-akseli: tarkkuus)

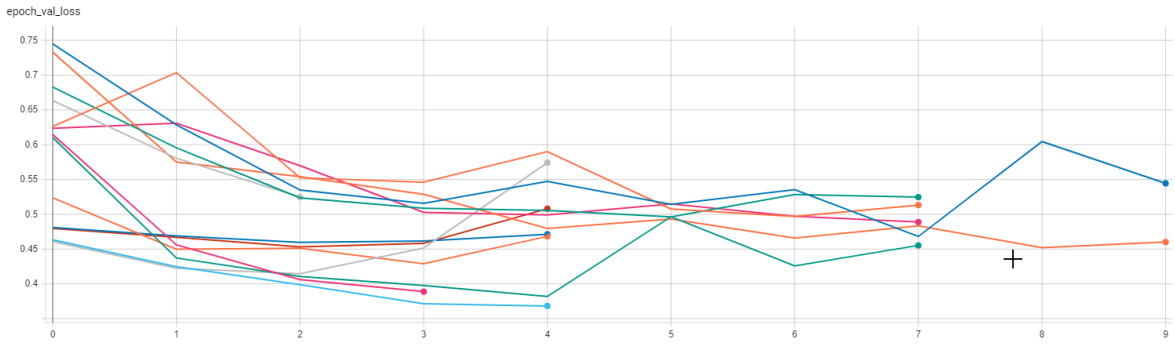
Molemmissa kuvioissa harmaa ja vihreä linja kuvaavat neuroverkkoja, jotka on opetettu datalla, jossa ei ole käytetty datan täydennystä. Sininen ja oranssi puolestaan on koulutettu täydennetyllä datalla. Neuroverkon malli ja parametrit ovat samanlaiset oranssissa ja harmaassa, sekä sinisessä ja vihreässä.

Kuviota 3 tarkastelemalla voidaan todeta, että täydennetyllä datalla saavutetaan parempi tulos tappiofunktiosta. Sininen neuroverkko ylikoulutettiin neljännellä iteraatiolla, mutta neuroverkon kerrosten parametrien säätämisen jälkeen (oranssi verkko) koulutus saavutti neljännen iteraation ilman ongelmia, ja sen tappiofunktion tulos oli silloin noin 0,04. Datalla, jota ei täydennetty, saavutettiin paras tappiofunktion tulos kuudennella iteraatiolla, mutta sekin oli vain 0,2.

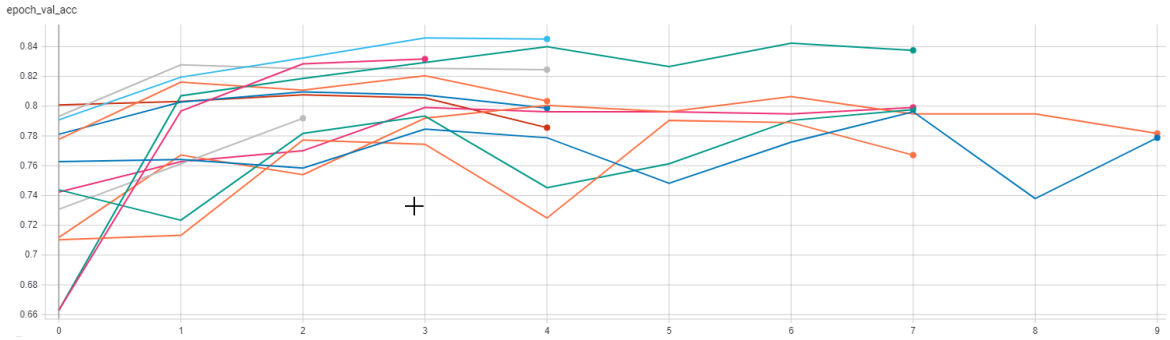
Kuvion 4 tarkkuuksissa voidaan seurata, miten tappiofunktion avulla neuroverkkoa opetetaan. Kun tappiofunktio kasvaa, luokittelun tarkkuus laskee. Tappiofunktion laskiessa tarkkuus puolestaan nousee.

4.4.2 Kuvien tulokset

Kuvia kokeiltiin alustavasti luokitella luvussa 4.1.2 mainittuihin alaluokkiin, mutta neuroverkon koulutuksessa ei onnistuttu luokittelemaan niitä tarkemmin, kuin satunnaisella valinnalla olisi pystynyt. Tämän vuoksi kuvien luokkien määrää vähennettiin, ja päädyttiin käyttämään vain kahta pääluokkaa. Kuvio 5 esittelee kuvien validoinnin tappiofunktion, ja kuvio 6 esittelee kuvien validoinnin tarkkuuden.



Kuvio 5. Kuvien validoinnin tappiofunktio (x-akseli: epookki, y-akseli: tappiofunktion tulos)



Kuvio 6. Kuvien validoinnin tarkkuus (x-akseli: epookki, y-akseli: tarkkuus)

Kuvien koulutuksessa luotiin useita neuroverkkomalleja, joissa kokeiltiin kouluttaa neuroverkkoa erilaisia parametreja, kuten kerrosten ja niiden neuroneiden määrää, säätelemällä. Näiden neuroverkkojen tarkkuustulokset on piirretty kuviossa 6. Saatujen tulosten mukaan voidaan todeta, että paras neuroverkko saatiin neljännellä epookilla, jolloin validointitarkkuus oli 0,84 eli 84 %.

Kuviosta 5 nähdään, että validoinnin tappiofunktion tulos oli alimmillaan noin 0,37, josta voidaan todeta, että neuroverkon opetuksessa on vielä parannettavaa, koska tavoitteena on saada tappiofunktion tulos mahdollisimman lähelle 0-arvoa.

4.5 Kuvausten yhteenveto

Kuvaustekstien luokittelu koneälyn avulla onnistui hyvin, koska koulutuksen validointitarkkuus oli melkein 100 % ja tappiofunktion tulos alimmillaan 0,04. Tämä tarkoittaa, että neuroverkko oppi tunnistamaan koulutettua dataa erittäin tarkasti.

Koulutuksen aikana huomattiin hankaluuksia datan vähäisyyden vuoksi. Nämä ratkaistiin datan täydennysmenetelmien avulla.

Kuvauksien luokissa oli paljon yhtäläisyyksiä, varsinkin keskijännite- ja pienjänniteluokissa. Nämä hankaloittivat luokkia erottavien piirteiden löytämistä.

4.6 Kuvien yhteenveto

Kuvien tarkkuudeksi saatiin kahden luokan kanssa vain 84 %, ja tappiofunktion tulos oli parhaimmillaan 0,37. Tämä tarkoittaa, että koulutuksessa saatiin opetettua tunnistamaan joitakin luokkiin kohdistuvia piirteitä, mutta tappiofunktion tuloksen perusteella koulutuksessa on vielä parannettavaa.

Mahdollisesti koulutusta voi hankaloittaa se, että kuvissa on erittäin paljon yhteneväisyyttä, kuten esimerkiksi puita, metsikköä ja sähkölinjoja. Tällöin luokkia erottavat piirteet eivät ole itsestään selviä, ja joissain tapauksissa luokan erottava piirre on sellainen, jota neuroverkko ei ole oppinut koulutusdatasta.

5 Johtopäätökset

Tässä luvussa käydään läpi johtopäätöksiä liittyen tehtyyn työhön, sekä pohditaan mahdollisia parannuksia. Luvussa myös kerrotaan myös, miten työn aikana luotuja neuroverkkoja tullaan käyttämään tulevaisuudessa, sekä miten voidaan käytön avulla kerätä lisää koulutusdataa.

5.1 Kuvaukset

Kuvauksien koulutusta auttoi datantäydennys, koska sen avulla saatiin lisää dataa keskijännite- ja nollavikaluokkiin. Neuroverkkoa ei saatu koulutettua luokittelemaan vikailmoituksia täydellisesti, minkä vuoksi se ei yksin voi priorisoida vikailmoituksia.

Koulutusten aikana saatiin lisää luokiteltua dataa, mikä paransi koulutustuloksia. Jos dataa saataisiin edelleen lisää, se voisi auttaa koulutusta vielä enemmän, mutta on vaikea arvioida, kuinka paljon sitä tarvittaisiin. Datan määrän kasvaessa neuroverkko oppii tunnistamaan enemmän erilaisia tapauksia.

5.2 Kuvat

Kuvien luokittelussa ei päästy yhtä hyviin tuloksiin kuin kuvaksien luokittelussa, mutta silti tulosten perusteella neuroverkko pystyi suurpiirteisesti luokittelemaan kuvia automaattisesti.

Vaikka neuroverkon tarkkuus ei ole täysin luotettava, sen avulla saadaan nopeutettua tärkeiden vikailmoitusten löytämistä.

Neuroverkon tarkkuutta voi mahdollisesti parantaa koulutusdatan puhdistaminen, jolloin datasta otettaisiin pois sellaiset kuvat, jotka ovat epäselviä tai eivät sisällä luokkaa erottavia piirteitä.

5.3 Prioriteettityökalu

Tulevaisuudessa neuroverkkojen luokittelua tullaan käyttämään prioriteettityökaluna, joka avustaa tärkeiden tai kriittisten vikailmoitusten löytämisessä. Jos neuroverkon luokittelemaa vikailmoitusta ei ole ennustettu oikein, käyttäjä voi merkitä luokittelun virheelliseksi ja antaa kyseiselle ilmoitukselle oikean luokan.

Työkalua käyttämällä saadaan myös luotua neuroverkoille lisää luokiteltua koulutusdataa, joka puolestaan auttaa kouluttamisessa.

LÄHTEET

Abadi M., Agarwal A., Barham P., Brevdo E., Chen Z., Citro C., Corrado G., Davis A., Dean J., Devin M., Ghemawat S., Goodfellow I., Harp A., Irving G., Isard M., Jia Y., Jozefowicz R., Kaiser L., Kudlur M., Levenberg J., Mané D., Monga R., Moore S., Murray D., Olah C., Schuster M., Shlens J., Steiner B., Sutskever I., Talwar K., Tucker P., Vanhoucke V., Vasudevan V., Viégas F., Vinyals O., Warden P., Wattenberg M., Wicke M., Yu Y. & Zheng X. 9.11.2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. [Verkkojulkaisu]. TensorFlow. [Viitattu 24.4.2020]. Saatavissa: https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/45166.pdf

Bhatt, S. 2018. 5 Things you need to know about reinforcement learning. [Verkkolähde]. KDnuggets. [Viitattu 20.4.2020]. Saatavissa: https://www.kdnuggets.com/2018/03/5-things-reinforcement-learning.html

Brownlee J. 6.1.2019. A Gentle introduction to the rectified linear unit (ReLU). [Verkkolähde]. Machine Learning Mastery. [Viitattu 28.4.2020]. Saatavissa: https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

Brownlee J. 17.4.2019. How do convolutional layers work in deep learning neural networks. [Verkkolähde]. Machine Learning Mastery. [Viitattu 28.4.2020]. Saatavissa: https://machinelearningmastery.com/convolutional-layers-for-deep-learning-neural-networks/

Brownlee J. 6.1.2017. Data, learning and modeling. [Verkkolähde]. Machine Learning Mastery. [Viitattu 29.4.2020]. Saatavissa: https://machinelearningmastery.com/data-learning-and-modeling/

Cambridge Dictionary. Ei päiväystä. Artificial intelligence. [Verkkolähde]. Cambridge University Press. [Viitattu 7.4.2020]. Saatavissa: https://dictionary.cambridge.org/dictionary/english/artificial-intelligence

Chollet F. 2015a. Core layers. [Verkkolähde]. Keras Documentation. [Viitattu 28.4.2020]. Saatavissa: https://keras.io/layers/core/

Chollet F. 2015b. Convolutional layers. [Verkkolähde]. Keras Documentation. [Viitattu 28.4.2020]. Saatavissa: https://keras.io/layers/convolutional/

Chollet F. 2015c. Getting started with the keras Sequential model. [Verkkolähde]. Keras Documentation. [Viitattu 28.4.2020]. Saatavissa: https://keras.io/getting-started/sequential-model-guide/

Chollet F. 2015d. Keras: The Python Deep Learning Library. [Verkkolähde]. Keras Documentation. [Viitattu 24.4.2020]. Saatavissa: https://keras.io/

Chollet F. 2015e. Normalization layers. [Verkkolähde]. Keras Documentation. [Viitattu 28.4.2020]. Saatavissa: https://keras.io/layers/normalization/

Chollet F. 2015f. Pooling layers. [Verkkolähde]. Keras Documentation. [Viitattu 28.4.2020]. Saatavissa: https://keras.io/layers/pooling/

Chollet F. 2015g. The Sequential model API. [Verkkolähde]. Keras Documentation. [Viitattu 28.4.2020]. Saatavissa: https://keras.io/models/sequential/

Khatun, A. 10.7.2018. Let's know Supervised and Unsupervised in an easy way. [Verkkolähde]. Chatbots Magazine. [Viitattu 20.4.2020]. Saatavissa: https://chatbotsmagazine.com/lets-know-supervised-and-unsupervised-in-an-easy-way-9168363e06ab

Liang J. 20.10.2018. An Introduction to Deep Learning. [Verkkolähde]. Towards data science. [Viitattu 24.4.2020]. Saatavissa: https://towardsdatascience.com/an-introduction-to-deep-learning-af63448c122c

Reaktor. Ei päiväystä. Koneoppimisen lajit. [Verkkolähde]. Elements of AI. [Viitattu 14.4.2020] Saatavissa: https://course.elementsofai.com/fi/4/1

Reaktor. Ei päiväystä. Miten neuroverkkoja rakennetaan? [Verkkolähde]. Reaktor. [Viitattu 24.4.2020]. Saatavissa: https://course.elementsofai.com/fi/5/2

Reaktor. Ei päiväystä. Miten teköäly määritellään? [Verkkolähde]. Elements of AI. [Viitattu 7.4.2020] Saatavissa: https://course.elementsofai.com/fi/1/1

Reaktor. Ei päiväystä. Muita aihepiirejä. [Verkkolähde]. Elements of AI. [Viitattu 15.4.2020] Saatavissa: https://course.elementsofai.com/fi/1/2

Reaktor. Ei päiväystä. Tekoälyn filosofiaa [Verkkolähde]. Elements of AI. [Viitattu 8.4.2020] Saatavissa: https://course.elementsofai.com/fi/1/3

Rouse, M. 2020. Artificial intelligence. [Verkkolähde]. TechTarget. [Viitattu 14.4.2020] Saatavissa: https://searchenterpriseai.techtarget.com/definition/AI-Artificial-Intelligence

Rouse, M. 2020. Machine learning. [Verkkolähde]. TechTarget. [Viitattu 14.4.2020] Saatavissa: https://searchenterpriseai.techtarget.com/definition/machine-learning-ML

Sharma A. 30.3.2017. Understanding Activation Functions in Neural Networks. [Verkkolähde]. The Theory Of Everything. [Viitattu 29.4.2020]. Saatavissa: https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

Solteq. 2019. [Verkkojulkaisu]. Vuosikertomus. [Viitattu 4.6.2020]. Saatavissa: https://cdn2.hubspot.net/hubfs/484933/Vuosikertomus_2019_FI.pdf

Stanford. 2016. The Turing Test. [Verkkolähde]. Stanford Encyclopedia of Philosophy. [Viitattu 7.4.2020] Saatavissa: https://plato.stanford.edu/entries/turing-test/#Tur195ImiGam

Stanford. 2020. The Chinese Room Argument. [Verkkolähde] Stanford Encyclopedia of Philosophy. [Viitattu 7.4.2020] Saatavissa: https://plato.stanford.edu/entries/chinese-room/

Suur-Savon Sähkö. Ei päiväystä. [Verkkolähde] Suur-Savon Sähkö -konserni. [Viitattu 4.6.2020]. Saatavissa: https://www.sssoy.fi/yrityksesta/suur-savon-sahko-konserni/

TensorFlow. 6.4.2020. Accuracy. [Verkkolähde]. TensorFlow Core v2.1.0. [Viitattu 29.4.2020]. Saatavissa: https://www.tensorflow.org/api_docs/python/tf/keras/metrics/Accuracy

Wilson, A. 29.9.2019. A Brief Introduction to Supervised Learning. [Verkkolähde]. Towards data science. [Viitattu 20.4.2020]. Saatavissa: https://towardsdatascience.com/a-brief-introduction-to-supervised-learning-54a3e3932590