Pienen datan ongelma koneoppimismallien koulutuksessa ja validoinnissa

Turun yliopisto
Tietotekniikan laitos
LuK-tutkielma
Tietojenkäsittelytiede
Maaliskuu 2025
Henri Sippola

TURUN YLIOPISTO
Tietotekniikan laitos

Henri Sippola: Pienen datan ongelma koneoppimismallien koulutuksessa ja validoinnissa

LuK-tutkielma, 26 s.
Tietojenkäsittelytiede
Maaliskuu 2025

Tekoäly on osoittautunut tärkeäksi työkaluksi tieteellisessä tutkimuksessa. Sen tehokas ja luotettava käyttö on kuitenkin riippuvainen käytetyn aineiston koosta ja laadusta. Vaikka tallennettua tietoa ja käytettäviä aineistoja on saatavilla suuria määriä, tieteellisessä tutkimuksessa työskennellään usein rajallisempien aineistojen kanssa. Tämä vaikeuttaa tekoälyn koulutusta ja hyödyntämistä. Tästä huolimatta tekoälyn käyttäminen on tärkeää pienienkin aineistojen kanssa.
Tämä tutkielma on kirjallisuuskatsaus pienien aineistojen ongelmista ja ratkaisuista koneoppimisessa. Tutkielmassa esitetään pienen datan ongelma, siihen johtavia syitä ja sen seurauksia. Lisäksi syvennytään tarkemmin aineistojen ja koneoppimismallien väliseen yhteyteen eli mitkä asiat aineistossa vaikuttavat mallien koulutukseen. Lopuksi tarkastellaan yleisimmin käytettyjä ratkaisumenetelmiä löydettyihin ongelmiin. Tutkielmassa selitetään myös lyhyesti koneoppimisen peruskäsitteitä ja toimintaa, joita tarvitaan tuloksien ymmärtämiseksi.
Tutkielman tuloksena havaittiin pienten aineistojen ylisovittuvan helposti koneoppimismallien koulutuksessa. Tämä johtuu pääosin aineistojen pienestä koosta tai suuresta määrästä aineistoa kuvaavia piirteitä. Suuri määrä piirteitä on vaikea oppia pienestä määrästä esimerkkejä. Pienet aineistot ovat ongelma lukuisilla aloilla tieteellisessä tutkimuksessa. Aineistot, ja myös ratkaisut, vaihtelevat aloittain. Suosituimpia ratkaisuja koneoppimismallien suorituksien parantamiseen pienien aineistojen kanssa ovat aineiston täydennys, siirto-oppiminen, ulottuvuuksien vähennys ja ristiinvalidointi.

Asiasanat: tekoäly, koneoppiminen, pieni aineisto, pieni data, ylisovitus

Sisällys

1 Johdanto **1**

2 Koneoppiminen **4**

2.1 Koneoppimismallien koulutus 4

2.2 Toiminta ja käyttö 5

3 Pienen datan ongelma **8**

3.1 Pienet aineistot 8

3.2 Datan keräys 10

3.3 Aineiston vaikutus koneoppimisessa 11

4 Ratkaisumenetelmät **14**

4.1 Aineiston täydennys 14

4.1.1 Kuvamanipulaatio 15

4.1.2 Synteettinen ylinäytteistys 16

4.1.3 Generatiivinen tekoäly 18

4.2 Siirto-oppiminen 20

4.3 Ulottuvuuksien vähentäminen 21

4.4 Ristiinvalidointi 22

5 Yhteenveto **24**

Lähdeluettelo **27**

1 Johdanto

Koneoppimisen käyttäminen sekä yrityksissä että akateemisessa maailmassa on yleistynyt, ja sillä on pystytty ratkaisemaan erilaisia ongelmia uusilla tavoilla ja entistä tehokkaammin [1]. Koneoppimismalli koulutetaan aineistolla, josta se voi oppia säännöllisyyksiä ja pystyä sen perusteella ennustamaan piirteitä tai luokittelemaan uutta tietoa, kuten kuvia [2].

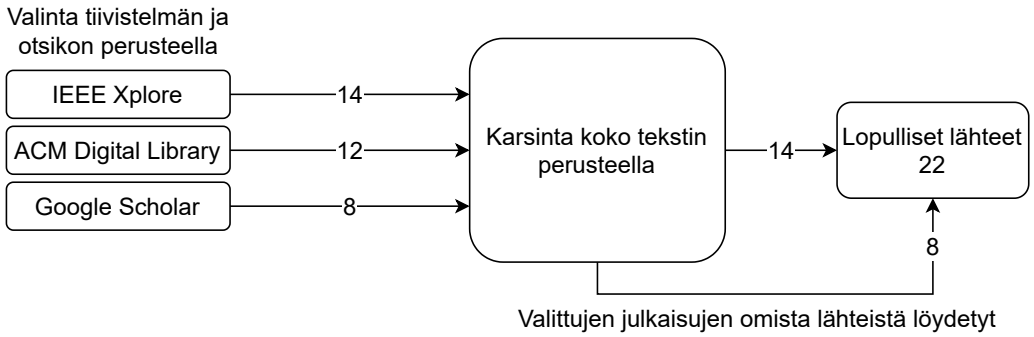
Luotettavan koneoppimismallin koulutus vaatii paljon dataa. Vaikka nykyään eletäänkin maailmassa, jossa tietoa kerätään ja sitä on saatavilla valtavia määriä lähes kaikkeen tarpeelliseen, etenkin tutkijat kohtaavat edelleen pieniä tutkimusaineistoja. Lääketieteessä, tekniikan aloilla, materiaalitieteessä, hiukkasfysiikassa ja monilla muilla aloilla työskennellään jatkuvasti uusien, harvinaisten tai muuten vaikeasti kerättävien tietojen kanssa. [3] Koneoppimisen hyödyntäminen on kuitenkin tärkeää tutkimustyössä aineistojen koosta huolimatta [4].

Tämän tutkielman tarkoitus on selvittää pieniin aineistoihin liittyviä ongelmia koneoppimismallien koulutuksessa sekä niihin käytettyjä ratkaisuja. Tutkielma pyrkii vastaamaan seuraaviin tutkimuskysymyksiin.

- **TK1**: Missä tilanteissa pieniä aineistoja kohdataan?
- **TK2**: Mitä vaikutuksia käytetyn aineiston koolla on koneoppimismallin koulutuksessa ja testauksessa?
- **TK3**: Mitkä muut tekijät datassa vaikuttavat koneoppimismallin tuloksiin?

- **TK4**: Mitä menetelmiä on pienen datan ongelmien välttämiseksi?

Tutkielma on kirjallisuuskatsaus aiheesta kirjotettuihin tieteellisiin julkaisuihin. Lähteiden hakuun käytettiin tietokantoja ACM Digital Library, IEEE Xplore Digital Library ja Google Scholar. Lisäksi lähteitä etsittiin löydettyjen julkaisujen omista lähdetiedoista. Hakulauseena oli *("machine learning" OR "deep learning" OR "artificial intelligence" OR ai) AND ("small dataset*" or "small data" or "small sample" or "limited sample")*, joka kohdistettiin julkaisun otsikkoon ja tiivistelmään. Lisäksi myöhempiä hakuja tehtiin tarkentamalla hakulausetta termeillä *"data augmentation"*, *validation*, *noise* tai *imbalance**. Lähteet valittiin alustavasti otsikon ja tiivistelmän perusteella, ja lopuksi koko tekstin perusteella. Valinnassa huomioitiin aiheen olennaisuus ja luotettavuus. Lähteiden luotettavuutta arvioitiin suoraan tekstin perusteella ja Julkaisufoorumin antamien tasoluokitusten avulla. Alustavassa valinnassa löydettiin yhteensä 34 lupaavaa lähdettä, joista valittiin lopuksi 14. Näiden neljäntoista julkaisun omista lähteistä löydettiin vielä 8 uutta lähdettä, jolloin tämän tutkielman kokonaislähdemääräksi tuli 22. Kuva 1.1 havainnollistaa tiedonhakuprosessia.



Kuva 1.1: Tiedonhakuprosessi.

Suurin osa löydetyistä lähteistä keskittyy pieneen dataan luokitteluongelmien kannalta, joten myös tämä tutkielma käsittelee pääasiassa luokittelumalleja. Tutkielma jatkuu johdannon jälkeen luvussa 2 lyhyellä selostuksella koneoppimismal-

lien toiminnasta ja koulutuksesta. Luvussa avataan tavallisimpia termejä ja käsitteitä, joita tarvitaan myöhempien lukujen ymmärtämiseksi. Luvussa 3 tarkastellaan pienen datan ongelmaa, sen esiintymistä ja syitä sekä aineiston koon ja muiden tekijöiden vaikutusta koneoppimiseen ja niiden aiheuttamia haasteita. Luvussa 4 esitellään erilaisia ratkaisumenetelmiä edellisessä luvussa löydettyihin ongelmiin. Lopuksi luvussa 5 kootaan tutkielman keskeiset löydökset tutkimuskysymyskohtaisesti.

2 Koneoppiminen

Koneoppiminen on tekoälyn osa-alue, jossa tekoälyalgoritmi oppii toistuvuuksia datasta ja pystyy niiden pohjalta ennustamaan piirteitä tai tekemään päätöksiä. Koneoppimismalli koulutetaan käyttötarpeen mukaisella aineistolla. Käyttökohteita voivat olla tieteellinen tutkimus tai erilaisten ongelmien ratkaiseminen. Suurimpia haasteita koneoppimismallien käytössä ja koulutuksessa on niiden vaativa datan määrä, sekä tulosten *selitettävyys*. [1], [2], [5]

2.1 Koneoppimismallien koulutus

Koneoppimisessa käytetty aineisto koostuu esimerkeistä, joita koulutettavalla mallilla halutaan analysoida. Esimerkit ovat alkioita, kuten kuvia tai datapisteitä, joilla on piirteitä eli niitä kuvaavia arvoja. Esimerkki voi kuvastaa vaikka yhtä potilasta, jolla on piirteinä ikä, sukupuoli, paino tai muita oleellisia tietoja. Piirteet voivat olla luokallisia tai numeerisia muuttujia. Piirteet edustavat datan ulottuvuuksia piirreavaruudessa (engl. *feature space*), jonne esimerkit sijoittuvat piirteiden arvojen mukaan. Yhden esimerkin kaikkien piirteiden arvoja yhdessä kutsutaan sen piirrevektoriksi (engl. *feature vector*). Mitä enemmän piirteitä datassa on, sitä korkeampi on datan ulotteisuus tai dimensionaalisuus (engl. *data dimensionality*). [1]

Koneoppiminen voidaan jakaa ohjattuun ja ohjaamattomaan oppimiseen. Ohjatussa oppimisessa malli koulutetaan merkityn koulutusaineiston perusteella luokittelemaan uutta dataa ennalta määrättyihin ryhmiin eli luokkiin. Ohjattu oppiminen

voi tuottaa joko kategorisia tai numeerisia vastauksia. Ohjaamattomassa oppimisessa mallin tulee löytää datasta tuntemattomia yhteyksiä tai luokkia ilman tietoa niiden olemassaolosta. Näiden lisäksi sitä voidaan käyttää ulottuvuuksien vähentämiseen (engl. *dimensionality reduction*), jossa koneoppimismalli tiivistää aineiston piirteet muutamaan tärkeimpään ja karsii turhat piirteet pois. Datan ulotteisuus käsitellään tarkemmin luvussa 3.3, ja ulottuvuuksien vähentäminen luvussa 4.3. Sekä ohjaamaton että ohjattu oppiminen voi vaatia paljon dataa, jotta malli pystyy löytämään todellisia yhteyksiä piirteistä. [1]

Koneoppimismallin kehitys alkaa koulutusaineistosta. Aineisto kerätään ja puhdistetaan eli siitä poistetaan poikkeavat tai puutteelliset esimerkit [4]. Tämän jälkeen aineisto jaetaan satunnaisesti kolmeen osaan: koulutus-, validointi- ja testausaineistoon [1], [2]. Koulutus aloitetaan usein piirteiden valitsemisesta. Jos piirteitä on paljon, on malli vaikea kouluttaa tunnistamaan säännöllisyyksiä näiden kaikkien välillä. Piirteiden valinnalla näistä valitaan koneoppimismallin kannalta oleellisimmat. Tämä voidaan tehdä joko manuaalisesti tai antaa mallin löytää ne itse. [4] Koulutuksen aikana malli oppii koulutusaineistosta toistuvuuksia ja yhteyksiä, joiden perusteella se voi ennustaa tai luokitella uutta dataa [2]. Koulutettu malli ja piirteiden valinta validoidaan aikaisemmin jaetulla validointiaineistolla. Validoinnin avulla valitaan parhaan tuloksen tuottavat piirteet ja mallin hyperparametrit, kuten *neuroverkon* kerroksien ja neuronien määrä [6]. Lopuksi testiaineistoa käytetään mallin tarkkuuden arvioimiseksi. [1], [2]

2.2 Toiminta ja käyttö

Tieteellisistä julkaisuista tehdyn bibliometrisen tutkimuksen [3] perusteella pienien aineistojen kanssa käytetyimpiä koneoppimismenetelmiä ovat muun muassa keinotekoiset neuroverkot (engl. *artificial neural network*), tukivektorikone ja päätöspuut.

Neuroverkko. Neuroverkko on ihmisen aivojen toimintaa jäljittelevä malli. Se koostuu joukosta toisiinsa kytkettyjä simuloituja neuroneita, jotka muodostavat kerroksia ja voivat olla yhdistettyinä toisiin kerroksiin [2]. Neuroneilla on yksi tai useampi syöte, painokertoimet, aktivaatiofunktio ja yksi tai useampi ulostulo. Jokaiselle neuronien väliselle linkille voidaan määrittää erilaiset painoarvot tai parametrit, ja jokaisen neuronin kohdalla on oma aktivaatiofunktio, johon painotettua syötettä sovelletaan. Tämän perusteella neuroni laskee, minkä syötteen se vie eteenpäin ulostulolinkkejä pitkin. [1]

Tukivektorikone. Tukivektorikone on koneoppimismenetelmä, jota käytetään aineiston luokitteluun. Tukivektorikone esittää aineiston pisteinä avaruudessa niiden piirteiden perusteella [7], ja pyrkii jakamaan aineiston luokkiin maksimoiden niiden välisen marginaalin [8]. Mallin tavoitteena on löytää optimaalinen rajaus luokkien välillä käyttäen tukivektoreita, jotka mittaavat marginaalitasojen etäisyyttä. Tukivektorikone on lineaarinen luokittelumalli, mutta piirreavaruus voidaan kuvata ydinfunktiolla, jolloin malli voi tuottaa epälineaarisia ratkaisuja. [1], [7]

Päätöspuu. Päätöspuu on luokittelumalli, joka toimii jakamalla tietoa erillisiin haaroihin yksinkertaisten ehtojen perusteella siten, että jokainen solmu sisältää esimerkit yhdestä luokasta. Luokittelu aloitetaan juurisolmusta, ja etenee jakautuen haaroihin tai oksiin. [1], [7]

Tekoälyä ja koneoppimista hyödynnetään nykyään monella alalla, ja käyttö on lisääntynyt etenkin viime vuosien aikana. Vaikka idea oppivista neuroverkoista onkin peräisin jo 1950-luvulta, on sen käyttö ja kehitys kasvanut suuresti vasta datamäärien ja tietokoneiden laskentatehon kasvaessa. Koneoppimismalleja käytetään etenkin akateemisessa työssä ja tutkimuksessa. Teknologian kehittyessä tekoälyratkaisuja voidaan käyttää yhä edistyneempiin ja abstraktimpiin ongelmiin, mutta lisääntyvä monimutkaisuus tuo mukanaan ongelman mallin selitettävyydessä. [1] Selitettä-

vyydellä tarkoitetaan kykyä perustella mallin toimintaa ja sen antamia vastauksia. Koneoppimismallit eivät tyypillisesti pysty ilmaisemaan epävarmuutta, joten niiden tarkkuudesta ei voida olla koskaan täysin varmoja. Selitettävyyden ongelmaa kutsutaan mustan laatikon ongelmaksi (engl. *the black box problem*), joka voi estää koneoppimisen hyödyntämisen monissa tilanteissa. [1], [5]

3 Pienen datan ongelma

Aineistojen rajallisuutta, pientä kokoa ja niiden ominaisuuksien aiheuttamia haasteita tekoälyn kehityksessä kutsutaan pienen datan ongelmaksi (engl. *small data problem*), mikä voidaan ajatella massadatan ja sen luomien mahdollisuuksien vastakohtana. Termillä "pieni data" on useita eri merkityksiä, mutta tässä tutkielmassa se on määritelty tekoälyn näkökulmasta datan rajallisuutena. Massadatalla viitataan datan suureen määrään, sen säilytyksen, jakamisen ja keräämisen nopeutuneeseen kasvuun. Tutkimustyössä eri aloilla ei kuitenkaan aina ole tarjolla tilanteeseen sopivia suuria datamääriä tai kehittyneitä datankeräystekniikoita [4], jolloin voidaan joutua käyttämään rajallisia ja liian pieniä aineistoja [3], [9], tai tutkimusdata joudutaan keräämään itse [4]. Joissain tilanteissa dataa voi myös olla määrällisesti paljon, mutta se on niin huonolaatuista, korkeaulotteista tai epätasaista, että se ei ole riittävä. Dataan voi myös liittyä eettisiä tai yksityisyyteen liittyviä ongelmia, jotka hankaloittavat sen käyttöä ja saatavuutta. [2], [10] Aineiston koko ja laatu ovat yhteydessä koneoppimismallin taipumukseen *ylisovittaa* data. [4]

3.1 Pienet aineistot

Pienen datan ongelma ilmenee koneoppimisessa rajallisten aineistojen käyttönä. Tarvittava aineiston koko riippuu kuitenkin sen käyttökohteesta ja ominaisuuksista. Dataa tarvitaan enemmän, jos se on laadultaan *kompleksista*. Myös *epätasainen* data voi vaatia aineiston kasvattamista sen tasaamiseksi. Todelliset syyt pienen datan

ongelmalle voivatkin olla datan kompleksisuus ja epätasaisuus, jotka näyttäytyvät aineiston koon riittämättömyytenä. [2]–[4], [10]

Kompleksisuus kuvaa datan monimutkaisuutta ja epäselvyyttä. Luokkien jakautumisen ja niiden välisten yhteyksien vaikeaselkoisuus sekä datan korkea ulotteisuus lisäävät datan kompleksisuutta, jolloin koneoppimismallin on vaikeampi oppia aineisto toivotusti. Kompleksisuutta lisää myös datan sisältämä kohina. Kohina on datassa esiintyvää näennäistä satunnaisuutta, joka ei pohjaudu tutkittavaan ilmiöön. Se voi olla aineiston luokkien tai piirteiden sisäistä vaihtelua, jota ei pystytä selittämään muiden piirteiden avulla, eikä siitä voida tehdä johtopäätöksiä. [3], [11]

Epätasaisuus aineistossa tarkoittaa, että siinä esiintyvät piirteet tai luokat eivät ole tasaisesti edustettuna [3]. Koneoppimismallin tarkoitus on etsiä datasta yhteyksiä tai luokkia, mutta se olettaa datan olevan täydellistä ja tasaista. Oikean maailman tilanteista kerätyssä datassa voi olla harvinaisempia luokkia tai piirteiden arvoja, jotka ovat siten aineistossa aliedustettuina. [10] Pienemmissä aineistoissa aineiston tasaisuuden merkitys kasvaa, sillä harvinaiset luokat voivat jäädä kokonaan edustamatta tai niistä on vain muutama esimerkki [3]. Myös satunnaisuus korostuu pienissä aineistoissa ja se voi saada tavallisesti esiintyvän luokan ali- tai yliedustetuksi [4].

Pienen datan ongelma on hyvin monialainen. Sitä kohdataan biologian aloilla [3], [4], materiaalitieteessä, insinöörialoilla, hiukkasfysiikassa, taloustieteessä [3] ja terveydenhuollossa/lääketieteessä [2]–[5], [9], [10]. Pieniä tai epätasaisia aineistoja joudutaan käyttämään muun muassa erilaisten sairausten diagnosoinnissa [2]–[5], [9], [10], pandemian tutkimuksessa, kasvojentunnistuksessa, sosiaalisen median analyysissa [3], liikkeen- ja katseenseurannassa sekä genomien tutkimuksessa [4].

Esimerkiksi genomien tutkimuksessa aineistot ovat keskimäärin korkeaulotteisia genomien sisältäessä paljon piirteitä, ja vaikka aineiston koko voi olla joissain tilanteissa suurikin, se ei ole riittävä. [3], [4] Koneoppimista käyttämällä genomien avulla

voidaan esimerkiksi ennustaa syövän etenemistä, mutta pienet aineistot rajoittavat mallien tarkkuutta. [12] Sama ongelma esiintyy hiukkasfysiikassa [13] ja materiaalitieteessä [14]; data ja tutkittavat ominaisuudet ovat luonteeltaan komplekseja, eikä aineisto ole riittävä ja sen hankinta on ongelmallista.

3.2 Datan keräys

Pienet aineistot johtuvat usein datan keräämiseen liittyvistä vaikeuksista. Kerääminen voi olla hidasta, kallista tai data voi olla harvinaista. Tämän lisäksi erilaiset eettiset ja yksityisyyteen liittyvät kysymykset, etenkin ihmisiltä kerätyissä aineistoissa, voivat haitata datan käyttämistä tutkimuksissa ja koneoppimismallien kehityksessä. [2], [4], [5], [9] Keräämisen lisäksi myös laitteistovaatimukset voivat rajoittaa datamäärien käyttöä tai tallennusta [3].

Lääketieteessä data voi olla esimerkiksi lääketieteellisellä kuvantamisella tuotettujen kuvien muodossa, joiden kerääminen ja luokittelu vaatii manuaalista työtä erikoistuneelta lääkäriltä. Tämä tekee aineiston keräämisestä sekä hidasta että kallista. [9] Keräämistä voi lisäksi vaikeuttaa sairauksien harvinaisuus [9] ja esimerkiksi neurokuvantamistutkimuksissa datan korkea ulotteisuus, joka luo tarpeen entistä isommalle aineistolle. [4] Ihmispotilailta kerätyissä kuvissa tiedon käytön oikeudet ja yksityisyys voivat nousta ongelmaksi. Diagnooseja ei myöskään tehdä vain kuvien perusteella vaan kuvien lisäksi koneoppimismallit tarvitsevat myös muuta dataa potilaalta. Syntymäaika, sukupuoli, pituus ja sairaushistoria voivat auttaa diagnoosin muodostamisessa, mutta lisäävät tietojen yksityisyyttä. [1], [2], [5]

Datan puutteessa joillain aloilla aineistoja voidaan luoda simuloinnilla, jos data pohjautuu johonkin laskennallisesti mallinnettavaan ilmiöön. Datan simulointi voi kuitenkin olla laitteiston puolesta todella vaativaa ja aikaa vievää. Lisäksi lopputulos on vain likiarvo todellisesta datasta, eikä välttämättä sisällä kaikkea oleellista tietoa, jota koneoppimismalli pystyisi hyödyntämään. [13]

Aineiston riittämättömyys ei aina johdu datan keräämisen ongelmista. Dataa voi olla saatavilla paljon, mutta vaadittu määrä on niin suuri, että sen tallennus ja käyttö koneoppimismallin koulutuksessa osoittautuu vaikeaksi. Joskus käytettävissä oleva laitteisto voi olla hyvinkin rajallinen, ja mielessä on pidettävä käytössä oleva tallennustila ja suorituskyky. [3] Erityisesti syväoppivat mallit, joita sovelletaan komplekseihin aineistoihin, vaativat suuria datamääriä ja nopeaa laskentatehoa [14].

Jo haasteelliseksi havaittua datan keräystä hankaloittaa lisäksi kutakin datalähdettä koskevat rajoitukset, luvat ja käyttöoikeudet. Suurin osa isoista kuva-aineistoista koostuu internetistä kerätyistä kuvista, joiden käyttö vaatii luvan tai sen käyttö kaupallisesti on kiellettyä. Tämä rajoittaa saatavilla olevan datan määrää ja myös näillä aineistoilla esikoulutettujen mallien hyödyntämistä *siirto-oppimisessa*. [15] Siirto-oppimista tarkastellaan luvussa 4.2.

3.3 Aineiston vaikutus koneoppimisessa

Aineiston koko on suoraan yhteydessä koneoppimismallin tarkkuuteen ja kykyyn oppia ja yleistää opittuja säännöllisyyksiä [2], [3]. Jos aineisto ei ole riittävä, malli voi ali- tai ylisovittaa datan. [4] Epätasaiset aineistot näkyvät mallin tuloksissa sen vaikeuksina tunnistaa aliedustettuja luokkia. Koneoppimismallit pyrkivät maksimoimaan kokonaistarkkuuden, joten ne suosivat suurimpia luokkia, mikä johtaa epärealistisiin tuloksiin. [10] Mallin tarkkuutta ja tasaisuutta voidaan kuitenkin hallita validoinnilla ja testauksella [16].

Korkeaulotteinen eli kompleksinen data vaatii keskimäärin suuremman koulutusaineiston. Aineiston piirre-esimerkki suhde, eli kutakin esimerkkiä kohden olevien piirteiden määrä suhteessa esimerkkien määrään, on hyvä tapa ennustaa herkkyyttä ylisovitukseen. Jos suhde on korkea, mallilla on taipumus ylisovittaa data. Lisäksi monet piirteistä voivat olla tutkittavan ilmiön kannalta merkityksettömiä, jolloin malli sovittaa näistä ylimääräistä kohinaa. [4] Alisovittaessa koneoppimis-

malli ei kykene oppimaan datan taustalla olevia säännöllisyyksiä. Alisovitus johtuu liian yksinkertaisesta mallista ja on korjattavissa kasvattamalla mallin kompleksisuutta. Mallin kompleksisuuden kasvaessa ylisovituksen todennäköisyys nousee [5]. Ylisovituksessa malli ylioppii aineiston eli sen sijaan, että se oppisi datassa olevia yleisiä säännöllisyyksiä tai yhteyksiä, se oppii koulutusaineiston täysin, mukaan lukien siinä olevan kohinan, eikä pysty yleistämään oppimaansa uuteen dataan, sillä uusi data sekä siinä oleva kohina ei vastaa koulutuksessa käytettyä dataa. [4]

Kompleksisemmat syväoppivat mallit suoriutuvat paremmin suurempien aineistojen kanssa ja pystyvät löytämään monimutkaisempia yhteyksiä datasta, mutta herkästi ylisovittavat pienen aineiston. Pienemmissä aineistoissa yksinkertaisemmat mallit suoriutuvat paremmin kuin monimutkaiset mallit, mutta liian yksinkertainen malli alisovittaa datan. [5] Koneoppimismallin koulutuksessa on tavoitteena löytää näiden välistä tila, jossa malli sovittaa datan tarpeeksi hyvin oppien oleelliset ominaisuudet karsien pois ylimääräisen kohinan ja pystyy yleistämään ennennäkemättömään dataan. Aineiston pieni koko ja suuri piirre-esimerkki suhde vaikeuttaa tätä ja voi tehdä tarkan mallin koulutuksesta haastavaa. [4]

Sovituksen lisäksi mallin tuloksiin ja käytettävyyteen vaikuttaa aineiston laatu ja edustavuus. Aineiston epätasaisuus ja kohina väärentävät tuloksia ja vaikeuttavat myös mallin sovitusta. [2] Epätasainen aineisto saattaa johtaa mallin korkeaan tarkkuuteen mallin ollessa käytännössä hyödytön. Esimerkiksi lääketieteellisiä diagnooseja tunnistava malli saattaa saada korkean prosentuaalisen tarkkuuden luokittelemalla kaikki esimerkit negatiivisiksi, jos positiiviset diagnoosit ovat harvinaisempia kuin negatiiviset eli terveet potilaat. Tällöin malli on korkeasta tarkkuudestaan huolimatta käyttökelvoton. [10]

Koulutuksen onnistuminen riippuu menestyksekkäästä validoinnista ja lopullisesta testauksesta. Validoinnilla voidaan hallita ylisovituksen määrää ja varmistaa onnistunut koulutus ja sopiva mallin kompleksisuus. Kuten luvussa 2.1 mainittiin,

validoinnissa koulutetun mallin tarkkuutta arvioidaan ja optimoidaan valitsemalla parhaan tuloksen tuottavat hyperparametrit ja lopullisen mallin tarkkuus arvioidaan testiaineistolla. Tähän käytetään tavallisesti erillisiä validointi- ja testiaineistoja vääristymisen välttämiseksi, mutta pienien aineistojen tapauksessa koulutusaineiston määrää pyritään maksimoimaan, eikä erillinen validointi- tai testiaineisto tule kyseeseen. Tällöin voidaan käyttää *ristiinvalidointia*. [4], [8], [16] Ristiinvalidointi esitellään luvussa 4.4.

4 Ratkaisumenetelmät

Koneoppimismallin tarkkuutta voidaan parantaa kasvattamalla aineiston kokoa, helpottamalla oppimista ulottuvuuksien vähentämisellä, hyödyntämällä siirto-oppimista mallin esikouluttamiseksi ja käyttämällä ristiinvalidointia, jotta erillisiä validointi- ja testiaineistoja ei tarvitse vähentää koulutusaineistosta. Kullakin menetelmällä on omat vahvuudet, heikkoudet ja tilanteet joissa niitä on sopiva käyttää. Niitä voidaan myös yhdistellä optimaalisen tarkkuuden saavuttamiseksi. [1]–[3]

4.1 Aineiston täydennys

Aineiston täydennys eli datan augmentointi (engl. *data augmentation*) on yksi useimmin käytetyistä menetelmistä parantaa koneoppimismallin tuloksia [3]. Se tarkoittaa aineiston koon kasvattamista joko manuaalisesti, laskennallisesti tai käyttämällä generoivaa tekoälyä. Aineiston täydennys on erityisen suosittu tekniikka tilanteissa, joissa aineistossa on epätasaisuutta luokkien välillä. Tällöin puhutaan ylinäytteistyksestä (engl. *oversampling*) eli tietyn luokan kasvattamisesta aineiston tasaamiseksi. [2], [10] Sen toiminta perustuu aineiston tasaisuuden korjaamiseen sekä aineiston koon ja mallin tarkkuuden suhteeseen; aineiston kokoa kasvattamalla mallin tarkkuuden tulisi myös kasvaa. Tämän onnistuminen on kuitenkin kiinni lisätyn aineiston laadusta, joka riippuu käytetystä tekniikasta. [17]

Ensisijainen tapa kasvattaa aineistoa on uusien esimerkkien kerääminen, mutta käsiteltäessä pieniä aineistoja tämä ei usein tule kyseeseen datan rajallisuuden vuok-

si. Aineiston täydennyksellä tarkoitetaan aineiston kasvattamista keinotekoisilla esimerkeillä joita voidaan luoda laajennettavan aineiston pohjalta. [2] Tästä käsitellään kahta erilaista lähestymistapaa: kuvamanipulaatio ja synteettiset ylinäytteistysalgoritmit. Näytteistysalgoritmit voidaan edelleen jakaa yksinkertaisempiin laskennallisiin algoritmeihin kuten *SMOTE* ja *ADASYN*, ja tekoälypohjaisiin algoritmeihin kuten *DetectorGAN* ja *DeepSMOTE*. [2], [9], [10].

4.1.1 Kuvamanipulaatio

Kuvantunnistukseen koulutettavien koneoppimismallien koulutuksessa käytetään monesti kuvamanipulaatiota koulutusaineiston kasvattamiseksi. Kuvamanipulaatiossa olemassaolevia esimerkkejä muokataan, jolloin niistä tulee uusia hieman erilaisia esimerkkejä, jotka lisätään vanhojen esimerkkien ohella aineistoon. [2] Kuvaa voidaan muokata esimerkiksi kääntämällä, rajaamalla, ottammalla sen peilikuva tai muuttamalla jotain sen ominaisuuksia kuten kontrastia, värejä, kohinaa tai valotusta. Kuvaan voidaan myös asetta erilaisia filttereitä, jotka voivat taivuttaa tai venyttää kuvaa. [2] Manipulaatio voidaan tehdä manuaalisesti tai automaattisesti jollain algoritmilla [17].

Kuvamanipulaatio on hyvä ratkaisu tilanteissa, missä kuvien edellämainituilla ominaisuuksilla ei ole tarkasti määriteltyjä edellytyksiä. Tämä siis ei välttämättä toteudu jos koneoppimismallia koulutetaan esimerkiksi lääketieteellisten kuvien analysointiin. Lääketieteellisellä kuvantamisella luodut esimerkit ovat kuvia ihmiskehosta, joissa on tietty rajaus, kontrasti, kulma ja värit. Kuvamanipulaation soveltuvuutta tulee siis arvioida tilannekohtaisesti. [2]

Uuden aineiston tulee säilyttää koneoppimismallin kannalta tärkeitä tunnistukseen tarvittavia osia kuvista, mutta sisältää erilaisuuksia, jotka auttavat mallin tarkkuutta ja kykyä yleistää tietoa. Kuviin pyritään lisäämään ylimääräistä ja merkityksetöntä tietoa, mikä auttaa mallia tunnistamaan kuvista oleelliset asiat ja sivuut-

tamaan tarpeettoman kohinan ja satunnaisuudet. [18] Kuvamanipulaatiota voidaan käyttää myös aineiston tasaukseen kohdistamalla se aliedustettuun luokkaan, kunnes aineisto on tasainen [2].

4.1.2 Synteettinen ylinäytteistys

Synteettinen ylinäytteistys on laskennalliseen tai tekoälypohjaiseen generointiin pohjautuvaa aineiston täydennystä. Laskennallisten tekniikoiden tarkoitus on pääasiassa korjata aineiston luokkien välistä epätasaisuutta ja sitä kautta parantaa koneoppimismallin tarkkuutta. [10] Ylinätteistysalgoritmeja on lukuisia. Tässä alaluvussa näistä esitellään satunnainen ylinäytteistys (engl. *random oversampling*), SMOTE (engl. *Synthetic Minority Oversampling Technique*) ja ADASYN (engl. *Adaptive Synthetic sampling*), ja seuraavassa alaluvussa generatiiviseen tekoälyyn perustuvat menetelmät.

Satunnainen ylinäytteistys on näistä vanhin ja toiminnaltaan yksinkertaisin. Aliedustetun luokan esimerkkejä monistetaan satunnaisesti ja lisätään uudelleen aineistoon luokkien tasaamiseksi. Satunnaisella ylinäytteistyksellä on taipumus ylisovittaa aliedustettu luokka, johon sitä sovelletaan, sillä se ei luo mitään uutta tietoa vaan vahvistaa vanhaa. Se on helppokäyttöinen eikä vaadi mitään ennakkotietämystä, mutta se on tehokkuudeltaan heikko, ja sille on parempia vastineita. [10] Se antaa kuitenkin perusidean ylinäytteistyksestä.

SMOTE on Chawlan ym. [19] kehittämä ylinäytteistysmenetelmä. Se käyttää hyväksi aineiston piirreavaruusesitystä, josta se laskee k-lähimmän naapurin menetelmää (engl. *k-nearest neighbor algorithm*, KNN) käyttäen uusia datapisteitä aliedustettuun luokkaan. [19] Se on vakiintunut ja suosittu, ja siitä on useita variantteja kuten myöhemmin käsiteltävä ADASYN [10]. SMOTE on satunnaista ylinäytteistystä älykkäämpi menetelmä ja korjaa epätasaisuuden luomalla uusia näytteitä vanhojen kopioimisen sijaan. Tästä huolimatta se ei varsinaisesti tuo muuta kuin satunnaisuu-

teen perustuvaa uutta tietoa aineistoon, sillä luodut esimerkit pohjautuvat täysin aikaisempiin.

SMOTE valitsee satunnaisesti aliedustetusta luokasta esimerkin ja soveltaa siihen KNN-algoritmia, eli se etsii aineiston piirreavaruudesta k kappaletta lähimpiä esimerkkejä samasta luokasta. Valituista esimerkeistä eli naapureista valitaan vielä satunnaisesti jokin ennalta päätetty määrä esimerkkejä, jolloin osa karsiutuu pois, eivätkä valikoituneet naapurit ole välttämättä näistä lähimmät. Valikoituneet naapurit ja alkuperäinen datapiste toimivat pohjana uudelle esimerkille. Uusi esimerkki luodaan satunnaiseen kohtaan alkuperäisen datapisteen ja valikoituneen naapurin väliselle suoran osalle. Uusi esimerkki saa siis piirrearvonsa piirreavaruuden sijaintinsa perusteella. Tämä toistetaan kaikille valikoitujen naapurien ja alkuperäisen datapisteen väleille. Tämän jälkeen valitaan uusi satunnainen datapiste, jolle algoritmi toistetaan. [19]

SMOTEa voi mukauttaa tarvittaessa aineiston luonteen ja epätasaisuuden määrän perusteella. Siitä voidaan muuttaa naapurien määrää k sekä tästä valittavien lopullisten naapurien määrää ja montako kertaa algoritmi toistetaan, eli kuinka monelle esimerkille etsitään naapureita. [19]

ADASYN on SMOTE-variaatio joka ottaa huomioon aineiston jakautumisen piirreavaruudessa ja laskee jokaiselle aliedustetun luokan esimerkille painoarvon, jonka perusteella synteettisiä esimerkkejä luodaan. Tarkoituksena on tunnistaa aineistosta vaikeasti opittavat esimerkit ja vahvistaa niitä luomalla uusia esimerkkejä niiden perusteella. Painoarvo lasketaan lähellä olevien enemmistöluokan esimerkkien määrästä. [10]

Vertailussa [10] Covid-19 aineiston luokitteluun koulutettu koneoppimismalli suoriutui ensimmäisessä aineistossa parhaiten ADASYN tekniikalla tehdyllä ylinäytteistyksellä verrattuna tavalliseen SMOTE-ylinäytteistykseen, satunnaiseen ylinäytteistykseen ja täydentämättömään aineistoon. Toisessa aineistossa tavallinen SMOTE

suoriutui parhaiten. Nämä kaikki suoriutuivat kuitenkin huonommin kuin syväoppinut generatiivinen näytteistysmalli, joka esitellään seuraavaksi.

4.1.3 Generatiivinen tekoäly

Aiemmin esitellyt aineiston täydennyksen tekniikat ovat olleet "ei-älykkäitä" eli niissä uutta aineistoa on luotu joko manuaalisesti tai yksinkertaisen algoritmin avulla. Täydennykseen voidaan myös käyttää syväoppivaa generoivaa tekoälyä, joka koulutetaan tarvittavan aineiston luomiseen. Tästä on useita eri tekniikoita, joista suurin osa pohjautuu generatiiviseen kilpailevaan verkostoon (engl. *generative adversarial network*, GAN) [2], [9], [17], [18]. Sen sijaan esimerkiksi DeepSMOTE on myös syväoppimista hyödyntävä täydennysstrategia, joka ei käytä GAN-arkkitehtuuria [10].

GAN on Goodfellowin ym. [20] kehittämä neuroverkkoarkkitehtuuri, jossa generatiivinen malli eli generaattori (engl. *generator*) asetetaan kilpailemaan luokitinmallia eli diskriminaattoria tai erotinta (engl. *discriminator*) vastaan. Generaattori yrittää tuottaa uusia esimerkkejä alkuperäisen aineiston pohjalta ja diskriminaattori yrittää erottaa tuotettuja esimerkkejä alkuperäisistä. Kilpailu mallien välillä saa kummatkin mallit kehittymään kunnes diskriminaattori ei pysty enää erottelemaan generaattorin tuottamia esimerkkejä alkuperäisistä. [20]

Käytetyn aineiston vaihtelevuus ja koko vaikuttavat GAN-mallin tuottamien esimerkkien laatuun [17]. Lisäksi, tyypillinen kuvien luontiin koulutettu GAN-malli on suunniteltu luomaan ihmisen kannalta realistisia kuvia [17], eikä se siis takaa niiden auttavan koneoppimismallin koulutusta. Sellaisenaan käytetty GAN-malli ei ole riittävä parantamaan toisen mallin tarkkuutta erityisesti mallien välisen palautteen puutteen vuoksi. Eli aineistoa generoiva malli ei siis tiedä auttaako generoitu aineisto koulutettavan mallin tarkkuutta, eikä mukaudu sen tarpeisiin. [9]

DetectorGAN on Liun ym. [9] kehittämä GAN-pohjainen koulutusmalli, joka on suunniteltu aineiston täydennykseen yhdistämällä koulutettava luokitinmalli eli de-

tektori (engl. *detector*) GAN-algoritmin kanssa. Samaan tyyliin kuin GAN-mallissa, detektori antaa palautetta luoduista kuvista sen perusteella, parantavatko ne detektorin tarkkuutta. Mallin toiminta on hieman epäintuitiivinen, sillä generaattori pyrkii maksimoimaan tuotettujen kuvien kohdalla detektorin virheellisen tunnistuksen. Tämä perustuu siihen, että vaikka generaattori ja diskriminaattori pyrkivät tuotettujen kuvien realistisuuteen, detektorin tarkoitus on tunnistaa oikein todellisia kuvia eikä generoituja kuvia. Generoitujen kuvien tunnistuksen maksimointi voi olla jopa haitallista. DetectorGAN-mallin todettiin parantavan keskimääräistä tunnistustarkkuutta 20 prosenttia ylittäen sen hetken näytteistystekniikoiden huipputason. [9]

Esimerkkinä syväoppivasta näytteistystekniikasta, joka ei ole GAN-pohjainen, mainittiin DeepSMOTE. Se on Dablainin ym. [21] kehittämä vuonna 2023 julkaistu epätasaiseen dataan tarkoitettu näytteistysmalli. Se koostuu koodaaja/dekoodaaja (engl. *encoder/decoder*) rakenteesta, tähän sovellettavasta parannellusta tappiofunktiosta (engl. *loss function*) ja SMOTE-ylinäytteistyksestä. Tappiofunktiolla mitataan mallin tuottaman datan samanlaisuutta lähdeaineistoon. Paranneltuun tappiofunktioon kuuluu lisäksi *rangaistustermi* (engl. *penalty term*), jolla lisätään varianssia prosessiin vertaamalla tuotettua dataa lähdeaineistosta satunnaisesti valittuun luokkaan. Koodaaja ja dekoodaaja koulutetaan aineistolla ottamaan vastaan puhdasta dataa ja tiivistämään se matalaulotteiseksi piirrevektoriesitykseksi ja rakentamaan se takaisin alkuperäisen muodon mukaiseksi. SMOTE-algoritmilla luodaan uusia esimerkkejä tiivistettyyn esitykseen. Dekoodaaja rakentaa SMOTE-generoiduista esimerkeistä uusia realistisia ja tunnistettavia esimerkkejä, joita voidaan käyttää uuden koneoppimismallin koulutuksessa. [21]

DeepSMOTEn vahvuudet GAN-pohjaisiin menetelmiin verrattuna ovat, että se ei vaadi erikseen koulutettavaa diskriminaattoria, se vaatii keskimäärin vähemmän dataa eikä se kärsi GAN-malleille tyypillisestä tilan romahduksesta, jossa generaattori tuottaa liian samanlaisia esimerkkejä [21]. DeepSMOTE ylitti sen julkaisussa

[21] tehdyssä vertailussa usean GAN-pohjaisen algoritmin suorituskyvyn, mutta perusteellista vertausta DeepSMOTEn ja DetectorGANin välillä ei ole tehty.

4.2 Siirto-oppiminen

Siirto-oppiminen on aineiston täydennyksen ohella käytetyimpiä menetelmiä koneoppimismallin tulosten parantamiseen pienien aineistojen yhteydessä [2]. Siirto-oppimisessa käytetään erillistä isoa aineistoa esikouluttamaan malli, joka erikoistetaan pienellä aineistolla tehtäväkohtaisesti. Malli oppii esikoulutuksessa yksinkertaisia matalan tason piirteitä, jolloin koulutusta ei tarvitse tehdä ainoastaan pienellä aineistolla. [14]

On toivottavaa, että esikoulutukseen käytetty aineisto on mahdollisimman samankaltainen kuin myöhemmin käytettävä pieni aineisto tuloksien maksimoimiseksi [2]. Tämä on kuitenkin tilannekohtaista, ja esimerkiksi luonnollisten kuvien tunnistuksessa mallit jakavat kaikille yhteisiä opittuja piirteitä kuten yksinkertaisten muotojen, käyrien ja suorien tunnistuksen. Tällöin siirto-oppimisella voidaan saavuttaa hyviä tuloksia, vaikka aineistot olisivat erilaiset. Toisaalta joissain tilanteissa siirto-oppiminen voi osoittautua tehottomaksi, jos tutkittavat kohteet ovat liian erilaisia kuin saatavilla olevat aineistot. Esimerkiksi materiaalitieteessä tutkittavat mikrorakenteet poikkeavat tyypillisistä luonnollisista objekteista, mitä esikoulutusaineistot tavallisesti kuvastavat, eikä siirto-oppiminen ole välttämättä paras ratkaisu. [14]

Tutkimusaineiston ja esikoulutusaineiston välistä eroavaisuutta voidaan yrittää ratkaista kaksin- tai moninkertaisella siirto-oppimisella. Tällöin esikoulutukseen käytetään kahta tai useampaa ulkoista aineistoa. Ensimmäisellä aineistolla mallille koulutetaan universaaleja piirteitä ja toisella aineistolla tarkennetaan osaamista lähemmäs tilannekohtaista tarvetta. Täten malli voidaan esikouluttaa tarkemmin tulosten parantamiseksi ilman tarvetta yhdelle isolle tilanteeseen sopivalle esikoulutusaineistolle. [2]

4.3 Ulottuvuuksien vähentäminen

Ulottuvuuksien vähentäminen (engl. *dimensionality reduction*) on piirreavaruuden ulottuvuuksien eli aineiston esimerkkien piirteiden vähentämistä. Aiemmin todettiin, että yksi aineiston ylisovituksen tärkeimmistä syistä on liian korkea ulotteisuus eli piirre-esimerkki suhde. Ulottuvuuksien vähentämisellä voidaan siis yrittää ratkaista pienen datan ongelma pienentämällä tätä suhdelukua. Koneoppimismallit suoriutuvat yleensäkin paremmin vähennetyssä piirreavaruudessa. [4] Ulottuvuuksien vähentämisellä voidaan vähentää niin datankeräys- kuin laitteistovaatimuksia, tehostaa oppimisnopeutta ja parantaa mallin yleistämiskykyä sekä tulkittavuutta [22].

Ulottuvuuksien vähentäminen on yksinkertaisimmillaan piirteiden valintaa. Aineistossa olevista piirteistä pyritään valitsemaan tärkeimmät, jotka edustavat koko aineistoa ja joiden perusteella koneoppimismalli voidaan kouluttaa. Samalla piirteistä karsitaan merkityksettömät, jotka sisältävät ylimääräistä kohinaa. [22] Valittavat piirteet eivät välttämättä ole yksinään merkityksellisimmät, mutta niillä voi olla yhteyksiä muihin piirteisiin ja luokkiin, joten ne tuottavat yhdessä parhaan tuloksen [4].

Piirteiden valintatekniikat voidaan jakaa kääreisiin (engl. *wrappers*), sulautettuihin menetelmiin (engl. *embedded methods*) ja suodattimiin (engl. *filters*). Kääreet käyttävät tietyn koneoppimisalgoritmin tuloksia piirteiden valintaan. Sulautetut menetelmät integroivat piirteiden valinnan suoraan mallin luokitteluvaiheeseen. Suodattimet käyttävät koneoppimismallista riippumattomia mittauksia tai pisteytyksiä, joiden perusteella piirteille lasketaan tärkeysarvoja. Näitä ovat muun muassa redundanssi (engl. *redundancy*) ja komplementaarisuus tai täydentävyys (engl. *complementarity*), jotka mittaavat piirteiden välisiä korrelaatioita. [22]

Piirteiden valinnan lisäksi ulottuvuuksien vähentäminen käsittää myös menetelmiä, joilla piirreavaruus ja esimerkit voidaan muuttaa vähempiulotteiseksi esityk-

seksi jättämättä kokonaisia piirteitä pois. Vanhoista piirteistä voidaan luoda uusia piirteitä tarkoituksena säilyttää suurin osan alkuperäisten piirteiden vaihtelevuudesta ja etäisyyksistä, mutta vähentäen niiden määrää [1]. Pääkomponenttianalyysi eli PCA (engl. *principal component analysis*) on yksi esimerkki tällaisesta tekniikasta. Siinä uudet piirteet eli komponentit lasketaan aineiston kovarianssimatriisia vastaavista ominaisvektoreista. [6]

4.4 Ristiinvalidointi

Luvussa 3.3 todettiin validoinnin olevan tärkeä osa koneoppimismallin koulutusta, sillä sen avulla voidaan kontrolloida ylisovituksen määrää ja varmistaa mallin tarkkuus. Ristiinvalidoinnilla voidaan säästyä erillisen validointiaineiston keräämiseltä tai rajaamiselta koulutusaineistosta. Ristiinvalidoinnista on olemassa useampia toisistaan pienesti eroavia variaatioita, mutta niiden toiminta pohjautuu pitkälti samaan periaatteeseen. Näitä ovat muun muassa k-kertainen ristiinvalidointi (engl. *k-fold cross-validation*) ja sisäkkäinen ristiinvalidointi (engl. *nested cross-validation*), joskus nimellä kaksinkertainen ristiinvalidointi (engl. *double cross-validation*), johon sisältyy sekä mallin valinta että lopullinen arviointi. [4], [8], [16]

k-kertaisessa ristiinvalidoinnissa aineisto jaetaan satunnaisesti k kappaleeseen yhtäsuuria osia. Yksi näistä osista valitaan vuorollaan validointiaineistoksi, ja loput käytetään koulutukseen samoin kuin tavallisessa koulutus- ja validointimenettelyssä erillisien aineistojen kanssa. Tämän jälkeen sama toistetaan, mutta validointiaineistoksi valitaan toinen aikaisemmin koulutuksessä käytetty osa, ja aiempi validointiaineisto lisätään koulutusaineistoon. Tätä toistetaan kunnes kaikkia osia ollaan käytetty kerran validoinnissa. Jokainen koulutussykli tuottaa hieman erilaisen validointituloksen, ja näiden kaikkien keskiarvo muodostaa ristiinvalidoinnin kokonaistuloksen. Tätä tulosta voidaan pitää arviona mallin tarkkuudesta, mikä saataisiin, jos malli olisi koulutettu koko aineistolla. [4], [8], [16]

Jos ristiinvalidointiin halutaan sisällyttää myös mallin ja piirteiden valinta, voidaan käyttää sisäkkäistä ristiinvalidointia. Siinä aineisto jaetaan myös satunnaisesti k osaan. Yksi näistä osista valitaan vuorollaan testiaineistoksi, ja loput muodostavat ulkoisen koulutusaineiston. Toisin kuin k-kertaisessa, ulkoinen koulutusaineisto jaetaan vielä uudelleen l osaan. Näistä sisemmistä osista yksi valitaan vuorollaan validointiaineistoksi, jota käytetään mallin arviointiin ja valintaan. Loput sisemmät osat muodostavat sisemmän eli varsinaisen koulutusaineiston, jolla malli koulutetaan. Malli koulutetaan ja validoidaan l kertaa käyttäen aina uutta osaa validointiaineistona. Tämä toistetaan useammilla erilaisilla malleilla, ja näistä validoiduista malleista valitaan paras, joka koulutetaan koko ulommalla koulutusaineistolla ja sovelletaan lopuksi ulompaan testiaineistoon. Tämän jälkeen testiaineisto vaihdetaan ja koulutus-validointivaihe alkaa alusta. Tämä toistetaan kunnes kaikki k ulompaa osaa ovat toimineet testiaineistona. [16]

Sisäkkäisen ristiinvalidoinnin sisempi kerros toimii mallinrakennusalgoritmina, joka yrittää etsiä sille annetun aineiston perusteella tarkimman mallin. Ulompien testitulosten keskiarvo on arvio mallinrakennusalgoritmin tuottaman mallin tarkkuudesta, jos se olisi valittu ja koulutettu koko aineistolla. [8], [16] Erillinen ulompi testiaineisto vaaditaan, koska mallin valinta on tehty validointiaineiston perusteella, jolloin validointiaineisto on kuulunut epäsuorasti koulutusprosessiin. [4], [8]

Ristiinvalidoinnin heikkous on sen laskennallinen vaativuus. [4] Koneoppimismallin koulutus vaatii aikaa, ja sisäkkäisessä ristiinvalidoinnissa koulutus tehdään kokonaisuudessaan $k*l$ kertaa jokaiselle mallivaihtoehdolle. Lisäksi piirteiden ja hyperparametrien valinta on osaltaan myös laskennallisesti vaativa prosessi, joka tulee myös suorittaa jokaisessa koulutusvaiheessa uudelleen. Tämä on kuitenkin välttämätöntä ylisovituksen estämiseksi. [8]

5 Yhteenveto

Tutkielmassa tarkasteltiin koneoppimismalleja, niiden koulutusta, validointia ja testaamista, koulutusaineiston merkitystä koneoppimismallin koulutuksessa ja pienten aineistojen ominaisuuksia ja vaikutusta mallin tuloksiin. Lopuksi tutkittiin ongelmien ratkaisuksi käytettyjä menetelmiä ja niiden vaikutusta. Tutkimus pyrki vastaamaan neljään tutkimuskysymykseen.

TK1. Pieniä aineistoja kohdataan pääosin tieteellisessä tutkimuksessa: materiaalitieteessä, tekniikan ja biologian aloilla, lääketieteessä, taloustieteessä ja hiukkasfysiikassa. Aineistot voivat joskus olla suuriakin, mutta esimerkkien määrä ei ole riittävä suhteessa niitä kuvaavien piirteiden määrään. Yleisimpiä käyttökohteita pienille aineistoille ovat muun muassa kuvantunnistus, lääketieteellinen diagnoosi, neurokuvantaminen sekä liikkeen- ja katseenseuranta. Pienet aineistot johtuvat datan keräämisen vaikeuksista. Tämä voi johtua datan harvinaisuudesta, vaikeasta tai kalliista kerättävyydestä, tai ongelmista koskien datan yksityisyyttä.

TK2. Koneoppimismallin tarkkuus on suoraan yhteydessä koulutusaineiston laatuun ja kokoon. Liian pienellä aineistolla malli saattaa joko alisovittaa tai ylisovittaa datan. Pienen aineiston jakaminen erillisiin koulutus- ja testiaineistoihin vaikeuttaa tilannetta entisestään. Tällöin käytetään erilaisia ristiinvalidointitekniikoita, jotta koko aineisto voidaan hyödyntää koulutukseen. Ne ovat kuitenkin laskennallisesti vaativampia verrattuna tavalliseen koulutus-testiaineistojakoon.

TK3. Aineiston koon lisäksi koneoppimismalliin vaikuttaa datan laatu eli sen sisältämä kompleksisuus, tasaisuus ja sen ulottuvuudet eli piirteet ja tarkemmin piirteiden määrä. Aineiston kokoa parempi mittari kuvaamaan koneoppimismallin todennäköisyyttä ylisovittaa data onkin ehkä koulutusaineiston piirre-esimerkkisuhde eli kuinka monta piirrettä aineistossa on yhtä esimerkkiä kohden. Jos piirteitä on paljon, puhutaan datan korkeasta ulotteisuudesta, jolloin yhteyksien ja luokkien tunnistaminen aineistosta on vaikeampaa, ja se vaatii edistyneemmän koneoppimismallin ja enemmän dataa.

TK4. Aineiston määrää voidaan kasvattaa sen esimerkkien pohjalta. Tämä voi tarkoittaa uuden aineiston luomista muuttamalla vanhan ominaisuuksia esimerkiksi lisäämällä kuvaan erilaisia suodattimia tai kääntämällä sitä. Myös kokonaan uusia esimerkkejä voidaan luoda synteettisillä ylinäytteistystekniikoilla. Nämä voivat olla yksinkertaisia algoritmeja (SMOTE, ADASYN) tai generatiivisia tekoälymalleja (DetectorGAN, DeepSMOTE). Tavallinen tekoälymalli ei kuitenkaan ole riittävä, sillä se pyrkii maksimoimaan generoidun datan realistisuuden ihmisen näkökulmasta eikä takaa sen auttavan koulutuksessa. Malli pitää olla suunniteltu juuri koulutusaineiston luomiseen. Siirto-oppimisella voidaan parantaa mallin oppimistehokkuutta. Tällöin esikoulutuksessa käytetyn aineiston tulee olla mahdollisimman samankaltainen kuin varsinainen koulutusaineisto. Siirto-oppiminen voidaan tehdä yksin- tai moninkertaisena. Aineiston ulottuvuuksien eli piirteiden määrää voidaan vähentää karsimalla turhia piirteitä pois tai tiivistämällä vanhat piirteet pienempään määrään piirteitä. Käyttämällä ristiinvalidointia voidaan hyödyntää koko aineisto koulutukseen, mutta piirteiden ja mallin valintaan liittyvät vaiheet on tehtävä erikseen jokaisessa validointisyklissä, mikä lisää laskennallista vaativuutta.

Aineistojen rajallisuus on monialainen ja monipuolinen ongelma. Ratkaisumenetelmät vaihtelevat käyttökohteesta riippuen ja ne tuottavat vaihtelevia tuloksia. Tutkimusta voidaan jatkaa alakohtaisesti erilaisista ratkaisumenetelmistä sekä nii-

den tehokkuudesta ja kehittämisestä. Lisäksi aineiston keräämisen ongelmia voidaan tutkia ja yrittää ratkaista esimerkiksi laajemmalla yhteistyöllä eri tahojen välillä aineistojen keräämisessä ja jakamisessa. Tekoälyn luotettava käyttö pienien aineistojen kanssa voi olla seuraava askel tekoälyn saatavuuden ja käytettävyyden parantamisessa.

Lähdeluettelo

[1] S. Badillo, B. Banfai, F. Birzele et al., "An Introduction to Machine Learning", en, *Clinical Pharmacology & Therapeutics*, vol. 107, nro 4, s. 871–885, 2020. DOI: 10.1002/cpt.1796.

[2] K. Wang, "An Overview of Deep Learning Based Small Sample Medical Imaging Classification", teoksessa *2021 International Conference on Signal Processing and Machine Learning (CONF-SPML)*, marraskuu 2021, s. 278–281. DOI: 10.1109/CONF-SPML54095.2021.00060.

[3] P. Kokol, M. Kokol ja S. Zagoranski, "Machine learning on small size samples: A synthetic knowledge synthesis", *Science Progress*, vol. 105, nro 1, tammikuu 2022. DOI: 10.1177/00368504211029777.

[4] A. Vabalas, E. Gowen, E. Poliakoff ja A. J. Casson, "Machine learning algorithm validation with a limited sample size", *PLOS ONE*, vol. 14, nro 11, 7. marraskuuta 2019. DOI: 10.1371/journal.pone.0224365.

[5] T. Prakash, T. Dhamija, R. Kumar ja J. Panda, "Leveraging Explainable Artificial Intelligence for Understanding the Effect of Model Capacity on Training Dataset Size", teoksessa *2022 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*, joulukuu 2022, s. 1–6. DOI: 10.1109/SOLI57430.2022.10294527.

[6] L. Yang ja A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice", *Neurocomputing*, vol. 415, s. 295–316, marraskuu 2020. DOI: 10.1016/j.neucom.2020.07.061.

[7] H. O. Alanazi, A. H. Abdullah ja K. N. Qureshi, "A Critical Review for Developing Accurate and Dynamic Predictive Models Using Machine Learning Methods in Medicine and Health Care", en, *Journal of Medical Systems*, vol. 41, nro 4, s. 69, maaliskuu 2017. DOI: 10.1007/s10916-017-0715-6.

[8] S. Varma ja R. Simon, "Bias in error estimation when using cross-validation for model selection", en, *BMC Bioinformatics*, vol. 7, nro 1, s. 91, joulukuu 2006. DOI: 10.1186/1471-2105-7-91.

[9] L. Liu, M. Muelly, J. Deng, T. Pfister ja L.-J. Li, "Generative modeling for small-data object detection", teoksessa *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, lokakuu 2019, s. 6072–6080. DOI: 10.1109/ICCV.2019.00617.

[10] S. U. Sabha, A. Assad, N. M. U. Din ja M. R. Bhat, "Comparative Analysis of Oversampling Techniques on Small and Imbalanced Datasets Using Deep Learning", teoksessa *2023 3rd International conference on Artificial Intelligence and Signal Processing (AISP)*, maaliskuu 2023, s. 1–5. DOI: 10.1109/AISP57993.2023.10134981.

[11] L. P. F. Garcia, A. C. P. L. F. de Carvalho ja A. C. Lorena, "Effect of label noise in the complexity of classification problems", *Neurocomputing*, vol. 160, s. 108–119, heinäkuu 2015. DOI: 10.1016/j.neucom.2014.10.085.

[12] S. Ko, J. Choi ja J. Ahn, "GVES: machine learning model for identification of prognostic genes with a small dataset", en, *Scientific Reports*, vol. 11, nro 1, s. 439, tammikuu 2021. DOI: 10.1038/s41598-020-79889-5.

[13] P. T. Komiske, E. M. Metodiev, B. Nachman ja M. D. Schwartz, "Learning to classify from impure samples with high-dimensional data", *Physical Review D*, vol. 98, nro 1, heinäkuu 2018. DOI: 10.1103/PhysRevD.98.011502.

[14] Z. Yang, R. Al-Bahrani, A. C. E. Reid et al., "Deep learning based domain knowledge integration for small datasets: Illustrative applications in materials informatics", teoksessa *2019 International Joint Conference on Neural Networks (IJCNN)*, heinäkuu 2019, s. 1–8. DOI: 10.1109/IJCNN.2019.8852162.

[15] B. Kim, C. Kim, J. Lee, J. Song ja G. Park, *Data-Efficient Deep Learning Method for Image Classification Using Data Augmentation, Focal Cosine Loss, and Ensemble*, heinäkuu 2020. url: http://arxiv.org/abs/2007.07805 (viitattu 28. 10. 2024).

[16] T. Burzykowski, M. Geubbelmans, A.-J. Rousseau ja D. Valkenborg, "Validation of machine learning algorithms", *American Journal of Orthodontics and Dentofacial Orthopedics*, vol. 164, nro 2, s. 295–297, elokuu 2023. DOI: 10.1016/j.ajodo.2023.05.007.

[17] T. Tran, T. Pham, G. Carneiro, L. Palmer ja I. Reid, "A Bayesian Data Augmentation Approach for Learning Deep Models", teoksessa *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017. url: https://arxiv.org/abs/1710.10564 (viitattu 07. 10. 2024).

[18] R. Ghnemat ja S. Al-mashaqbeh, "Novel Image Data Augmentation Technique for Deep Learning Using Least Significant Bit Encryption", sarja ICMLT '24, Association for Computing Machinery, 2024, s. 143–152. DOI: 10.1145/3674029.3674053.

[19] N. V. Chawla, K. W. Bowyer, L. O. Hall ja W. P. Kegelmeyer, ”SMOTE: Synthetic Minority Over-sampling Technique”, en, *Journal of Artificial Intelligence Research*, vol. 16, s. 321–357, kesäkuu 2002. DOI: 10.1613/jair.953.

[20] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., ”Generative Adversarial Nets”, teoksessa *Advances in Neural Information Processing Systems*, vol. 27, Curran Associates, Inc., 2014. url: https://proceedings.neurips.cc/paper_files/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html (viitattu 20. 11. 2024).

[21] D. Dablain, B. Krawczyk ja N. V. Chawla, ”DeepSMOTE: Fusing Deep Learning and SMOTE for Imbalanced Data”, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, nro 9, s. 6390–6404, syyskuu 2023. DOI: 10.1109/TNNLS.2021.3136503.

[22] Y. Zhang, R. Zhu, Z. Chen, J. Gao ja D. Xia, ”Evaluating and selecting features via information theoretic lower bounds of feature inner correlations for high-dimensional data”, *European Journal of Operational Research*, vol. 290, nro 1, s. 235–247, huhtikuu 2021. DOI: 10.1016/j.ejor.2020.09.028.