Melinda Suvivirta

Oppiva tekoäly ja takaovihyökkäykset

Tietotekniikan kandidaatintutkielma

5. toukokuuta 2022

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Melinda Suvivirta

Yhteystiedot: melinda.m.suvivirta@student.jyu.fi

Ohjaaja: Timo Tiihonen

Työn nimi: Oppiva tekoäly ja takaovihyökkäykset

Title in English: Learning AI and backdoor attacks

Työ: Kandidaatintutkielma

Sivumäärä: 28+0

Tiivistelmä: Tekoälyä voidaan opettaa havainnoimaan itsenäisesti ja tekemään päätöksiä sen pohjalta. Tämä tutkielma kuvaa tekoälyn ohjattua opettamista ja esittelee tekoälyn opettamiseen liittyviä riskejä, joista pääpainotus on datan myrkyttämiseen liittyvissä takaovihyökkäyksissä. Riskejä on sekä tahattomia että tahallisia, mutta yhteistä niille on, että ne pohjautuvat vahvasti siihen, millaisella datalla tekoälyä opetetaan. Takaovihyökkäykset luovat vakavan uhan monille tekoälyohjelmille ja niiden tunnistaminen on usein haastavaa.

Avainsanat: tekoäly, koneoppiminen, datan myrkytys, takaovihyökkäykset

Abstract: AI can be teached to make observations and then make independent choices based on them. This thesis describes supervised learning of AI and undergoes different risks that are involved with teaching AI. The main focus is on backdoor attacks that are involved with data poisoning. There are both unintentional and intentional risks associated with teaching AI. The risks are mainly based on what kind of data the AI is being taught with. Backdoor attacks create a serious threat to several AI programs and it is often difficult to detect them.

Keywords: artificial intelligence, machine learning, data poisoning, backdoor attacks

Kuviot

Sisällys

1 Johdanto

Tekoäly on kehittynyt merkittäväksi osaksi yhteiskuntaa ja sen luotetaan toimivan itsenäisenä päätöksentekijänä useissa automatisoiduissa prosesseissa (Duan, Edwards ja Dwivedi 2019). Tästä johtuen tekoälyn varaan luotetaan paljon ja sen voidaan antaa osallistua esimerkiksi valtion hallituksen päätöksentekoon (Janssen ym. 2020) ja toimia itsenäisesti myös kriittisissä turvallisuutta vaativissa tehtävissä (Chen ym. 2017). Tekoälyn itsenäinen päätöksenteko aiheuttaa toisinaan huolta, mutta samalla siihen myös luotetaan (Araujo ym. 2020).

Tekoälyn tarkoitus on auttaa ihmisiä toimimaan niin yksityiselämän kuin yhteiskunnan osallisina helposti ja turvallisesti. Ratkaisuna tekoälyyn liittyviin ongelmiin ei voida pitää sitä, että lakattaisiin käyttämästä tekoälypohjaisia ratkaisuja ja annettaisiin ihmisten hoitaa kaikki tekoälyn tehtävät. Tekoälyä hyödynnetään jo nyt niin laajoissa määrin kaikkialla, että ihmiset eivät enää mitenkään pystyisi tekemään kaikkea sitä työtä, mitä tekoäly tekee jatkuvasti tälläkin hetkellä (Kotsiantis, Zaharakis, Pintelas ym. 2007). Esimerkiksi YouTubeen julkaistaan satoja tuhansia tunteja videoita päivittäin (Cheng, Dale ja Liu 2008). Jokainen näistä videoista päätyy tekoälyn tarkastamaksi esimerkiksi sisällön sopivuuden arvioimiseksi. Olisi mahdotonta palkata ihmisiä tarkistamaan jokainen sekunti jokaisesta videosta, joten tehtävä on luotettava tekoälyn vastuulle.

Tässä tutkielmassa keskitytään oppivaan tekoälyyn, jolla tarkoitetaan tekoälyä joka kykenee kehittymään paremmaksi tehtävässään. Esimerkiksi Googlen hakukone hyödyntää oppivaa tekoälyä, sillä hakukone oppii ehdottamaan parempia hakusanoja ja tarjoamaan käyttäjälleen juuri tämän haluamia hakutuloksia (Mühlhoff 2020). Tekoäly voi kuitenkin oppia myös väärin, niin tahallisesti aiheutettuna kuin vahingossa. Väärinoppimisen aiheuttamien vahinkojen laajuus on tapauskohtaista. Kyseessä voi olla yksittäiseen henkilöön kohdistuva ikävä haitta, tai suuri yhteiskunnallinen ongelma. Tutkielma keskittyy oppivan tekoälyn tahallisesti aiheutettuihin riskeihin, jotka vaikuttavat jo tekoälyn opetusvaiheessa sen toimintaan. Näistä riskeistä pääpaino on opetusdatan myrkytykseen liittyvissä takaovihyökkäyksissä.

Tutkielma käy aluksi läpi koneoppimisen perustoimintaa ja keskittyy ohjattuun oppimiseen. Toisessa luvussa käsitellään ohjattuun oppimiseen liittyviä käsitteitä, kuten luokittelu ja regres-

sio. Kolmannessa luvussa tutkielma siirtyy käsittelemään datan merkitystä tekoälyn toiminnalle ja sitä, millaisia riskejä tekoälyn opettamiseen liittyy. Luvussa käydään läpi esimerkkejä niin tahattomista kuin tahallisesti aiheutetuista riskeistä. Tahallisesti aiheutetuista riskeistä tutkielma pääsee neljännessä luvussa käsittelemään takaovihyökkäyksiä. Tutkielma käy läpi takaovihyökkäyksien toimintaa, antaa esimerkkkejä takaovihyökkäyksistä sekä lopuksi viidennessä luvussa pyrkii tarjoamaan ratkaisuja takaovihyökkäyksiltä suojautumiseen.

2 Oppiva tekoäly

Oppiva tekoäly hyödyntää koneoppimista. Koneoppiminen on laajasti hyödynnetty tekoälyn osa-alue, jossa nimensä mukaisesti pyritään opettamaan konetta, kuten robottia tai ohjelmaa, toimimaan itsenäisesti halutulla tavalla. Koneoppimisella pyritään saamaan ohjelma tekemään oikeita arvauksia siitä, kuinka sen tulisi toimia tietyissä tilanteissa (Mohri, Rostamizadeh ja Talwalkar 2018). Opettaminen ei tapahdu if-else -lausekkeilla tai valmiilla algoritmeilla, sillä olisi mahdotonta tuottaa valmis toimintamalli jokaiseen tilanteeseen. Sen sijaan ohjelmalle tarjotaan paljon esimerkkejä oikeine vastauksineen ja ohjelman toivotaan päätyvän esimerkkien kautta oikeaan lopputulokseen vastaavissa esimerkkien kaltaisissa tapauksissa (Mohri, Rostamizadeh ja Talwalkar 2018). Tekoälyä voidaan opettaa useilla eri tavoilla ja eri tavat soveltuvat erilaisiin tavoitteisiin. Näitä tapoja avataan seuraavassa alaluvussa, jonka jälkeen perehdytään tutkielman kannalta olennaiseen ohjattuun oppimiseen ja lopuksi avataan tekoälyn opettamista erilaisten esimerkkien kautta.

2.1 Tekoälyn opettamisen keinoja

Koneoppimisen kaksi suurinta luokkaa ovat ohjattu oppiminen (engl. supervised learning) ja ohjaamaton oppiminen (engl. unsupervised learning). Ohjattu oppiminen tarjoaa ohjelmalle syöte-vaste -pareja sisältävää opetusdataa (Kotsiantis, Zaharakis, Pintelas ym. 2007). Opettaminen on ohjattua juuri siksi, että ohjelmalle tarjotaan oikeita vastauksia, eli vasteita, joiden mukaisesti ohjelman tulee oppia toimimaan. Ohjelma oppii siis antamaan syötteenä saadulle datalle jonkin arvon, joka pyritään saamaan mahdollisimman lähelle vastetta. Tarkoituksena on, että ohjelma myöhemmin kykenee toimimaan itsenäisesti opetusdatan kaltaisen datan parissa eli määrittää itsenäisesti vasteen saamalleen syötteelle. Ohjattua oppimista hyödynnetään erilaisissa luokittelua vaativissa tehtävissä, kuten kasvojentunnistuksessa (Kotsiantis, Zaharakis, Pintelas ym. 2007) ja sähköpostien lajittelussa (Awad ja ELseuofi 2011).

Ohjaamattomassa oppimisessa tekoälyn avulla pyritään löytämään suhteita eri syötteiden välillä (Kotsiantis, Zaharakis, Pintelas ym. 2007). Ohjaamattoman oppimisen keskeinen käsite on klusterointi (engl. clustering), joka kuvaa datan sisältämien objektien jakamista ryhmiin.

Ryhmien sisäisten objektien toivotaan olevan keskenään mahdollisimman samankaltaisia ja samalla poikkeavan mahdollisimman paljon muiden ryhmien objekteista (Madhulatha 2012). Klusterointia voidaan hyödyntää esimerkiksi biologiassa muodostamaan eläimille ja kasveille luokkia niiden ominaisuuksien perusteella tai markkinoinnissa etsimään yhteyksiä asiakkaiden ostokäytöksessä, jotta heille voidaan markkinoida juuri oikeanlaisia tuotteita (Madhulatha 2012).

Ohjatun ja ohjaamattoman oppimisen lisäksi on muitakin tapoja opettaa tekoälyä. Yksi tapa on vahvistusoppiminen (engl. reinforcement learning), jossa ohjelman niin sanottu oppiva agentti (engl. learning agent) saa ympäristöstään niin positiivista kuin negatiivista palautetta. Ohjelma pyrkii siten muuttamaan toimintaansa niin, että saisi mahdollisimman paljon positiivista, ja mahdollisimman vähän negatiivista palautetta (Sutton 1992). Vahvistusoppimista hyödynnetään esimerkiksi robotti-imureissa, jotka oppivat tuntemaan ympäristönsä ja siten esimerkiksi välttämään alueita, joihin ne eivät pääse (Donepudi 2020).

2.2 Ohjattu oppiminen

Mainituista tekoälyn opettamisen keinoista keskitytään tässä tutkielmassa yksinkertaisuuden vuoksi pelkästään ohjattuun oppimiseen, johon perehdytään seuraavaksi tarkemmin. Ohjattu oppiminen perustuu siis tekoälyohjelmalle annettavaan syöte-vaste -pareja sisältävään opetusdataan, jonka syötteen ohjelma sitten opettelee luokittelemaan tai muodostaa siitä ennustettavia lukuarvoja vasteen mukaisesti (Kotsiantis, Zaharakis, Pintelas ym. 2007).

Ohjatussa opettamisessa on kyse regressiosta (eng. regression) tai luokittelusta (eng. classification). Regressio pyrkii muodostamaan funktion, joka saadun syötteen perusteella palauttaa jonkin numeerisen arvon (Letzgus ym. 2021). Regressio palauttaa siis lukuarvon, joka on tekoälyn antama ennuste vasteelle annetun syötteen perusteella. Regressiolla toimiva tekoälyohjelma voi esimerkiksi arvioida asuntojen hintaa tietyllä alueella (Manasa, Gupta ja Narahari 2020) tai ennustaa lämpötilojen vaihtelua (Radhika ja Shashi 2009).

Luokittelu on yksi yleisimmistä koneoppimisen käyttötarkoituksista ja sitä hyödynnetään esimerkiksi roskapostien erottamisesta tärkeiden viestien joukosta (Novaković ym. 2017). Dataa luokitteleva ohjelma palauttaa regressiosta poiketen syötteeseen sopivan vasteen, joka

kuvastaa ennalta määriteltyä luokkaa (Mohri, Rostamizadeh ja Talwalkar 2018). Näitä luokkia voi olla kaksi tai enemmän. Esimerkiksi sähköpostin tunnistaminen tärkeäksi tai roskapostiksi vaatii kaksi luokkaa (Awad ja ELseuofi 2011), kun taas käsin kirjoitettujen numeroiden tunnistaminen vaatii 10 luokkaa, jotka ovat luvut 0-9 (Shamim ym. 2018). Tekoäly kykenee oikein opetettuna jopa ihmistä paremmin ja nopeammin luokittelemaan dataa (Gu ym. 2019).

Luokitteluun liittyy oleellisesti käsitteet herkkyys (eng. sensitivity) ja spesifisyys (eng. specificity), jotka kuvaavat sitä, kuinka usein luokitteleva ohjelma antaa oikeita tuloksia. Käsitteillä siis voidaan arvioida sitä, kuinka varmasti ohjelma antaa oikean tuloksen. Positiiviset tapaukset tässä tarkoittavat sitä, että luokitteluehto täyttyy, ja negatiiviset tapaukset taas sitä, että ehto ei täyty. Ylläolevassa luvussa mainitussa käsinkirjoitettuja numeroita luokittelevassa ohjelmassa jokaista käsinkirjoitettua numeroa edustaisi yksi positiivinen tapaus, eli se luku jota numero edustaa, ja kaikki muut luvut olisivat negativiisia tapauksia. Jos käsinkirjoitettu symboli ei edustaisi mitään lukua, olisivat kaikki negatiivisia tapauksia. Herkkyys kuvaa sitä, kuinka monta positiivista tapausta ohjelma löytää kaikkien positiivisten tapausten joukosta ja spesifisyys sitä, kuinka monta negatiivista tapausta ohjelma löytää kaikkien negatiivisten tapausten joukosta (Mohri, Rostamizadeh ja Talwalkar 2018). Korkea herkkyys ja spesifisyys kertovat siitä, että ohjelma luokittelee saamansa syötteen varsin luotettavasti oikein.

Kun ohjelma tunnistaa positiiviset tapaukset oikein, eli sen herkkyys on korkea, sanotaan näiden tulosten olevan arvoltaan true positive. Vastaavasti kun ohjelma tunnistaa negatiiviset tapaukset oikein, eli sen spesifisyys on korkea, ovat nämä oikein luokitellut tulokset arvoltaan true negative. Käsitteet false negative ja false positive taas kuvaavat ohjelman luokittelussa tapahtuvaa virhettä (Mohri, Rostamizadeh ja Talwalkar 2018). False negative antaa negatiivisen tuloksen vaikka luokittelun ehto täyttyy ja false positive taas palauttaa positiivisen arvon, vaikka sen ehto ei täyty.

		Todellinen luokka				
		iloinen	surullinen	vihainen	ystävällinen	säikähtänyt
Ennustettu luokka	iloinen	51	2	1	1	1
	surullinen	3	23	1	1	0
	vihainen	2	2	17	0	0
	ystävällinen	0	1	2	9	1
	säikähtänyt	1	0	1	1	18

Kuvio 1. Novaković ym. (2017) luoma sekoitusmatriisi (engl. confusion matrix) näyttää tuloksia tekoälyohjelmasta, joka arvioi ihmisen tunnetilaa videolta.

Yllä olevasta kuviosta voidaan nähdä sekoitusmatriisi, joka sisältää erilaisia tuloksia arvoineen. Tapaukset joissa todellinen luokka (engl. actual class) ja ennustettu luokka (engl. predicted class) ovat samoja, esimerkiksi ’iloinen - iloinen’, ovat arvoltaan true positive. True negative tapaukset taas näkyvät siinä, kun esimerkiksi todellisen luokan ollessa ’säikähtänyt’ ei tekoäly ennusta sen olevan ’surullinen’ tai ’vihainen’ eli ruudussa oleva luku on 0. Kun luokan ’vihainen’ edustajan tulkitaan virheellisesti olevan ’ystävällinen’ on kyseessä false positive ja taas jokainen ’surullinen’ jonka ei tunnistettu olevan luokaltaan ’surullinen’ edustaa false negative -tapausta.

2.3 Esimerkkejä luokittelevista tekoälyohjelmista

Viimevuosina vaikuttanut Covid 19 -pandemia on osaltaan vaikuttanut tekoälyohjelmien kehittämiseen. Tekoälyä voidaan esimerkiksi hyödyntää potilaiden kunnon arvioimiseen luokittelemalla potilaat tiettyihin kategorioihin oireiden perusteella (Ainapure ym. 2021). Ainapure ym. (2021) esittää neljä kategoriaa potilaiden kunnon määrittämiseksi: "kotihoidon tarpeessa", "täysin terve", "sairaalahoidon tarpeessa"ja "hengenvaarassa". Tekoäly pystyi luokittelemaan potilaat oikeisiin kategorioihin jopa 98.5% tarkkuudella (engl. accuracy). Tarkkuus saadaan laskemalla true positive -tapauksien suhde kaikkiin tapauksiin.

Pahar ym. (2021) on kehittänyt tekoälyohjelman, joka älypuhelimen mikrofonin kautta ääntä tallentamalla analysoi yskimistä, ja päättelee sen perusteella, onko yskivä henkilö sairastunut koronaan. Ohjelma saatiin parhaimmillaan päättelemään sairastuminen 98% varmuudella oi-

kein. Apostolopoulos ja Mpesiana (2020) kehittämä ohjelma taas havaitsee röntgenkuvista merkkejä koronaviruksesta. Ohjelman herkkyys saatiin parhaimmillaan toimimaan 98.66% tarkkuudella ja spesifisyys 94.46% tarkkuudella. Tämä tarkoittaa sitä, että useammin tulkittiin terve ihminen sairaaksi false positive -tapauksena kuin sairas ihminen terveeksi false negative -tapauksena. Voisi olettaa että tämä on tarkoituksellista, sillä on parempi tunnistaa terve ihminen sairaaksi, jolloin tämä turhaan on sairaslomalla ja saa hoitoa ilman tarvetta sille, kuin tunnistaa sairas ihminen terveeksi, jolloin tämä ei saa tarvitsemaansa hoitoa ja voi sairastua vielä vakavammin.

Koronapandemian aikana misinformaation määrä on lisääntynyt sosiaalisessa mediassa sisältäen esimerkiksi virheellistä tai väärin johdattelevaa tietoa terveydenhuollosta ja hallituksesta (Brennen ym. 2020). Tekoälyohjelmia voidaan hyödyntää esimerkiksi misinformaation, valeuutisten ja propagandan, sekä näitä levittävien bottien tunnistamiseen ja hallitsemiseen. Valeuutisten havaitseminen esimerkiksi Facebookin valtavasta datamäärästä voi kuitenkin tuottaa vaikeuksia tekoälyllekin (Manzoor, Singla ym. 2019). Valeuutisten tunnistamisen hankaluus johtuu siitä, että on vaikea määrittää tarkasti tiettyjä valeuutisille ominaisia piirteitä (Shu ym. 2017). Esimerkiksi Facebook, Twitter ja YouTube käyttävät oikeita ihmisiä faktojen tarkistamiseen (engl. fact-checker), mutta koronapandemian aikana levinneen misinformaation määrä on niin laaja, että he eivät saa kaikkia julkaisuja tarkistettua (Brennen ym. 2020). Vaikka tekoäly ei tunnista kaikkia valeuutisia, voidaan sitä silti hyödyntää ainakin osittain ihmistarkastajien taakan vähentämiseksi (Manzoor, Singla ym. 2019).

Vaikka tekoälyä toisinaan verrataan ihmisen älykkyyteen, ei koneen tai ohjelman päätöksenteko kuitenkaan toimi samalla tavalla kuin ihmisen. Koneoppimisella varustettu ohjelma ei osaa tehdä assosiaatioita eri asioiden välille tai muodostaa konsepteja laajemmista käsitteistä (Comiter 2019). Comiter (2019) kuvaa seuraavan esimerkin tekoälyn toimimisesta epätoivotulla tavalla. Ohjelma voi oppia tunnistamaan, että stop-merkkiin kuuluu punainen väri, kahdeksankulmainen muoto ja kirjaimet ’S’, ’T’, ’O’ ja ’P’, mutta se ei ymmärrä mikä stop-merkin tarkoitus on eikä välttämättä tunnista esimerkiksi graffitilla sotkettua, tai vinossa olevaa stop-merkkiä. Ohjelma tietää vain, että tietyt tunnisteet tarkoittavat sitä että objektin nimi on stop-merkki ja että stop-merkin kohdalla auto pysäytetään. Koska tekoäly ei ymmärrä asiayhteyksiä, ei se tiedä mikä ero on liikennetolpassa olevassa stop-merkissä ja jalan-

kulkijan paidassa olevassa printissä, jossa on stop-merkin kuva. Juuri tämä ero ihmismielen ja tekoälyn välillä mahdollistaa tekoälyohjelmien väärinkäytön. Ihmistä ei voi huijata uskomaan, että vinossa oleva stop-merkki tarkoittaisi vihreää liikennevaloa, mutta koneen voi. Tulevissa luvuissa avataan sitä, kuinka konetta voidaan huijata sen opetusvaiheessa.

3 Oppivan tekoälyn riskejä

Oppivaan tekoälyyn sisältyy useita riskejä, jotka voivat olla joko tahattomasti tai tahallisesti aiheutettuja. Tahattomilla riskeillä tarkoitetaan sitä, että kukaan ei tietoisesti yritä saada tekoälyohjelmaa toimimaan väärin, vaan esimerkiksi ohjelman kehittäjiltä jää opetusvaiheessa jokin asia huomaamatta. Tahallisesti aiheutetuilla riskeillä taas kuvataan riskejä, joihin liittyy selkeästi jokin hyökkäävä taho, joka tietoisesti pyrkii saamaan ohjelman toimimaan jollakin tapaa väärin. Koska koneoppimista hyödyntävän ohjelman toiminta perustuu niin vahvasti sen saamaan opetusdataan, suurin osa oppivan tekoälyn riskeistä pohjautuu joko tahattomasti tai tahallisesti syötettyyn manipuloituun, liian suppeaan tai yksipuoliseen dataan (Chen ym. 2017). Seuraavissa kappaleissa käsitellään sekä tahattomien virheiden aiheuttamia riskejä, että tahalliseen vaikuttamiseen perustuvia riskejä.

3.1 Tahattomat riskit

Opetettu tekoälyohjelma toimii täysin saamansa datan perusteella (Chen ym. 2017). Mikäli opetusdatan syöte-vaste -parit sisältävät virheitä, ei tekoäly välttämättä opikaan toimimaan oletetulla tavalla, vaan saattaa tehdä yllättäviä ja jopa vaarallisia päätöksiä perustuen virheelliseen datan tulkintaan. Tällöin ohjelman herkkyys ja spesifisyys laskevat. Huolimattomasti opetettu ohjelma siis toimii huolimattomasti. Opetusdatan toimivuuden varmistaminen on ehdottoman tärkeä turvallisuustoimi tekoälyohjelmien väärinkäytön ja virheiden ennaltaehkäisemiseksi (Gu ym. 2019).

Opetusdatassa on tärkeää huomioida sekä sen määrä (engl. quantity) että laatu (engl. quality). Määrällä kuvataan sitä, että ohjelma saa tarpeeksi paljon dataa, jolloin se oppii muodostamaan laajoja yhteyksiä (Ying 2019). Oppiva tekoäly vaatii valtavia määriä syöte-vaste -pareja oppiakseen mahdollisimman monipuolisesti erilaisia tilanteita. Jos dataa ei ole tarpeeksi, saattavat jotkin tapaukset korostua siinä liialti, jolloin ohjelma saattaa tehdä virheellisiä assosiaatioita luokittelussa. Tätä kutsutaan ylisovittamiseksi (engl. overfitting), jolloin ohjelma toimii erinomaisesti opetusdatan kanssa, mutta mikä tahansa muu data aiheuttaa virheellisiä tuloksia (Ying 2019).

Opetusdatan laadulla kuvataan sitä, että data on tarpeeksi monipuolista eikä täten sisällä esimerkiksi ennakko-oletuksia (Bolukbasi ym. 2016). Liian yksipuolisella datalla opetettu tekoälyohjelma voi muodostaa virheellisiä päätelmiä ja tehdä sopimattomia yhdistelmiä asioiden välille, jotka eivät oikeasti korreloi keskenään (Zhao ym. 2017). Esimerkiksi sanoja luokitteleva tekoälyohjelma yhdisti termin 'cooking' yli 33% todennäköisemmin naisiin kuin miehiin, sillä sen opettama data sisälsi tietynlaisia odotuksia sukupuolesta riippuen (Zhao ym. 2017). Tekoälyllä toimiva kasvojentunnistusohjelma ei välttämättä tunnistaisi tummaihoisia käyttäjiä, jos tekoäly on opetusvaiheessaan saanut harjoitella lähinnä vain valkoihoisten henkilöiden tunnistamista (Leslie 2020). Tällöin opetusdata ei ole ollut tarpeeksi kattavaa kaikkien dataobjektien suhteen tasaisesti.

3.2 Tahalliset riskit

Tekoälyyn perustuvan automaation yleistyminen kaikkialla altistaa sen kyberhyökkäyksille yhä useammin (Terziyan, Golovianko ja Gryshko 2018). Itsenäisesti toimivien tekoälyohjelmien toimintaa ei välttämättä tarkkailla jatkuvasti, jolloin hyökkääjät näkevät tämän mahdollisuutena vaikuttaa tekoälyn toimintaan haitaten sen alkuperäistä käyttötarkoitusta. Tekoälyä hyödynnetään paljon esimerkiksi pankkien ja virastojen suojauksissa sekä haitallisten tietokoneohjelmien tunnistamisessa (Chen ym. 2017). Tästä johtuen monet toimet ovat hyvin haavoittuvaisia, mikäli niistä päättävä tekoäly ei toimisikaan odotetulla tavalla tai joku pääsisi manipuloimaan tekoälyn toimintaa. Tutkielmassa käsitellään hyökkäyksiä, joissa ohjelman manipulointi tapahtuu sen käyttämän datan kautta.

3.2.1 Motiivit hyökkäyksien taustalla

Hyökkääjillä voi olla useita syitä tekoälyohjelmien manipuloinnin taustalla. Hyökkääjien motiivina voi olla esimerkiksi aiheuttaa vahinkoa, piilottaa tietoa tai herättää epäluottamusta jotakin järjestelmää kohtaan (Comiter 2019). Vahingon aiheuttaminen voi olla esimerkiksi sitä, että itseajava auto tunnistaa virheellisesti stop-merkin vihreäksi liikennevaloksi ja aiheuttaa liikenneonnettomuuden, tai että syöpää röntgenkuvista tunnistava ohjelma ei tunnista kaikkia kasvaimia, jolloin potilaat eivät saa tarvitsemaansa hoitoa (Comiter 2019). Nämä false negative -tulokset voivat olla jopa katastrofaalisia, jos liikenneonnettomuus esimerkik-

si sattuisi hyvin vilkkaalla tiellä tai useita henkilöitä kuolisi hoidon puutteen vuoksi. Hyökkääjä pyrkii siis syöttämään ohjelmalle sellaista dataa, joka saa ohjelman tuottamaan false negative -tuloksia.

Tekoälyohjelmaa manipuloimalla voidaan piilottaa tietoa niin, että sitä ei tunnisteta haitalliseksi tai vaaralliseksi (Comiter 2019). Kyseessä voisi olla esimerkiksi sosiaalisen median alustalla pyörivä tekoälyohjelma, joka tunnistaa haitallisia julkaisuja ja tunnistettuaan poistaa julkaisun ja estää viestin julkaisijaa kirjoittamasta lisää vastaavia viestejä. Haitalliset julkaisut voisivat olla esimerkiksi propagandaa tai misinformaatiota levittäviä lähteitä tai sisältää jopa asekauppaa tai lapsipornografiaa. Ohjelman opetusdataa manipuloimalla olisi kuitenkin mahdollista saada ohjelma jättämään huomioimatta esimerkiksi tiettyjä sanoja sisältävät julkaisut. Jos ehtona on esimerkiksi se, että julkaisu ei sisällä mitään haitallista sisältöä, niin tällaiset false positive -tapaukset pääsisivät kuitenkin läpi eli ne jäisivät esille ja voisivat aiheuttaa vakavia ongelmia esimerkiksi edistämällä rikollisuutta ja misinformaation levittämistä.

Manipuloinnin tarkoituksena voi olla myös epäluottamuksen herättäminen jotakin ohjelmaa tai systeemiä kohtaan (Comiter 2019). Tällainen toiminta voi hyödyttää esimerkiksi murtovarkaita tai muita rikollisia, jos he pääsevät käsiksi tekoälyllä varustetun valvontakameran toimintaan. Jos esimerkiksi kyseinen valvontakamera hyökkääjän toimesta hälyttää usein virheellisesti, saatetaan valvontakamera kytkeä pois päältä turhien hälytysten estämiseksi (Comiter 2019). Tämä luo oivan tilaisuuden toimia jäämättä kiinni ja rikoksen tapahduttua ei välttämättä osattaisi aavistaa, että kamera oli kytketty pois juuri hyökkääjien aiheuttamien turhien hälytysten takia.

Epäluottamuksen herättämisen tavoitteena voi olla muukin kuin suoraan rikollinen toiminta. Tarkoituksena voi olla esimerkiksi saada ihmiset välttämään jonkin tietyn yrityksen palveluita, jolloin yrityksen kilpailijat voisivat hyötyä tilanteesta. Tekoälyohjelman voi opettaa tekemään jopa seksistisiä tai rasistisia päätelmiä ihmisistä, mikäli sille annettu data sisältää ennakko-oletuksia esimerkiksi sukupuoleen tai kansallisuuteen liittyen (Bolukbasi ym. 2016). Kuvien julkaisemiseen perustuva sosiaalinen media voisi hyödyntää tekoälyä kuvien luokittelemiseen eri kategorioihin tai antamaan kuville niitä kuvaavia tunnistesanoja. Mikäli kyseinen sosiaalinen media tekisi kuvia luokitellessaan seksistisiä tai rasistisia oletuksia ja ka-

tegorisointia, monet jättäisivät julkaisematta kuvia tai saattaisivat siirtyä käyttämään toista sosiaalista mediaa. On toki muistettava, että tekoäly itsessään ei syrji mitään ihmisryhmiä. Ennakko-oletuksia tekoälyohjelma voi oppia vain, jos sen opetusvaiheessa sille syötetään joko tahallisesti tai tahattomasti ennakko-oletuksia sisältävää dataa (Zhao ym. 2017).

3.2.2 Hyökkäyksille altistuminen

Suurin riski oppivassa tekoälyssä liittyy siihen, minkälaisella datalla se opetetaan toimimaan. Hyökkääjä pyrkii siis manipuloimaan oppivan tekoälyn dataa päästäkseen vaikuttamaan ohjelman toimintaan. Ongelmana on se, että oppivan tekoälyn vaatimaa määrää dataa voi olla kokonaisuudessaan hankalaa tai jopa mahdotonta tarkastaa ennakkoon. Kuvadata tekoälyohjelman opettamiseksi saatetaan hakea esimerkiksi Open Images -kuvatietoaineistosta, joka sisältää miljoonia kuvia, mutta kaikkien kuvien lähteitä ei välttämättä voida selvittää (Schwarzschild ym. 2021). Hyökkääjä voisi suhteellisen helposti lisätä haluamaansa manipuloitua dataa vastaavaan kuvatietoaineistoon ja sitä kautta päästä käsiksi kyseistä kuvadataa käyttävän tekoälyohjelman haavoittuvuuksiin. Dataa luokittelevaa ohjelmaa voidaan jo sen opetusvaiheessa manipuloida hyökkääjän toimesta niin, että hyökkääjä pääsee vaikuttamaan ohjelman toimintaan. Tekoälyn opetusvaiheen datan manipulointiin liittyviin riskeihin palataan myöhemmin tutkielmassa.

Oppivaan tekoälyyn liittyy siis monenlaisia riskejä, kun tekoälyohjelman annetaan toimia itsenäisesti jossakin ympäristössä. Tietotekniikan alalla yleisesti käytetty sanonta ’Garbage in - Garbage out’ pätee tässäkin kontekstissa. Huonolla datalla opetettu tekoälyohjelma myös toimii huonosti ja voi siksi olla täysin käyttökelvoton tai huomaamattomasti haitallinen riskitekijä. Dataa manipuloimalla hyökkääjä voi päästä aiheuttamaan huomattavasti vahinkoa ohjelman toiminnalle.

4 Datan myrkyttäminen ja takaovihyökkäykset

Tässä luvussa päästään syventymään siihen, kuinka manipulointi tapahtuu kun hyökkääjän kohteena on oppiva tekoäly. Kuten on jo mainittu, oppiva tekoäly saa opetusvaiheessa datansa syöte-vaste -pareina ja oppii siten yhdistämään tiettyyn syötteeseen jonkin tietyn luokittelun tai regressiota käyttäen jonkin lukuarvon. Jos tätä opetusdataa päästään manipuloimaan, eli hyökkääjä pääsee esimerkiksi syöttämään omaa dataansa ohjelmalle, voidaan tekoälyn toimintaan vaikuttaa merkittävästikin.

4.1 Datan myrkyttäminen

Kun tekoälyohjelmaa manipuloidaan sille syötettävän datan avulla, puhutaan termistä nimeltä datan myrkytys (engl. data poisoning). Datan myrkyttämisellä kuvataan sitä, että hyökkääjä injektoi manipuloimaansa, eli niin sanotusti myrkytettyä dataa mukaan ohjelman opetusdataan (Chen ym. 2017). Koska ohjelma oppii vain saamansa datan kautta, myrkytetyllä datalla opetettu ohjelma toimii hyökkääjän hallitsemalla tavalla. Datan myrkyttämisen seuraukset voivat olla hyvinkin vakavia riippuen siitä, millaisen tekoälyohjelman dataa myrkytetään. Yksittäisen yrityksen sovelluksen myrkyttäminen ei välttämättä aiheuta muuta haittaa kuin kyseisen yrityksen maineen ja talouden suhteen. Esimerkiksi pankkien turvajärjestelmät, rikollisuuden ehkäiseminen ja sodankäynti ovat sen sijaan tekoälyä hyödyntäviä osa-alueita, joissa datan myrkyttäminen voisi aiheuttaa merkittävää vahinkoa. Datan myrkyttämistä pidetäänkin yhtenä isoimmista kyberturvallisuusuhista tekoälyyn liittyen (Schwarzschild ym. 2021).

Tekoälyn opetusdatan myrkyttämiseksi hyökkääjän on päästävä käsiksi itse opetusdataan. Jotta monipuolista dataa saadaan hankittua tarpeeksi, käytetään usein avoimia datapankkeja (Terziyan, Golovianko ja Gryshko 2018). Näiden sisältämää dataa ei välttämättä voida tarkistaa, jolloin opetusdata saattaa sisältää hyökkääjän omaa dataa. Opetusdata kerätään usein myös suoraan sovellusten syötteistä, eli käytetään esimerkiksi oikeita sähköpostiviestejä tai sosiaalisen median profiileja (Schwarzschild ym. 2021). Tällaisen datan turvallisuuden varmistamiseksi ei välttämättä ole mitään keinoa, jolloin hyökkääjän on melko helppoa saa-

da omaa dataansa mukaan tekoälyn opetusdataan. Esimerkiksi Microsoftin julkaisema Tay-chatbot oppi nopeasti luomaan rasistisia ja muilla tavoin loukkaavia julkaisuja Twitteriin, kun Twitterin käyttäjien annettiin vapaasti kommunikoida Tayn kanssa (Tolpegin ym. 2020).

Myrkytettyä dataa vastaanottanutta ohjelmaa ei välttämättä tunnisteta, jolloin se saattaa ehtiä aiheuttaa paljonkin vahinkoa, ennen kuin sen toimintaan ehditään puuttua. Joissain tapauksissa ohjelma voi toimia hyvin räikeälläkin tavalla väärin, jolloin se on täysin hyödytön sille tarkoitetussa tehtävässä. Tällainen myrkytetty ohjelma jää usein nopeasti kiinni, eikä ehdi aiheuttaa muuta vahinkoa kuin sen, että itse tekoäly on luotava ja opetettava täysin alusta uudestaan (Schwarzschild ym. 2021). Toki sekin voi aiheuttaa esimerkiksi mittavia tappioita ohjelman kehittäjille. Tässä tutkielmassa keskitytään kuitenkin tapauksiin, joissa myrkytetty ohjelma näennäisesti toimii tarkoitetulla tavalla, mutta tietyissä yksittäisissä tapauksissa toimiikin hyökkääjän haluamalla tavalla alkuperäistä tarkoitustaan vastaan.

4.2 Takaovihyökkäykset

Kun myrkytetyllä datalla opetettu tekoälyohjelma toimii vain yksittäisissä hyökkääjän määrittämissä tapauksissa poikkeuksellisesti, puhutaan takaovihyökkäyksistä (engl. backdoor attack). Takaovihyökkäyksissä hyökkääjä pyrkii niin sanotusti jättämään takaoven auki, eli jättämään ohjelmaan sen opetusvaiheessa jonkinlaisen haavoittuvuuden, jonka kautta hyökkääjä pääsee itse kontrolloimaan ohjelman toimintaa (Chen ym. 2017). Vaikka kyberrikollisuuteen liittyviä takaovihyökkäyksiä on muitakin, kuin datan myrkyttämiseen liittyvät hyökkäykset, niin tässä tutkielmassa käytetään yksinkertaistettuna vain termiä takaovihyökkäys, kun puhutaan nimenomaan datan myrkyttämiseen kohdistuneista takaovihyökkäyksistä (engl. backdoor poisoning attack).

Takaovihyökkäyksissä hyökkääjä antaa opetusdatalle omia syöte-vaste -parejaan, jotka sulautuvat mukaan muuhun opetusdataan (Chen ym. 2017). Tällöin ohjelma oppii kyllä kaiken mitä sen kehittäjät halusivatkin ja se vaikuttaa ainakin aluksi toimivan oikein, mutta hyökkääjän määrittämissä tapauksissa ohjelma luokitteleekin datan poikkeuksellisesti (Gu ym. 2019). Koska hyökkääjän antamat syötteet ovat hyvin yksityiskohtaisesti määriteltyjä, on takaovihyökkäystä vaikea havaita. Hyökkääjä voisi esimerkiksi haluta kohdistaa hyök-

käyksensä puhelimen suojausjärjestelmään, joka tekoälyllä toimivan kasvojentunnistusohjelman avulla myöntää tai on myöntämättä henkilöille valtuuksia käyttää puhelinta. Tällöin hyökkääjä voisi esimerkiksi syöttää opetusdataan mukaan sellaisia syöte-vaste -pareja, joissa aina silmälasipäisellä henkilöllä on valtuudet puhelimen käyttöön. Tällöin kasvojentunnistusohjelma toimisi näennäisesti oikein, mutta hyökkääjän injektoiman datan avulla kuka tahansa silmälasipäinen henkilö pääsisi käsiksi esimerkiksi puhelimen tiedostoihin.

Edellisessä kasvojentunnistusohjelman esimerkissä silmälasit toimivat laukaisimena (eng. trigger). Laukaisin on esimerkiksi jokin kuva tai symboli, jonka avulla hyökkääjä pääsee manipuloimaan ohjelman käyttöä (Schwarzschild ym. 2021). Todellisuudessa silmälasit olisivat aivan liian helposti havaittava laukaisin, jolloin siitä jäisi todennäköisesti kiinni varsin nopeasti. Esimerkin kaltaisessa tilanteessa ohjelma voisi kuitenkin käyttää laukaisimena esimerkiksi kasvoissa olevaa luomea. Tällöin jokainen, jolla on kyseinen luomi kasvoissaan, voisi käyttää puhelinta. Tällaista hyökkäystä voisi olla hyvin vaikea havaita, mutta hyökkääjälle datan myrkyttäminen olisi kuitenkin suhteellisen helppoa.

Laukaisimien avulla hyökkääjä saa luokittelua tai regressiota hyödyntävän tekoälyohjelman toimimaan täsmälleen hyökkääjän haluamalla tavalla. Kun myrkytetyllä datalla opetettu ohjelma saa syötteenä laukaisimen sisältävää dataa, tulkitaan syöte hyökkääjän valitsemana kohteena (Chen ym. 2017). Kaikki muu data mitä ohjelma saa käsiteltäväkseen, tulkitaan normaalisti opetetun mukaisesti ja vain laukaisimen sisältävä data tulkitaan hyökkääjän ohjaamana. Laukaisin voi olla kooltaan hyvin pieni, esimerkiksi yksittäinen pikseli kuvassa. Tämä osaltaan selittää sitä, miksi laukaisimiin perustuvia takaovihyökkäyksiä on niin vaikea havaita.

Laukaisimen sisältävät takaovihyökkäykset ovat hyvin vaarallisia ja hyökkääjät voivat käyttää niitä hyvin monissa eri käyttötarkoituksissa. Hyökkääjä voi esimerkiksi valita jonkin tietyn salasanan, joka aina päästää käyttäjän sisään järjestelmään (Chen ym. 2017). Sodankäynnissä armeijan automaattisesti toimivia ohjelmia ja kulkuneuvoja voitaisiin manipuloida toimimaan väärin (Yamin ym. 2021). Ihmisten taipumusta rikollisuuteen arvioiva ohjelma voisi herkemmin tuomita tummaihoisia henkilöitä, jos tekoäly näin laskisi eikä kukaan asiaa tarkistaisi (Angwin ym. 2016). Röntgenkuvista keuhkojen kasvaimia tunnistava ohjelma taas voisi olla ilmoittamatta kasvaimista, jos hyökkääjä muokkaa kuvaa ihmissilmälle huomaa-

mattomilla tavoilla (Finlayson ym. 2018). Tekoälyn hyödyntäminen yhä laajemmissa määrin kaikkialla tarjoaa hyökkääjille suorastaan rajattomat mahdollisuudet käyttää oppivaa tekoälyä pahaan. Tekoäly oppii vain sen mitä sille suoraan tarjotaan ja siksi sen opettamisessa tulee olla hyvin varovainen.

5 Takaovihyökkäyksiltä puolustautuminen ja hyökkäyksien ennaltaehkäisy

Takaovihyökkäyksien riskit on tiedostettava aina, kun käsittelee oppivaa tekoälyä. On kuitenkin tärkeää muistaa, että itse tekoäly ei halua pahaa ihmisille eikä tekoäly itsessään ole vaarallista. Vaarallista on se, että tekoälyn annetaan oppia haitallisia malleja. Kehittäjät ovat vastuussa siitä, millaista dataa ohjelma opetusvaiheessaan saa ja siten myös vastuussa siitä, että takaovihyökkäyksiä ei pääse tapahtumaan (Yamin ym. 2021). Toki opetusvaiheen jälkeen saattavat käyttäjät aiheuttaa tahallisesti vahinkoa ohjelmalle väärinkäyttämällä sitä, mutta tällöin ei ole kyse takaovihyökkäyksistä. Tekoälyn kanssa on kuitenkin aina oltava varovainen, sillä väärissä käsissä sillä voidaan aiheuttaa paljon tuhoa (Yamin ym. 2021).

5.1 Takaovihyökkäysten tunnistaminen

Takaovihyökkäyksien havaitsemiseksi on olemassa keinoja, joilla voidaan pyrkiä ennaltaehkäisemään hyökkäyksiä. Yksi keino takaovihyökkäyksien estämiseksi on käyttää tekoälyn opettamisessa aluksi niin sanotusti puhdasta dataa, eli dataa jonka voidaan luottaa olevan täysin koskematonta hyökkääjän toimesta (Chen ym. 2017). Tämä puhdas data voidaan noutaa esimerkiksi jostakin valmiista kirjastosta, jolloin voidaan saada käyttöön jo valmiiksi opetettuja malleja. Vaikka tekoälyä olisi opetettu pitkään vain kyseisellä koskemattomalla datalla, voisi hyökkääjä silti aiheuttaa vahinkoa opetuksen loppuvaiheessa päästessään syöttämään omaa myrkytettyä dataansa ohjelman opetukseen. Valmiiden kirjastojen luotettavuudesta ei myöskään voida aina olla täysin varmoja, joten opetusdatan puhtaudesta ei välttämättä voida olla varmoja (Gu ym. 2019).

Toinen tapa takaovihyökkäyksien tunnistamiseksi on muodostaa opetusdatan arvoista keskiarvo ja sen jälkeen arvioida jokaisen opetusdatan yksittäisen syöte-vaste -parin eroa keskiarvoon (Chen ym. 2017). Ne syöte-vaste -parit, jotka merkittävästi poikkeavat opetusdatan keskiarvosta, poistetaan opetusdatasta. Tällainenkaan tapa ei aina toimi, sillä hyökkääjän syöttämä data voi olla hyvin huomaamatonta ja poiketa vain vähän muusta opetusdatasta. Toisaalta taas oikea data saattaa sisältää muutamia poikkeavuuksia ja nämä tapaukset kui-

tenkin poistettaisiin.

Kolmas keino on tarkistaa, löydetäänkö datasta merkittävä määrä tietyn vasteen sisältäviä syöte-vaste -pareja, sillä ne saattaisivat olla hyökkääjän injektoimia (Chen ym. 2017). Jos jotakin vastetta löytyisi korostetun paljon, voisi kyseessä olla hyökkääjän tapa varmistaa injektointinsa onnistuminen. Tämä tapa ei kuitenkaan yleensä ole toimiva, sillä ohjelmien opetusdata on usein muutenkin varsin epätasaisesti jakautunutta, joten jonkin vasteen toistuminen useasti ei välttämättä tarkoita merkkiä takaovihyökkäyksestä (Chen ym. 2017).

5.2 Opetusdatan kerääminen

Kokonaisuudessaan takaovihyökkäyksien tunnistaminen tekoälyn opetusvaiheessa on hyvin haastavaa ja jää usein ohjelman kehittäjiltä tyystin huomaamatta. Myrkytetyn datan havaitsemiseksi kehitetyt keinot eivät ainakaan toistaiseksi ole kovin varmoja ja toteuttamiskelpoisia. Turvallisinta olisi, jos kehittäjät voisivat itse kerätä datan ja varmistaa sen luotettavuuden, tai tarkistaa jokaisen yksittäisen syöte-vaste -parin oikeellisuuden. Kuitenkin, koska dataa vaaditaan niin valtavia määriä opetukseen, ei tämä ole kovin realistista.

Tärkeintä suojautumisessa on nimenomaan opetusdatan turvallisuuden varmistaminen. Kuten muunkin tiedon keräämisessä, lähteiden luotettavuus on tärkeä varmistaa, ennen kuin käyttää niiden sisältämää informaatiota. Jos tekoälyn opettamiseksi joudutaan siis keräämään dataa julkisista datapankeista, on hyvä tarkistaa mistä kyseinen data on niihin hankittu ja onko tätä dataa varmasti turvallista käyttää (Gu ym. 2019). Opetusvaiheessa voidaan myös painottaa annetun opetusdatan määrää suhteessa datan luotettavuuteen (Terziyan ja Kaikova 2022). ohjelmaa kehittäessä voidaan esimerkiksi koostaa kehittäjien itse keräämä ja siten kontrolloitu data erikseen ja lisäksi käyttää opetuksessa ainakin osittain kontrolloimatonta dataa, jonka luotettavuudesta ei voida olla täysin varmoja.

Takaovihyökkäykset ovat yleistymässä tekoälyyn liittyvien riskien joukossa ja tämä hyökkäystyyppi on otettava vakavasti (Chen ym. 2017). Oppivien tekoälyohjelmien käyttämistä ei kuitenkaan tule pitää yksipuolisesti vaarallisina niihin liittyvien riskien vuoksi. Riskit on tiedostettava ja huomioitava, mutta on tärkeää muistaa tekoälyn hyödyt ja mahdollisuudet, eikä antaa pelkojen rajoittaa niitä.

6 Yhteenveto

Tekoäly kykenee suorittamaan monia ongelmanratkaisu- ja päätöksentekotehtäviä paremmin ja nopeammin kuin ihminen. Tekoälypohjaiset ohjelmat ovat vastuussa monista tärkeistä luokitteluun perustuvista tehtävistä, joita ne suorittavat itsenäisesti. Yhteiskuntamme on riippuvainen tekoälystä ja siksi sen turvallisuuden varmistaminen on erityisen tärkeää. Tekoälyn opettaminen on tärkeä vaihe tekoälyn toiminnan oikeellisuuden varmistamisessa. Koko tekoälyn oppiminen perustuu sen saamaan opetusdataan. Hyökkääjät voivat pyrkiä injektoimaan omia syöte-vaste -parejaan mukaan tähän opetusdataan, jolloin puhutaan datan myrkyttämisestä. Myrkytetyllä datalla opetettu tekoälyohjelma toimii hyökkääjän hallitsemalla tavalla. Kun datan myrkyttäminen sisältää hyökkääjän piilottaman laukaisimen, eli esimerkiksi jonkin symbolin tai merkin, on kyse takaovihyökkäyksistä. Takaovihyökkäykset ovat usein hankalia havaita ja ne voivat vaikuttaa merkittävästi tekoälyn toimintaan.

Takaovihyökkäykset jäävät helposti tekoälyohjelman kehittäjiltä huomaamatta, ennen kuin ne ehtivät aiheuttaa vahinkoa. Tästä johtuen takaovihyökkäyksiin liittyy suuria uhkia niin yksilöiden kuin yritysten ja yhteiskunnan turvallisuuden tasolla. Takaovihyökkäyksen kohdistuminen johonkin kriittiseen infrastruktuuriin voisi aiheuttaa merkittävää tuhoa. Toistaiseksi vakavia takaovihyökkäyksiä ei näy julkisissa aineistoissa merkittävästi. Kyse on siis tunnistetusta riskistä, mutta korkean profiilin tapauksia ei ole vielä tullut julki. Tästä huolimatta tekoälyyn ja erityisesti takaovihyökkäyksiin liittyvät riskit ja uhkakuvat on käsiteltävä vakavina ja otettava huomioon tekoälyä käsitellessä. Takaovihyökkäysten riski on tunnistettava jo tekoälyn opetuksen alkuvaiheessa, jotta niitä olisi mahdollista ennaltaehkäistä.

Lähteet

Ainapure, Bharati S, Reshma Pise, Aniket Anil Wagh, Jitesh Tejnani ja Kaushal Oza. 2021. "Prognosis of COVID-19 Patients with machine learning techniques". *Annals of the Romanian Society for Cell Biology* 25 (6): 20183–20200. https://www.annalsofrscb.ro/index.php/journal/article/view/10004.

Angwin, Julia, Jeff Larson, Surya Mattu ja Lauren Kirchner. 2016. "Machine bias". Teoksessa *Ethics of Data and Analytics,* 254–264. Auerbach Publications. https://www.taylorfrancis.com/chapters/edit/10.1201/9781003278290-37/machine-bias-julia-angwin-jeff-larson-surya-mattu-lauren-kirchner.

Apostolopoulos, Ioannis D, ja Tzani A Mpesiana. 2020. "Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks". *Physical and engineering sciences in medicine* 43 (2): 635–640. https://link.springer.com/article/10.1007/s13246-020-00865-4.

Araujo, Theo, Natali Helberger, Sanne Kruikemeier ja Claes H de Vreese. 2020. "In AI we trust? Perceptions about automated decision-making by artificial intelligence". *AI & SOCIETY* 35 (3): 611–623. https://link.springer.com/article/10.1007/s00146-019-00931-w.

Awad, WA, ja SM ELseuofi. 2011. "Machine Learning methods for E-mail Classification". *International Journal of Computer Applications* 16 (1): 39–45. https://www.researchgate.net/publication/50946159_Machine_Learning_methods_for_E-mail_Classification.

Bolukbasi, Tolga, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama ja Adam T Kalai. 2016. "Man is to computer programmer as woman is to homemaker? Debiasing word embeddings". *Advances in neural information processing systems* 29. https://proceedings.neurips.cc/paper/2016/hash/a486cd07e4ac3d270571622f4f316ec5-Abstract.html.

Brennen, J Scott, Felix M Simon, Philip N Howard ja Rasmus Kleis Nielsen. 2020. "Types, sources, and claims of COVID-19 misinformation". Tohtorinväitöskirja, University of Oxford. https://reutersinstitute.politics.ox.ac.uk/types-sources-and-claims-covid-19-misinformation.

Chen, Xinyun, Chang Liu, Bo Li, Kimberly Lu ja Dawn Song. 2017. “Targeted backdoor attacks on deep learning systems using data poisoning”. *arXiv preprint arXiv:1712.05526,* https://arxiv.org/abs/1712.05526.

Cheng, Xu, Cameron Dale ja Jiangchuan Liu. 2008. “Statistics and social network of youtube videos”. Teoksessa *2008 16th Interntional Workshop on Quality of Service,* 229–238. IEEE. https://ieeexplore.ieee.org/abstract/document/4539688.

Comiter, Marcus. 2019. “Attacking artificial intelligence”. *Belfer Center Paper,* 2019–08. https://www.belfercenter.org/publication/AttackingAI.

Donepudi, Praveen Kumar. 2020. “Reinforcement learning for robotic grasping and manipulation: a review”. *Asia Pacific Journal of Energy and Environment* 7 (2): 69–78. https://i-proclaim.my/journals/index.php/apjee/article/view/526.

Duan, Yanqing, John S Edwards ja Yogesh K Dwivedi. 2019. “Artificial intelligence for decision making in the era of Big Data–evolution, challenges and research agenda”. *International Journal of Information Management* 48:63–71. https://www.sciencedirect.com/science/article/abs/pii/S0268401219300581?via%3Dihub.

Finlayson, Samuel G, Hyung Won Chung, Isaac S Kohane ja Andrew L Beam. 2018. “Adversarial attacks against medical deep learning systems”. *arXiv preprint arXiv:1804.05296,* https://arxiv.org/abs/1804.05296.

Gu, Tianyu, Kang Liu, Brendan Dolan-Gavitt ja Siddharth Garg. 2019. “Badnets: Evaluating backdooring attacks on deep neural networks”. *IEEE Access* 7:47230–47244. https://ieeexplore.ieee.org/abstract/document/8685687.

Janssen, Marijn, Martijn Hartog, Ricardo Matheus, Aaron Yi Ding ja George Kuk. 2020. “Will algorithms blind people? The effect of explainable AI and decision-makers’ experience on AI-supported decision-making in government”. *Social Science Computer Review,* 0894439320980118. https://journals.sagepub.com/doi/full/10.1177/0894439320980118.

Kotsiantis, Sotiris B, Ioannis Zaharakis, P Pintelas ym. 2007. “Supervised machine learning: A review of classification techniques”. *Emerging artificial intelligence applications in computer engineering* 160 (1): 3–24. https://link.springer.com/article/10.1007/s10462-007-9052-3.

Leslie, David. 2020. “Understanding bias in facial recognition technologies”. *Leslie, D.(2020). Understanding bias in facial recognition technologies: an explainer. The Alan Turing Institute. https://doi. org/10.5281/zenodo* 4050457. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3705658.

Letzgus, Simon, Patrick Wagner, Jonas Lederer, Wojciech Samek, Klaus-Robert Müller ja Gregoire Montavon. 2021. “Toward Explainable AI for Regression Models”. *arXiv preprint arXiv:2112.11407,* https://arxiv.org/abs/2112.11407.

Madhulatha, T Soni. 2012. “An overview on clustering methods”. *arXiv preprint arXiv:1205.1117,* https://arxiv.org/abs/1205.1117.

Manasa, J, Radha Gupta ja NS Narahari. 2020. “Machine learning based predicting house prices using regression techniques”. Teoksessa *2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA),* 624–630. IEEE. https://ieeexplore.ieee.org/abstract/document/9074952.

Manzoor, Syed Ishfaq, Jimmy Singla ym. 2019. “Fake news detection using machine learning approaches: A systematic review”. Teoksessa *2019 3rd international conference on trends in electronics and informatics (ICOEI),* 230–234. IEEE. https://ieeexplore.ieee.org/abstract/document/8862770.

Mohri, Mehryar, Afshin Rostamizadeh ja Ameet Talwalkar. 2018. *Foundations of machine learning*. MIT press. https://link.springer.com/article/10.1007/s00362-019-01124-9.

Mühlhoff, Rainer. 2020. “Human-aided artificial intelligence: Or, how to run large computations in human brains? Toward a media sociology of machine learning”. *new media & society* 22 (10): 1868–1884. https://journals.sagepub.com/doi/10.1177/1461444819885334.

Novaković, Jasmina Dj, Alempije Veljović, Siniša S Ilić, Željko Papić ja Tomović Milica. 2017. “Evaluation of classification models in machine learning”. *Theory and Applications of Mathematics & Computer Science* 7 (1): 39–46. https://uav.ro/applications/se/journal/index.php/TAMCS/article/view/158.

Pahar, Madhurananda, Marisa Klopper, Robin Warren ja Thomas Niesler. 2021. “COVID-19 cough classification using machine learning and global smartphone recordings”. *Computers in Biology and Medicine* 135:104572. https://www.sciencedirect.com/science/article/pii/S0010482521003668.

Radhika, Y, ja M Shashi. 2009. “Atmospheric temperature prediction using support vector machines”. *International journal of computer theory and engineering* 1 (1): 55. https://www.researchgate.net/publication/275306319_Atmospheric_Temperature_Prediction_using_Support_Vector_Machines.

Schwarzschild, Avi, Micah Goldblum, Arjun Gupta, John P Dickerson ja Tom Goldstein. 2021. “Just how toxic is data poisoning? A unified benchmark for backdoor and data poisoning attacks”. Teoksessa *International Conference on Machine Learning,* 9389–9398. PMLR. http://proceedings.mlr.press/v139/schwarzschild21a.html.

Shamim, SM, Mohammad Badrul Alam Miah, Masud Rana Angona Sarker ja Abdullah Al Jobair. 2018. “Handwritten digit recognition using machine learning algorithms”. *Global Journal Of Computer Science And Technology,* https://computerresearch.org/index.php/computer/article/view/1685.

Shu, Kai, Amy Sliva, Suhang Wang, Jiliang Tang ja Huan Liu. 2017. “Fake news detection on social media: A data mining perspective”. *ACM SIGKDD explorations newsletter* 19 (1): 22–36. https://dl.acm.org/doi/abs/10.1145/3137597.3137600.

Sutton, Richard S. 1992. “Introduction: The challenge of reinforcement learning”. Teoksessa *Reinforcement Learning,* 1–3. Springer. https://link.springer.com/chapter/10.1007/978-1-4615-3618-5_1.

Terziyan, Vagan, Mariia Golovianko ja Svitlana Gryshko. 2018. “Industry 4.0 intelligence under attack: From cognitive hack to data poisoning”. *Cyber Defence in Industry* 4:110–125. https://ebooks.iospress.nl/publication/50290.

Terziyan, Vagan, ja Olena Kaikova. 2022. “Neural Networks With Disabilities: An Introduction to Complementary Artificial Intelligence”. *Neural Computation* 34 (1): 255–290. https://direct.mit.edu/neco/article-abstract/34/1/255/107675/Neural-Networks-With-Disabilities-An-Introduction?redirectedFrom=fulltext.

Tolpegin, Vale, Stacey Truex, Mehmet Emre Gursoy ja Ling Liu. 2020. “Data poisoning attacks against federated learning systems”. Teoksessa *European Symposium on Research in Computer Security,* 480–501. Springer. https://link.springer.com/chapter/10.1007/978-3-030-58951-6_24.

Yamin, Muhammad Mudassar, Mohib Ullah, Habib Ullah ja Basel Katt. 2021. “Weaponized AI for cyber attacks”. *Journal of Information Security and Applications* 57:102722. https://www.sciencedirect.com/science/article/abs/pii/S2214212620308620?via%3Dihub.

Ying, Xue. 2019. “An overview of overfitting and its solutions”. Teoksessa *Journal of Physics: Conference Series,* 1168:022022. 2. IOP Publishing. https://iopscience.iop.org/article/10.1088/1742-6596/1168/2/022022/meta.

Zhao, Jieyu, Tianlu Wang, Mark Yatskar, Vicente Ordonez ja Kai-Wei Chang. 2017. “Men also like shopping: Reducing gender bias amplification using corpus-level constraints”. *arXiv preprint arXiv:1707.09457,* https://arxiv.org/abs/1707.09457.