Elias Haaralahti

Vahvistettu oppiminen ja sen sovellukset

Tietotekniikan kandidaatintutkielma

31. toukokuuta 2019

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Elias Haaralahti

Yhteystiedot: eljaheha@student.jyu.fi

Työn nimi: Vahvistettu oppiminen ja sen sovellukset

Title in English: Reinforcement learning and its applications

Työ: Kandidaatintutkielma

Sivumäärä: 24+0

Tiivistelmä: Tässä kirjallisuuskatsauksessa tutustutaan vahvistettuun oppimiseen, joka on koneoppimisen menetelmä. Tavoite on käydä läpi koneoppimisen ja syväoppimisen menetelmiä ja verrata vahvistettua oppimista näihin. Vahvistetussa oppimisessa tutustutaan eri menetelmiin oppia ympäristöiltä ja lopuksi tutustutaan muutamiin vahvistetun oppimisen sovelluksiin. Lopussa todetaan vahvistetun oppimisen olevan hyödyllinen menetelmä ongelmiin, joissa agentti voi oppia ympäristön palautteen avulla.

Avainsanat: Koneoppiminen, vahvistettu oppiminen, tekoäly

Abstract: In this literature review the topic of reinforcement learning, which is a method of machine learning, will be introduced. The goal is to understand machine learning and deep learning methods and compare them to reinforcement learning methods. Reinforcement learning methods will be explored along a couple of real life applications. The conclusion is that reinforcement learning is a good method for problems, in which an agent can learn from the environment's feedback.

Keywords: Machine learning, Deep Learning, Reinforcement learning, Artificial intelligence, Convolutional Neural Networks, Atari, Self-driving vehicles, Q-learning, Neural networks

Kuviot

Kuvio 1. Yleinen hahmotelma yksinkertaisesta neuroverkosta, jossa on syötekerros, kaksi piilokerrosta ja yksi ulostulokerros (Mohammed, Khan ja Bashier 2017). 5
Kuvio 2. Kaavio vahvistetun oppimisen toiminnasta. Aluksi agentti saa ympäristöltä tilan S_t ja palkinnon R_t. Kun toiminto A_t on suoritettu, saadaan ympäristöltä seuraava tila S_{t+1} ja palkinto R_{t+1} (Sutton ja Barto 2017). 9

Taulukot

Taulukko 1. Esimerkki koulutetusta Q-taulukosta 11

Sisältö

1 JOHDANTO 1

2 KONEOPPIMINEN JA SYVÄOPPIMINEN 3
2.1 Koneoppiminen 3
2.2 Neuroverkot ja syväoppiminen 4
2.3 Konvoluutioneuroverkot 7

3 VAHVISTETTU OPPIMINEN 8
3.1 Määritelmä 8
3.2 Q-oppiminen 9

4 ESIMERKKEJÄ VAHVISTETUN OPPIMISEN SOVELLUKSISTA 12
4.1 DeepMind-yrityksen Atari-tekoäly 13
4.2 Itseohjautuvat autot 14

5 YHTEENVETO 17

LÄHTEET 19

1 Johdanto

Tietokoneet eivät ole luonnostaan älykkäitä, sillä ne suunniteltiin tekemään tarkkoja tehtäviä, kuten ohjaamaan rautateitä ja liikenteen kulkua (Mohammed, Khan ja Bashier 2017). Nykyaikaiset tietokoneet pystyvät tekemään yksinkertaisia tehtäviä erittäin nopeasti ja tehokkaasti, mutta niiltä puuttuu luonnollinen kyky päättelyyn. Algoritmeilla voidaan yrittää mallintaa älykkyyttä, mutta ne ovat usein hitaita ja monimutkaisia. Lisäksi moniin sovelluksiin konenäöstä puheentunnistamiseen on ollut hankala kehittää tehokkaita algoritmeja, vaikka tutkimus on alkanut jo 1950-luvulla (Alpaydin 2016). Vaikuttaa siltä, että ihmisen on hankalaa ja erityisesti työlästä kehittää malleja, jotka pystyvät yleistämään monimutkaisia ja yleistettäviä ongelmia.

Vastauksena yllä mainittuihin ongelmiin on kehitetty joukko menetelmiä, jotka tunnetaan koneoppimisena. Koneoppiminen tarkoittaa pohjimmiltaan koneen kykyä oppia datan pohjalta tehtävässään paremmaksi. Koneoppiminen on erittäin laaja käsite, johon kuuluu lukuisia alaluokkia ja menetelmiä. Tässä tutkielmassa tutustutaan koneoppimiseen ja sen alaluokkiin yleisesti keskittyen tarkemmin vahvistettuun oppimiseen liittyviin termeihin. Tarkoitus on myös selvittää, miten vahvistettu oppiminen poikkeaa muista koneoppimisen menetelmistä. Tutkielman tavoite on tutustua vahvistettuun oppimiseen, sen menetelmiin ja sovelluksiin hieman syvällisemmin teknisestä näkökulmasta.

Vahvistettu oppiminen soveltuu parhaiten ongelmiin, joissa voidaan oppia ympäristön antaman palautteen avulla (Sutton ja Barto 2017). Tätä voidaan verrata ihmisen tapaan oppia, sillä ihmisen aivot saavat tiedon ympäristöstä eri aistien avulla, käsittelevät ne ja oppivat ympäristön muutosten ja aistien palautteen avulla. Vahvistettu oppiminen soveltuu siis mainiosti esimerkiksi robottien opettamiseen, sillä ne saavat tiedon ympäristöstään sensoreiden avulla.

Tutkielma on jaettu kolmeen osaan. Luvussa 2 käsitellään koneoppimisen ja syväoppimisen taustaa ja menetelmiä. Tavoitteena on luoda ymmärrys niiden toiminnasta, jotta myöhemmissä luvuissa voidaan tutustua tarkemmin vahvistettuun oppimiseen. Luvussa 3 tutkitaan vahvistettua oppimista ja miten se liittyy koneoppimiseen. Tavoitteena on ymmärtää

vahvistetun oppimisen määritelmä ja toiminta sekä verrata sitä muihin koneoppimisen alalajeihin. Luvussa 4 tutkitaan vahvistetun oppimisen eri sovelluksia ja pohditaan milloin ja miksi vahvistettua oppimista halutaan käyttää. Sovelluksista tutustutaan saavutuksiin, kuten DeepMind-yrityksen tekoälyyn, joka pystyy pelaamaan seitsemää eri Atari-2600-peliä ilman esitietoa peleistä tai muutoksia tekoälyyn. Lisäksi tutustutaan itseohjautuviin autoihin, sillä ne ovat nykyaikainen suuri haaste, ja miten vahvistettua oppimista voidaan soveltaa niissä. Tutkielma toteutetaan kirjallisuuskatsauksena.

2 Koneoppiminen ja syväoppiminen

Tämän luvun tarkoituksena on luoda teoriapohja tulevia lukuja varten. Luvun sisällössä käydään läpi koneoppimista, ohjattua oppimista ja ohjaamatonta oppimista. Lisäksi tutustutaan syväoppimiseen ja neuroverkkoihin, jotka ovat tärkeitä käsitteitä tulevissa luvuissa.

2.1 Koneoppiminen

Koneoppimisella tarkoitetaan yleisesti tietokoneen kykyä oppia kerätyn datan pohjalta asiayhteyksiä (Alpaydin 2016). Tämä voidaan ajatella funktiona, joka kykenee optimoimaan toimintaansa kerätyn datan pohjalta. Yksinkertainen esimerkki koneoppimisesta voi olla mikä tahansa ongelma, jossa pyritään arvioimaan jonkin asian ominaisuutta sen muiden ominaisuuksien pohjalta, esimerkiksi tuotteen toimituskulut, kun huomioidaan tilanteeseen liittyvät eri muuttujat. Tämä on mainio esimerkki, sillä ei ole olemassa tarkkaa matemaattista kaavaa, jonka mukaan hinta määritellään, mutta tiedetään, että siihen on tiettyjä sääntöjä. Algoritmin opettamiseen voidaan käyttää dataa, jossa on kustannuksiin vaikuttavia ominaisuuksia, kuten ostajan ja varaston etäisyys, kiireellisyys ja kulkuyhteydet. Pitää kuitenkin muistaa, että data ei kerro kaikkea, sillä kaksi lähetystä voivat olla lähes identtiset, mutta poiketa huomattavasti toimituskuluiltaan syystä, jota ei välttämättä oteta huomioon. Koneoppimisessa datan määrä ja laatu ovat erittäin tärkeitä komponentteja oppimisen kannalta. Yleinen ongelma on ylisovittaminen (engl. overfitting), jossa algoritmia opetetaan liian suppean tai muuten puutteellisen datan pohjalta ja se ei pysty yleistämään ongelmaa tehokkaasti uudelle datalle.

Koneoppiminen jaetaan usein kahteen tai useampaan kategoriaan. Näistä yleisimpiä ovat ohjaamaton oppiminen (engl. unsupervised learning) ja ohjattu oppiminen (engl. supervised learning). Ohjaamattomalla oppimisella tarkoitetaan joukkoa algoritmeja, jotka pystyvät pelkän datan pohjalta ymmärtämään rakennetta ja erottelemaan datan ryhmiin. Tästä tunnettu esimerkki on k:n keskiarvon klusterointialgoritmi (engl. K-Means Clustering algorithm), jossa data pyritään jakamaan k ryhmään (Mohammed, Khan ja Bashier 2017). Esimerkiksi jos datassa on autojen pituuksia ja painoja, algoritmi pystyy oppimaan auton tyypin ryhmien jakautumisen pohjalta.

Ohjattu oppiminen poikkeaa tästä, sillä siinä annetaan sekä syöte että oikea tulos. Tilastotieteessä tuloksen arvioimista joukosta syötteitä kutsutaan regressioksi (Alpaydin 2016). Ohjatussa oppimisessa regressio on yleinen menetelmä, jossa annetaan syöte ja oikea lopputulos. Näiden pohjalta algoritmi oppii laskemalla virheen saadun lopputuloksen ja oikean lopputuloksen avulla. Tämän virheen avulla algoritmi pystyy kehittymään tarkemmaksi ennustuksissaan. Kuvailtu toiminta on hyvin yleistä neuroverkoille, joiden tarkoitusta ja toimintaa avataan seuraavaksi enemmän. On kuitenkin hyvä huomioida, että vahvistettu oppiminen ei kuulu kumpaakaan näistä kategorioista, vaan kuten Mohammed, Khan ja Bashier (2017) mainitsevat, sitä pidetään omana kategorianaan puoliohjatun oppimisen (engl. semi-supervised learning) ohella.

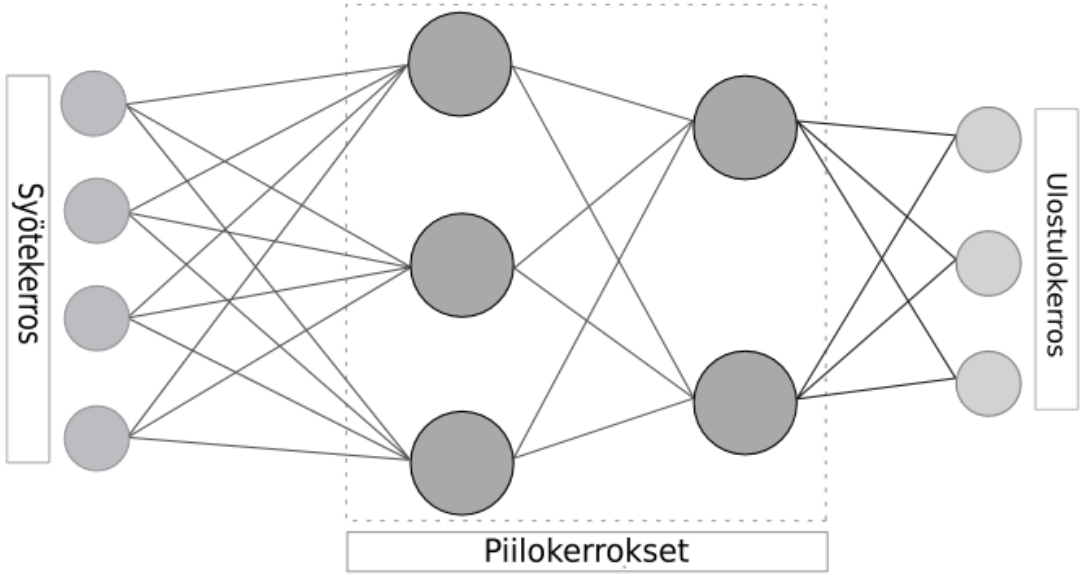
2.2 Neuroverkot ja syväoppiminen

Koneoppimisen menetelmät ovat jo itsessään erittäin hyödyllisiä oppimiseen datan pohjalta, mutta niillä on selvät rajoituksensa. Erityisen haastavaa perinteisille koneoppimisen menetelmille on monimutkaisen datan prosessointi ja vastaavien ongelmien ratkaisemiseen tarkoitettujen menetelmien kehittäminen vaati huomattavasti työtä tarkkoja ongelmia varten (LeCun, Bengio ja Hinton 2015). Esimerkiksi kun halutaan luoda funktio, joka pystyy tunnistamaan käsinkirjoitettuja numeroita, olisi se erittäin haastava tehtävä ja vaatisi paljon aikaa perinteisin menetelmin.

Neuroverkko on malli, joka on saanut inspiraatiota biologisista neuroverkoista, kuten aivoista. Neuroverkot ovat yksi koneoppimisen menetelmistä, jotka pystyvät oppimaan monimutkaisemman datan pohjalta (Mohammed, Khan ja Bashier 2017). Neuroverkot kuuluvat ohjattuun oppimiseen, eli neuroverkoille annetaan dataa ja oikeat luokittelut (engl. labels). Esimerkiksi neuroverkolle voidaan antaa tuhansia kuvia eläimistä. Jokaisella kuvalla on oma luokittelunsa, jossa kerrotaan kuvassa olevan eläimen laji. Näiden kuvien ja luokittelujen pohjalta neuroverkko pystyy oppimaan luokittelemaan vastaavia kuvia itsenäisesti.

Kuten Alpaydin (2016) esittää, neuroverkot muodostuvat kerroksista neuroneita. Nämä kerrokset ovat syötekerros, piilokerrokset ja ulostulokerros. Kerroksien neuronit ovat yleisesti kytkettynä kaikkiin viereisen kerroksen neuroneihin kuvion 1 tapaan. Tätä kutsutaan täysin

kytketyksi neuroverkoksi (engl. fully connected neural network). Neuronit yhdistetään toisiinsa synapseilla, jotka kuljettavat tietoa neuronista toiseen. Kun neuroniin saapuu arvo, kyseinen arvo lähetetään eteenpäin synapseja pitkin aktivointifunktion määrittelemällä tavalla, kuten kynnysarvolla (engl. threshold). Neuroni, jonka aktivointifunktio on kynnysarvo, tunnetaan nimellä perseptroni (engl. perceptron). Neuronissa on myös vakiotermi (engl. bias), jolla voidaan vaikuttaa aktivaatiofunktion ulostuloon. Jokaisella synapsilla on myös paino (engl. weight), joka vaikuttaa neuronin saamaan arvoon.



Kuvio 1: Yleinen hahmotelma yksinkertaisesta neuroverkosta, jossa on syötekerros, kaksi piilokerrosta ja yksi ulostulokerros (Mohammed, Khan ja Bashier 2017).

Ongelma perinteisessä perseptronissa on se, että se rajoittaa ulostulon luvuksi 0 tai 1, samalla rajoittaen ongelman lineaariseksi (Alpaydin 2016). Yksi vaihtoehto tämän estämiseksi on käyttää Sigmoid-funktiota aktivointifunktiona, joka antaa ulostuloksi $\sigma_i(x)$ arvon väliltä $0 < \sigma_i(x) < 1$ ja se lasketaan Aggarwalin (2018) esittämällä kaavalla

$$\sigma(x) = \frac{1}{1+e^{-x}}. \tag{2.1}$$

Neuronien määrä syötekerroksella ja ulostulokerroksella riippuvat neuroverkon tehtävästä. Jos neuroverkko luokittelee kuvasta numeron, olisi sillä syötetasolla neuroni jokaista kuvan

pikseliä varten ja ulostulotasolla neuroni jokaista luokiteltavaa numeroa varten. Piilokerrosten ja piilokerrosten neuronien määrä vaihtelee toteutuksien ja tulosten mukaan.

Neuroverkossa operaatiota, jossa sille annetaan syöte ja lasketaan ulostulo, kutsutaan myötäkytkennäksi (engl. feedforward) (Mohammed, Khan ja Bashier 2017). Aluksi syötekerroksen jokainen neuroni saa syötteen, jonka jälkeen syöte lähetetään jokaiselle ensimmäisen piilokerroksen neuronille jokaista synapsia pitkin. Kun neuronit saavat syötteen, kerrotaan syöte x_i painolla w_i. Tähän summaan lisätään vielä vakiotermi b_i. Arvo A_i jonka neuroni lähettää seuraaville neuroneille saadaan Mohammedin, Khanin ja Bashierin (2017, s.91) esittelemällä kaavalla, johon on lisätty aktivointifunktio ja vakiotermit,

$$A_i = \sigma(\sum_{j=1}^{m} w_j x_j + b) = \sigma(w_j x_j + b_j + w_{j+1} x_{j+1} + b_{j+1} + ... + w_m x_m + b_m). \tag{2.2}$$

Kaavassa (2.2) neuroniin tulee m synapsia ja aktivointifunktio on sigmoid-funktio. Symboli w viittaa synapsien painoihin, symboli x neuronin syötteeseen ja symboli b on neuronin vakiotermi.

Tätä operaatiota toistetaan jokaisella neuronilla jokaisella neuroverkon kerroksella, kunnes päädytään ulostulokerrokseen. Ulostulokerroksella jokaiselle neuronille lasketaan kaavan 2.2 mukaisesti arvo. Neuroverkon ulostulokerroksen neuronien arvo kuvaa ennustuksen todennäköisyyttä (LeCun, Bengio ja Hinton 2015). Jos neuroverkolle annetaan kuvia kissoista ja koirista ja sillä on kaksi neuronia ulostulokerroksella, joista neuroni N_1 tarkoittaa koiraa ja neuroni N_2 tarkoittaa kissaa, niin neuronien N_1 ja N_2 arvot tarkoittavat, kuinka vahvasti neuroverkko uskoo kuvassa olevan koiran tai kissan. Voidaan esimerkiksi verrata näitä arvoja suoraan ja sanoa, jos N_1 arvo on suurempi kuin N_2, niin kuvassa on koira, muuten kissa.

Yleensä ennen kouluttamista neuroverkkojen painot ja vakiotermit alustetaan satunnaisilla arvoilla. Neuroverkkojen kouluttaminen tapahtuu yleisesti vastavirta-algoritmilla (engl. backpropagation). Ensin saadaan neuroverkolta syötteelle ulostulo. Arvion virhe lasketaan virhefunktiollta saadun lopputuloksen ja oikean lopputuloksen perusteella. Virheen perusteella muutetaan neuroverkon painoja ja vakiotermejä gradienttimenetelmällä (engl. gradient decent) laskemalla osittaisderivaattoja, tavoitteena pienentää neuroverkon virhettä. Neuro-

verkon painoja ja vakiotermejä muutetaan määritellyn oppimisnopeuden (engl. learning rate) mukaan. (Mohammed, Khan ja Bashier 2017). Tätä operaatiota toistetaan iteratiivisesti, kunnes neuroverkon arviot ovat riittävän tarkkoja. Periaatteessa vastavirta-algoritmin tavoite on löytää funktion globaali minimi, jossa virhe on mahdollisimman pieni.

2.3 Konvoluutioneuroverkot

Konvoluutioneuroverkot (engl. convolutional neural networks) ovat neuroverkkoja, joita käytetään usein kuvien tunnistamisessa. Huomattava määrä syväoppimisen menestymisestä perustuu erikoistuviin arkkitehtuureihin, kuten konvoluutioneuroverkkoihin (Aggarwal 2018). Konvoluutioneuroverkot ovat suunniteltu ottamaan syötteenä moniulotteisia matriiseja, kuten kuvan kaikki pikselit (LeCun, Bengio ja Hinton 2015).

Aggarwalin (2018) mukaan, konvoluutioneuroverkkojen toiminta perustuu konvoluutiokerroksiin. Konvoluutiokerroksissa on suodatin (engl. filter), joka kartoittaa aktivaatiot kerrokselta toiselle. Seuraavan kerroksen syöte muodostetaan laskemalla suodattimen pistetulo jokaisessa kuvan pikselissä. Käytännössä suodattimen voidaan ajatella olevan funktio, joka pystyy tunnistamaan kuvioita. Suodatinta kuljetetaan kuvan päällä ja jokaisessa pikselissä lasketaan matriisin pistetulo. Pistetuloista muodostetaan uusi matriisi sijoittamalla pistetulon arvo vastaavien pikselien paikalle ja tämä ulostulo annetaan seuraavalle kerrokselle. Uusi pistetulo kuvaa, kuinka hyvin suodatin täsmäsi kuvassa olevaa muotoa kyseisessä kohdassa. Tätä operaatiota toistetaan, kunnes päädytään ulostulokerrokselle ja saadaan neuroverkolta ennustus.

Esimerkiksi jos halutaan tunnistaa kuvasta ihmisen kasvot, ensimmäisellä piilokerroksella voidaan tunnistaa käyriä. Nämä yhdistetään seuraavalla kerroksella, jolloin voidaan tunnistaa esimerkiksi jo kasvojen eri piirteitä. Lopuksi viimeisellä kerroksella tunnistetaan kasvot kuvasta.

3 Vahvistettu oppiminen

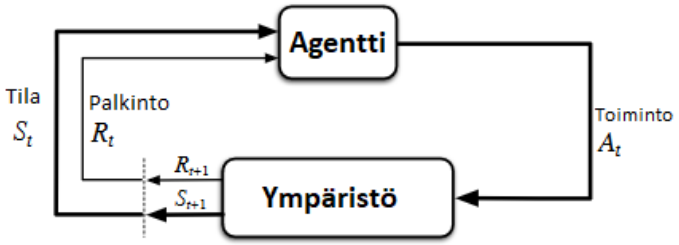
Edellisissä luvuissa on tutustuttu koneoppimiseen ja sen keskeisiin termeihin ja niiden toimintaan. Tässä luvussa tutustutaan vahvistettuun oppimiseen soveltaen edellisten lukujen teoriaa ja tutkimalla vahvistetun oppimisen ominaisuuksia verraten sitä muihin koneoppimisen menetelmiin.

3.1 Määritelmä

Kuten luvussa 2 on todettu, koneoppiminen jaetaan usein kahteen paradigmaan, jotka ovat ohjattu ja ohjaamaton oppiminen. Vahvistettu oppiminen poikkeaa näistä paradigmoista, sillä se ei ohjatun oppimisen tapaan tarvitse syöte-tulos-pareja, eikä se ohjaamattoman oppimisen tapaan tulkitse rakenteita. Sen sijaan vahvistettu oppiminen pyrkii vain maksimoimaan palkinnon mahdollisimman tehokkaasti, ympäristön tilan ja palautteen avulla (Sutton ja Barto 2017). Näin ollen voidaan todeta myös vahvistetun oppimisen olevan oma paradigmansa monen muun vähemmän tunnetun vaihtoehdon ohella.

Kun ajatellaan oppimista, ensimmäinen asia joka voi tulla mieleen on vuorovaikutus ympäristön kanssa. Kun vauva leikkii, vauvalla ei ole opettajaa, vaan tieto ympäristöstä tulee aistien kautta, mukaan lukien mikä toiminta aiheuttaa halutun lopputuloksen. Vahvistettu oppiminen tarkoittaa agentin vuorovaikutusta ympäristön kanssa, jossa agentti toiminnolla saa ympäristöltä palkinnon kuvion 2 mukaisesti. Palkinto määritellään palkintofunktiolla (engl. reward function), joka on menetelmä mitata tekojen hyödyllisyyttä tavoitteen saavuttamista varten, ja näin opettaa agenttia (Rummery ja Niranjan 1994). Agentti ei tiedä oikeita toimintoja, vaan sen täytyy kokeilemalla löytää toiminnot, joiden avulla saadaan suurin mahdollinen palkinto. Pitää kuitenkin huomata, että aina palkintoa ei saada heti, vaan palkinnon saanti voi vaatia joukon oikeita toimintoja. Esimerkiksi shakissa palkinto voitaisiin saada syömällä vastustajan nappula, mutta sen toteuttaminen voi vaatia useamman vuoron verran liikkeitä.

Vahvistettu oppiminen soveltuu hyvin ongelmien yleistämiseen, sillä ympäristössä opittu tieto auttaa agenttia toimimaan myös muissa vastaavissa ympäristöissä. Vahvistettu oppiminen



Kuvio 2: Kaavio vahvistetun oppimisen toiminnasta. Aluksi agentti saa ympäristöltä tilan S_t ja palkinnon R_t. Kun toiminto A_t on suoritettu, saadaan ympäristöltä seuraava tila S_{t+1} ja palkinto R_{t+1} (Sutton ja Barto 2017).

onkin osa suurempaa tavoitetta tekoälyssä, jossa pyritään kehittämään yleistä tekoälyä (Sutton ja Barto 2017).

Vahvistetussa oppimisessa on monia hyötyjä muihin menetelmiin verrattuna. Opittu tieto on helposti sovellettavissa muissa vastaavissa ympäristöissä, eikä se rajoitu ihmisen ymmärrykseen ongelmasta. Vahvistetussa oppimisessa ihminen ei ole antamassa vastauksia agentille, vaan sen pitää satunnaisilla liikkeillä tai muilla algoritmeilla pystyä löytämään tapa ratkaista ongelma. Näin agentti pystyykin löytämään yllättäviä ratkaisuja ongelmiin, joita ihminen ei osaisi ennakoida. Lisäksi vahvistettu oppiminen ei tarvitse dataa ympäristön ulkopuolelta, vaan pystyy oppimaan kaiken ympäristöltä. Tämän ansiosta agentti pystyy harjoittelemaan ongelman ratkaisemista huomattavasti nopeammin kuin esimerkiksi ihminen pystyisi, esimerkiksi pelaamalla peliä. Toisaalta vahvistettu oppiminen rajoittuu ongelmiin, jotka voidaan helposti ratkaista ympäristön ja sen palautteen avulla. Lisäksi yksi suurista vahvistetun oppimisen haasteista on agentin opettaminen korkean ulottovuuden sensorien syötteestä (engl. high-dimensional sensory inputs) (Mnih ym. 2013). Esimerkiksi itseohjautuvan auton sensoreista on tärkeää pystyä erottelemaan asiat, jotka voivat vaikuttaa agentin päätöksiin, kuten kaistat ja muut autot.

3.2 Q-oppiminen

Vahvistetussa oppimisessa voidaan myös mallien, kuten neuroverkkojen, sijaan käyttää muita algoritmeja. Tätä kutsutaan mallittomaksi oppimiseksi (engl. model-free learning) (Marco

Wiering 2012, S.27-28). Näitä algoritmeja voidaan tosin käyttää myös mallien kanssa, jolloin mallien tehokkuutta voidaan nostaa entisestään. Tässä aliluvussa tutustutaan menetelmään nimeltä Q-oppiminen (engl. Q-learning).

Vahvistettua oppimista käytetään usein tilanteissa, joita on hankala esittää yksinkertaisen datan avulla. Q-oppiminen perustuu Q-funktioon, joka ennakoi palkinnon määrää jokaiselle toiminnolle. Tätä arviointia päivitetään aina kun toiminnasta saadaan palaute. (Rummery ja Niranjan 1994). Q-funktio on suunniteltu löytämään toiminnot, joilla saavutetaan suurin mahdollinen palkinto mahdollisimman lyhyellä aikavälillä (Sutton ja Barto 2017). Toisin sanoen, Q-oppimisen tehtävä on löytää mahdollisimman optimaalinen ratkaisu ongelmaan, kun ympäristöä ei tunneta. Q-oppimista voidaan käyttää ilman erillistä mallia, kuten neuroverkkoa, jolloin voidaan suoraan valita oletettu parhain toiminto ennakoidun palkinnon pohjalta. Kun käytetään neuroverkkoa, se oppii algoritmeilta tilojen ja toimintojen yhteyksistä, ja käyttää tätä tietoa painojen ja vakiotermien päivittämisessä.

Q-funktio päivitetään jokaisen toiminnon jälkeen: kun tilassa S_t suoritetaan toiminto A_t ja päädytään tilaan S_{t+1}, saadaan toiminnosta palautteena palkinto. Tätä palkintoa käytetään Q-funktion päivittämiseen käyttäen Aggarwalin (2018, s.388) esittämää kaavaa

$$Q(s_t, a_t) \Leftarrow Q(s_t, a_t)(1-\alpha) + \alpha[r_t + \gamma max_a Q(s_{t+1}, a)] \tag{3.1}$$

Kaava (3.1) antaa uuden Q-funktion arvon tilalle s_t ja toiminnolle a_t, kun sille annetaan nykyinen Q-funktion arvo $Q(s_t, a_t)$ ja $max_a Q(s_{t+1}, a)$, joka tarkoittaa suurinta mahdollista palkintoa toiminnolla a tilassa s_t. Symboli α tarkoittaa oppimisnopeutta, eli kuinka nopeasti funktion arvo muuttuu, ja symboli γ määrittelee, kuinka tärkeä painoarvo suurimmalla mahdollisella palkinnolla on. Symboli R_t tarkoittaa saatua palkintoa, kuten kaaviossa 2 on esitetty.

Alkuun Q-funktio ei voi antaa parhaita toimintoja tilassa S_t, sillä oppimista ei ole tapahtunut vielä. Tämän seurauksena pitää toimintoja valita satunnaisesti, jotta voidaan oppia niiden pohjalta. Myöhemmin kun Q-funktio on oppinut mikä toiminto A_t tilassa S_t palauttaa suurimman mahdollisen palkinnon, ei aina valita toimintoa A_t. Esimerkkinä, jos on kaksi mah-

dollista toimintaa, joista toiminta A_1 antaa aina palkinnoksi +1 ja toiminta A_2 antaa viiden prosentin mahdollisuudella palkinnoksi +100, muuten 0. Jos Q-oppimisen algoritmi alkuun kokeilee satunnaisesti toimintoja, se saattaa todeta toiminnon A_1 olevan aina parempi, vaikka A_2 saattaisi antaa huomattavasti enemmän palkintoa pidemmällä aikavälillä.

Tämän seurauksena Q-oppimisessa on idea hyödyntämisestä (engl. exploitation) ja satunnaisesti etsinnästä (engl. exploration). Oppimisen aikana halutaan kokeilla satunnaisesti toimintoja, sillä silloin saatetaan löytää uusia toimintoja, jotka palauttavat suuremman palkinnon (Marco Wiering 2012, s.35-36). Yksi tapa toteuttaa tämä on esitellä muuttuja L, joka tarkoittaa mahdollisuutta kokeilla toimintoa satunnaisesti. Alkuun L arvo on esimerkiksi 100%, mutta sen arvo alkaa laskemaan oppimisen edetessä, kunnes lopulta se on jo erittäin lähellä nollaa. Algoritmi siis kokeilee koko ajan vähemmän satunnaisia liikkeitä.

Esitellään yksinkertainen esimerkki Q-oppimisen käytöstä. Oletetaan, että pelilautana on matriisi, joista ruudut voivat olla tyyppiä alku, tyhjä, este ja maali. Palkinnot ovat seuraavat: tyhjä -1, este -10 ja maali +10. Tämä tarkoittaa, että tavoite on päästä maaliin mahdollisimman nopeasti vältellen esteitä. Agentti voi liikkua ruudukossa ylös, alas, vasemmalle tai oikealle. Pelissä on näin ollen rajallinen määrä mahdollisia tiloja ja toimintoja. Tiloista ja toiminnoista luodaan matriisi 1 mukaisesti. Tämä matriisi on nimeltään Q-taulukko (engl. Q-table). Algoritmi aloittaa valitemalla satunnaisen toiminnon, sillä dataa toiminnoista, jotka tuottavat palkintoja, ei ole vielä. Kun toiminto on suoritettu, päivitetään Q-taulukon tila palkinnon avulla. Taulukko 1 on esimerkki Q-pöydästä, joka on oppinut tilojen ja toimintojen välisiä suhteita ja pystyy näin ennakoimaan palkinnon määrää.

	Vasen	Oikea	Ylös	Alas
Alku	-10	+10	-1	-10
Tyhjä	-1	-1	+10	-1
Este	-1	-10	+10	-1
Maali	-1	-10	-1	-1

Taulukko 1: Esimerkki koulutetusta Q-taulukosta

4 Esimerkkejä vahvistetun oppimisen sovelluksista

Tässä luvussa käsitellään vahvistetun oppimisen sovelluksia ja saavutuksia. Kuten luvussa 3 käsiteltiin, vahvistettu oppiminen soveltuu parhaiten tehtäviin, joissa voidaan oppia ympäristön palautteesta. Tästä hyviä esimerkkejä ovat pelit ja robotiikka.

Peleistä erityisesti shakki on ollut klassinen ongelma tekoälylle. DeepMind-yritys on kehittänyt tekoälyohjelma AlphaZeron, joka pystyy pelaamaan shakkia, go:ta ja shogia. Näistä erityisesti go:n pelaaminen on huomattava saavutus, sillä go:ssa on huomattava määrä mahdollisia eri tiloja ja toimintoja, joka tarkoittaa että ongelmaa ei voida ratkoa laskemalla kaikki mahdolliset liikkeet annetun aikarajoitteen aikana. AlphaZeron tekoäly oppi pelaamalla itseään vastaan käyttäen hyväksi vahvistettua oppimista, syvää neuroverkkoa ja Monte-Carlo -puuhaku menetelmää (engl. Monte Carlo tree search) (Silver ym. 2018). Monte-Carlo -puuhaku vastaa Q-oppimista, mutta toisin kuin Q-oppimisessa, Monte-Carlo -puuhaku simuloi pelin loppuun ja tallentaa tuloksen (Chaslot ym. 2008).

Robotiikassa haasteena on kehittää robotteja, jotka pystyvät oppimaan ja yleistämään osaamistaan. Vahvistettu oppiminen tarjoaa hyvän menetelmän oppimiselle, koska robotti voi oppia ympäristöltä suoraan tai simulaation kautta. Tavoitteena on luoda autonomisia robotteja, jotka pystyvät yleistämään ongelmia ja oppimaan uutta. Vaikka materiaalit, moottorit ja sensorit ovat kehittyneet huomattavasti, robottien tekoäly ei ole vielä kehittynyt tarpeeksi pitkälle, jotta ne vastaisivat toiminnaltaan ihmisiä (Mnih ym. 2015). Robotiikassa vahvistetun oppimisen tavoite on luoda roboteille kyky oppia, parantua tehtävässään ja mukautua muuttuviin ympäristöihin. Vahvistettu oppiminen soveltuu tähän erinomaisen hyvin, sillä oppiminen tapahtuu valmiissa ympäristössä, jota voidaan havainnoida sensorien kautta. Toisaalta robottien pitää pystyä oppimaan korkean ulottuvuuden datasta tehokkaasti, ja tämä on vahvistetussa oppimisessa suuri haaste, erityisesti kun otetaan huomioon myös robotin oma motoriikka (Kormushev, Calinon ja Caldwell 2013). Vahvistettu oppiminen soveltuu robottien opettamiseen erityisen hyvin, jos ongelmaa voidaan simuloida, sillä simulaatioissa ei tarvitse välittää mahdollisesta fyysisestä vahingosta ja simulaatio on helppo aloittaa uudestaan.

Sovelluksista tutkitaan tarkemmin DeepMind-yrityksen Atari-tekoälyä teknisestä näkökul-

masta ja pohditaan vahvistetun oppimisen roolia itseohjautuvien autojen kehityksessä. Pelit ja itseohjautuvat autot ovat hyviä esimerkkejä vahvistetulle oppimiselle, koska niissä on valmis ympäristö, josta voidaan kerätä dataa esimerkiksi kuvan avulla.

4.1 DeepMind-yrityksen Atari-tekoäly

Tässä luvussa tutustutaan DeepMind-yrityksen kehittämään Atari-tekoälyyn pääasiassa Mnih ym. (2013) julkaisuun pohjautuen. Kuten luvussa 3 mainitaan, yksi suurista vahvistetun oppimisen haasteista on agentin opettaminen korkean ulottuvuuden sensorien syötteestä. Näiden haasteiden ratkaiseminen on kuitenkin oleellista, jotta ympäristöistä voidaan kerätä dataa esimerkiksi kuvan avulla. DeepMind-yrityksen Atari-tekoäly pystyy pelaamaan seitsemän eri Atari-2600-peliä, ilman merkittäviä muutoksia peleihin tai tekoälyyn. Tämä on merkittävää, sillä yksi tekoälyn suurimmista tavoitteista on aina ollut yleinen tekoäly (engl. general artificial intelligence).

Ohjatussa oppimisessa kuvien pohjalta oppiminen on helppoa, sillä niissä ihminen voi erikseen jakaa kuvat kategorioihin. Vahvistetussa oppimisessa taas kuvan pohjalta oppiminen on huomattavasti haastavampaa, sillä oppiminen tapahtuu palkintojen pohjalta, joita saatetaan saada harvoin ja toiminto ei välttämättä johda välittömästi palkintoon. Tämän vuoksi vahvistettu oppiminen on erinomainen menetelmä opettamaan tekoälyä, jonka pitää pystyä oppimaan eri ympäristöiltä ja yleistämään niiden välillä ilman erillistä tietoa niistä.

DeepMind-yrityksen Atari-tekoäly suoriutuu ihmisen tasoisesti tai huomattavasti paremmin peleissä Breakout, Enduro, Pong ja Beam Rider. Pelit joissa se ei menesty niin hyvin ovat Q*bert, Seaquest ja Space Invaders. Jälkimmäiset pelit ovat haastavampia, sillä niissä neuroverkon pitäisi pystyä löytämään toimiva strategia, joka koostuisi joukosta toimintoja pidemmällä aikavälillä, eikä palkintoa saada välittämästi. DeepMind-yrityksen Atari-tekoälyn toiminta perustuu konvoluutioneuroverkkoon, Q-oppimiseen ja kokemuksen toistomekanismiin (engl. experience replay mechanism). Konvoluutioneuroverkko koulutettiin käyttäen Q-oppimista ja se saa syötteeksi kuvankaappauksen pelistä.

Pelin kuvankaappauksista voidaan konvoluutioneuroverkkojen avulla kerätä tärkeää tietoa, kuten pelaajan ja vihollisten sijainteja. Toisaalta kuvankaappauksessa on myös paljon tur-

haa tietoa, kuten tausta. Konvoluutioneuroverkolle annetaan neljä viimeisintä kuvaa pelistä. Näiden kuvien avulla peli tietää, mitä edellisten tilojen aikana on tapahtunut ympäristössä. Kuvat ovat kooltaan 210x160 pikseliä. Yleensä kuvia käsitellään ennen neuroverkoille syöttämistä, sillä se voi olla prosessoinnin kannalta erityisen raskasta. Tämän vuoksi pelin kuvankaappaus muunnetaan 84x84 pikselin kokoiseksi ja mustavalkoiseksi. Lopulta neuroverkon syötekerroksella on 84x84x4 neuronia.

Q-oppimisen toteutus poikkeaa perinteisestä toteutuksesta. Kuten neuroverkolle, myös Q-funktiolle annetaan syötteenä neljä edellistä kuvankaappausta. Lisäksi käytössä on ollut arkkitehtuuri, jossa jokaiselle toiminnolle on oma yksikkö ulostulolle (engl. output unit) ja ainoastaan tila S_i annetaan neuroverkolle. Tämän menetelmän etu on se, että neuroverkon tarvitsee vain kerran käsitellä syöte ja sen pohjalta voidaan laskea Q-funktion arvot kaikilla mahdollisille toiminnoille A_i tilassa S_i.

Kokemuksen toistomekanismi avulla agentti pystyy muistamaan ja käyttämään kokemuksia uudelleen. Yksi tämän hyödyistä on harvinaisten tapahtumien muistaminen. (Schaul ym. 2015). Toisin sanoen kokemuksen toistolla tarkoitetaan menetelmää, jossa tallennetaan eri tilanteita muistiin ja niitä käytetään uudelleen neuroverkon kouluttamiseen, kun niitä ei esiinny pitkään aikaan. Näin ollen vaikuttaa siltä, että tämä mekanismi on erityisen hyödyllinen, kun tekoälyä pyritään kouluttamaan pelaamaan monia eri pelejä. Jos neuroverkko opetetaan ensin pelaamaan peliä A ja sen jälkeen peliä B, neuroverkko alkaa unohtamaan, miten peliä A pelattiin. Kun neuroverkolle koulutetaan pelin A tilanteita myöhemmin satunnaisesti, se pystyy muistamaan opitut asiat.

DeepMind-yrityksen Atari-tekoäly on erinomainen esimerkki syvien konvoluutioneuroverkkojen, vahvistetun oppimisen ja Q-oppimisen soveltamisesta yleistettävän ongelman ratkaisemiseen. Agentti pystyy toimimaan monissa eri poikkeavissa ympäristöissä ilman muutoksia ympäristöön tai agenttiin.

4.2 Itseohjautuvat autot

Tässä luvussa pohditaan vahvistetun oppimisen soveltuvuutta itseohjautuvissa autoissa ja pyritään ymmärtämään sen haasteita. Itseohjautuvat autot ovat yksi tämän hetken suurimmista

haasteista tekoälyn puolella. Tehtävä on teknisesti erittäin haastava ja yhteiskunnallisesti itseohjautuvilta autoilta odotetaan lähes täydellisyyttä. Yksikin vaaratilanne voi aiheuttaa mittavaa vahinkoa.

Tällä hetkellä standardi lähestymistapa itseohjautuviin autoihin vaikuttaa olevan ohjattu oppiminen, sillä tehtävää pyritään jakamaan pienempiin osiin, kuten esineiden ja ympäristöjen tunnistamiseen. Vahvistettua oppimista pidetään vahvana menetelmänä, mutta toistaiseksi sitä ei ole onnistuttu hyödyntämään itseohjautuvissa autoissa vaaditulla tasolla (Sallab ym. 2017). Ohjatun oppimisen menestyminen perustuu tehtävän monimutkaisuuteen, jonka vuoksi itseohjautuva auto on helpompi kouluttaa esimerkin avulla.

Kuten luvussa 3 on todettu, yksi suurimmista vahvistetun oppimisen ongelmista on agentin opettaminen korkean ulottuvuuden datasta. Itseohjautuvat autot saavat tiedon ympäristöstään sensoreiden kautta, kuten kuvista. Toisaalta Sallab ym. (2017) toteavat, että vaikka kuvat ovat korkean ulottuvuuden dataa, ei kuvasta tarvita niin paljon tarkkaa dataa. Tällä tarkoitetaan sitä, että kuvasta on tärkeää löytää esteet, muut autot ja kaistojen viivat, ei edessä ajavan auton merkkiä. Kuten luvussa 3 todettiin, vahvistetussa oppimisessa haasteena on palkintojen ja toimintojen yhteyden oppiminen, sillä se on usein epäselvää agentille ja tämän vuoksi agentin on hankala oppia oikeat toiminnot, joilla palkintoja saadaan. Lisäksi haasteena on se, että agentti joutuu oppimaan tyhjästä erittäin haastavan ja monimutkaisen tehtävän. Kun huomioidaan kaikki edellä mainitut ongelmat, on mahdollista nähdä miksi vahvistettu oppiminen ei ole ainakaan tällä hetkellä parhain mahdollinen menetelmä tähän tehtävään.

Kuten Sallab ym. 2017 mainitsevat, itseohjautuvan auton toiminta voidaan jakaa kolmeen kategoriaan. Ensimmäinen näistä on tilanteen havainnointi. Agentin on pystyttävä tunnistamaan esteitä, liikennemerkkejä ja muita tärkeitä elementtejä. Toinen kategoria on ennakointi, sillä agentin on pystyttävä ennakoimaan tilanteita ja muiden kulkuneuvojen käyttäytymistä. Kolmas kategoria on toimintojen suunnittelu, agentin pitää pystyä luomaan lista toimintoja havainnointien ja ennakointien perusteella.

Koska vahvistetussa oppimisessa opitaan virheiden kautta, ei agenttia luonnollisesti voida kouluttaa oikealla autolla. Tämä on yhtä aikaa heikkous ja vahvuus vahvistetulle oppimiselle, sillä agentin pitää oppia simulaatiossa. Simulaation pitää olla tarkka ja realistinen, mutta

jos tässä onnistutaan niin agentti voi harjoitella simulaatiossa huomattavasti enemmän kuin ihminen pystyisi. Voidaan siis todeta ohjatun oppimisen olevan yleisesti parempi menetelmä, mutta vahvistetulla oppimisella on paljon potentiaalia.

5 Yhteenveto

Tutkielman tarkoituksena oli pyrkiä ymmärtämään vahvistettua oppimista ja sen sovelluksia. Aiheen ymmärtämiseksi jouduttiin myös avaamaan termejä, kuten ohjattu ja ohjaamaton oppiminen. Vahvistettu oppiminen poikkeaa selvästi ohjatusta ja ohjaamattomasta oppimisesta, sillä niissä annetaan dataa tulkittavaksi. Vahvistetussa oppimisessa tekoäly joutuu oppimaan valmiissa ympäristössä suorittamalla toimintoja. Agentti voi havainnoida ympäristöä esimerkiksi pelin kuvankaappauksien tai sensorien keräämään datan avulla. Kun agentti suorittaa ympäristössä toiminnon, niin saadaan ympäristöltä palkinto. Palkinnon avulla agentti pystyy oppimaan tilojen ja toimintojen yhteyksistä.

Myös vahvistetun oppimisen menetelmään nimeltä Q-oppiminen tutustuttiin. Q-oppiminen on menetelmä, jolla voidaan ennakoida toimintojen tulevia palkintoja oppimalla ympäristön tilojen ja toimintojen yhteyksiä. Q-funktio palauttaa toiminnon, jolla se uskoo agentin saavan suurimman mahdollisen palkinnon. Toisin sanoen Q-funktion tehtävä on löytää agentille ne toiminnot, jotka maksimoivat palkinnon määrän mahdollisimman vähäisellä määrällä toimintoja.

Sovelluksista käsiteltiin DeepMind-yrityksen AlphaZeroa ja robotiikkaa yleisestä näkökulmasta. AlphaZeron kohdalla esiteltiin Monte-Carlo -puuhaku, joka vastaa hieman Q-oppimista. Vahvistetun oppimisen todettiin olevan hyvä menetelmä robotiikkaan, sillä tavoite on yleistää ongelmia eri ympäristöissä ja antaa roboteille kyky oppia itse ympäristöltä.

Tarkemmin käsiteltiin DeepMind-yrityksen Atari-tekoälyä. Se pystyy yleistämään oppimaansa monessa eri pelissä. Samalla todettiin, että yksi vahvistetun oppimisen suurimmista heikkouksista ovat viivästyneet palkinnot. Tämä tarkoittaa, että palkintoa ei aina saada välittömästi toiminnon jälkeen, vaan se voi esimerkiksi vaatiia joukon toimintoja. Oikean strategian löytäminen voi olla haastavaa monimutkaisessa ympäristössä, jos palkintoja on harvassa. Tämän seurauksena DeepMind-yrityksen Atari-tekoälyllä olikin ongelmia suoriutua peleissä Q*bert, Seaquest ja Space Invaders.

Vastaavasti myös itseohjautuvien autojen toimintaa pohdittiin ja verrattiin ohjatun oppimisen ja vahvistetun oppimisen menetelmiä kouluttamisessa. Luvussa todettiin ohjatun oppimisen

olevan ainakin toistaiseksi ylivoimainen menetelmä, sillä itseohjautuvien autojen tehtävä on erittäin monimutkainen ja ihmisen esimerkit auttavat sitä huomattavasti oppimisessa. Toisaalta jos itseohjautuvaa autoa pystytään simuloimaan ympäristössä realistisesti, se pystyy ajamaan huomattavasti enemmän kuin ihminen ikinä voisi.

Lähdekirjallisuuteen viitaten avattiin ymmärrystä vahvistetun oppimisen menetelmistä ja erityisesti sen rajoitteista ja haasteista. Vahvistetun oppimisen todettiin olevan vahva menetelmä tapauksiin, joissa oppiminen tapahtuu ympäristössä tai ympäristöä on mahdollista simuloida järkevästi. Vahvistetun oppimisen yksi suurimmista eduista on se, ettei se tarvitse dataa ympäristön ulkopuolelta oppiakseen, ja agentti pystyy harjoittelemaan huomattavasti enemmän kuin ihminen.

Lähteet

Aggarwal, Charu C. 2018. *Neural Networks and Deep Learning*. https://doi-org.ezproxy.jyu.fi/10.1007/978-3-319-94463-0.

Alpaydin, Ethem. 2016. *Machine Learning : The New AI*. MIT Press, Cambridge. https://ebookcentral.proquest.com/lib/jyvaskyla-ebooks/detail.action?docID=4714219.

Chaslot, Guillaume, Sander Bakkes, Istvan Szita ja Pieter Spronck. 2008. "Monte-Carlo Tree Search: A New Framework for Game AI." Teoksessa *AIIDE*.

Kormushev, Petar, Sylvain Calinon ja Darwin G. Caldwell. 2013. "Reinforcement Learning in Robotics: Applications and Real-World Challenges". *Robotics* 2 (3): 122–148. ISSN: 2218-6581. doi:10.3390/robotics2030122. http://www.mdpi.com/2218-6581/2/3/122.

LeCun, Yann, Yoshua Bengio ja Geoffrey Hinton. 2015. "Deep learning". *nature* 521 (7553): 436. https://creativecoding.soe.ucsc.edu/courses/cs523/slides/week3/DeepLearning_LeCun.pdf.

Marco Wiering, Martijn van Otterlo. 2012. *Reinforcement Learning: State-of-the-Art*. Springer-Verlag Berlin Heidelberg. https://link-springer-com.ezproxy.jyu.fi/book/10.1007%2F978-3-642-27645-3.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra ja Martin A. Riedmiller. 2013. "Playing Atari with Deep Reinforcement Learning". *CoRR* abs/1312.5602. arXiv: 1312.5602. http://arxiv.org/abs/1312.5602.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski ym. 2015. "Human-level control through deep reinforcement learning". *Nature* 518 (7540): 529. https://daiwk.github.io/assets/dqn.pdf.

Mohammed, Mohssen, Muhammad Badruddin Khan ja Eihab Bashier Mohammed Bashier. 2017. *Machine Learning : Algorithms and Applications.* CRC Press. ISBN: 9781498705387. http://search.ebscohost.com.ezproxy.jyu.fi/login.aspx?direct=true&db=nlebk&AN=1293656&site=ehost-live.

Rummery, G. A., ja M. Niranjan. 1994. "On-line Q-learning using connectionist systems". http://mi.eng.cam.ac.uk/reports/svr-ftp/auto-pdf/rummery_tr166.pdf.

Sallab, Ahmad EL, Mohammed Abdou, Etienne Perot ja Senthil Yogamani. 2017. "Deep reinforcement learning framework for autonomous driving". *Electronic Imaging* 2017 (19). https://www.ingentaconnect.com/content/ist/ei/2017/00002017/00000019/art00012#.

Schaul, Tom, John Quan, Ioannis Antonoglou ja David Silver. 2015. "Prioritized experience replay". *arXiv preprint arXiv:1511.05952.* https://arxiv.org/abs/1511.05952.

Silver, David, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot ym. 2018. "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play". *Science* 362 (6419): 1140–1144. ISSN: 0036-8075. doi:10.1126/science.aar6404. eprint: https://science.sciencemag.org/content/362/6419/1140.full.pdf. https://science.sciencemag.org/content/362/6419/1140.

Sutton, Richard S, ja Andrew G Barto. 2017. *Reinforcement learning: An introduction.* MIT press. http://incompleteideas.net/book/bookdraft2017nov5.pdf.