Saku Kärkkäinen

Neuroevoluutio koneoppimismenetelmänä

Tietotekniikan kandidaatintutkielma

26. huhtikuuta 2018

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Saku Kärkkäinen

Yhteystiedot: sapekark@student.jyu.fi

Ohjaaja: Sanna Mönkölä

Työn nimi: Neuroevoluutio koneoppimismenetelmänä

Title in English: Neuroevolution as a method of machine learning

Työ: Kandidaatintutkielma

Sivumäärä: 23+0

Tiivistelmä: Neuroevoluutiolla tarkoitetaan neuroverkkojen suunnittelua ja opettamista evoluutiolaskennan avulla. Tutkielmassa tutustutaan neuroevoluution kokonaisvaltaisesti, samalla kartoitetaan sen käyttökelpoisuutta vaihtoehtoisena koneoppimismenetelmänä eri sovellusalueilla. Tutkielman johtopäätöksenä esitetään neuroevoluution menestyvän tällä hetkellä erityisesti vahvistusoppimisongelmien ratkaisussa. Lisäksi, laskentatehon kasvaessa on todennäköistä, että neuroevoluutiolla on kirkas tulevaisuus syvien neuroverkkojen suunnittelussa.

Avainsanat: Neuroevoluutio, neuroverkot, evoluutioalgoritmit, tekoäly, koneoppiminen

Abstract: The process of using evolutionary algorithms for designing and teaching neural networks is called neuroevolution. The thesis provides a thorough introduction to neuroevolution, while also depicting a view on possible uses for it as an alternative method of machine learning. The thesis concludes by stating that, at the moment, the method excels especially in solving reinforcement learning problems. Furthermore, as computing power increases, the method is likely to have a bright future in designing deep neural networks.

Keywords: Neuroevolution, neural networks, evolutionary algorithms, artificial intelligence, machine learning

Kuviot

Sisältö

1 Johdanto

Tekoäly on ollut yksi viime vuosien puhuttavimmista aiheista. Sen potentiaali muuttaa ihmiselämää pysyvästi on herättänyt valtioiden välistä kilpailua johtoaseman saavuttamiseksi. Kesällä 2017 Kiina julkaisi tavoitteekseen saavuttaa johtava asema tekoälyn saralla vuoteen 2030 mennessä (Mozur 2017). Maaliskuun 2018 lopussa Ranska puolestaan esitteli suunnitelmansa käyttää 1,5 miljardia euroa nostaakseen maan johtavaksi valtioksi tekoälyn saralla Kiinan ja Yhdysvaltojen rinnalle (Rabesandratana 2018). Tällaiset valtiolliset investoinnit korostavat tekoälytutkimuksen merkitystä eräänä tulevien vuosikymmenien merkittävistä tutkimusaloista.

Eräs tällä hetkellä suosittu tekoälymenetelmä on neuroverkko. Tyypillisesti neuroverkon käyttäjä suunnittelee käsin, yrityksen ja erehdyksen kautta, rakenteen käyttämälleen verkolle (Whiteson 2012). Verkon rakenne koostuu neuroneista ja niiden välisistä yhteyksistä. Jotta neuroverkko tarjoaa ratkaisun käyttäjän ongelmaan, tulee sitä opettaa. Verkon opettaminen tapahtuu optimoimalla verkon sisäisiä parametreja yleensä hyödyntäen esimerkiksi gradienttimenetelmää käyttävää opetusalgoritmia. Neuroverkoille on kuitenkin olemassa myös vaihtoehtoisia suunnittelu- ja opetustapoja, kuten neuroevoluutio. Neuroevoluutiossa neuroverkko suunnitellaan ja opetetaan vastaamaan ongelmaan itsenäisesti evoluutiolaskennan avulla.

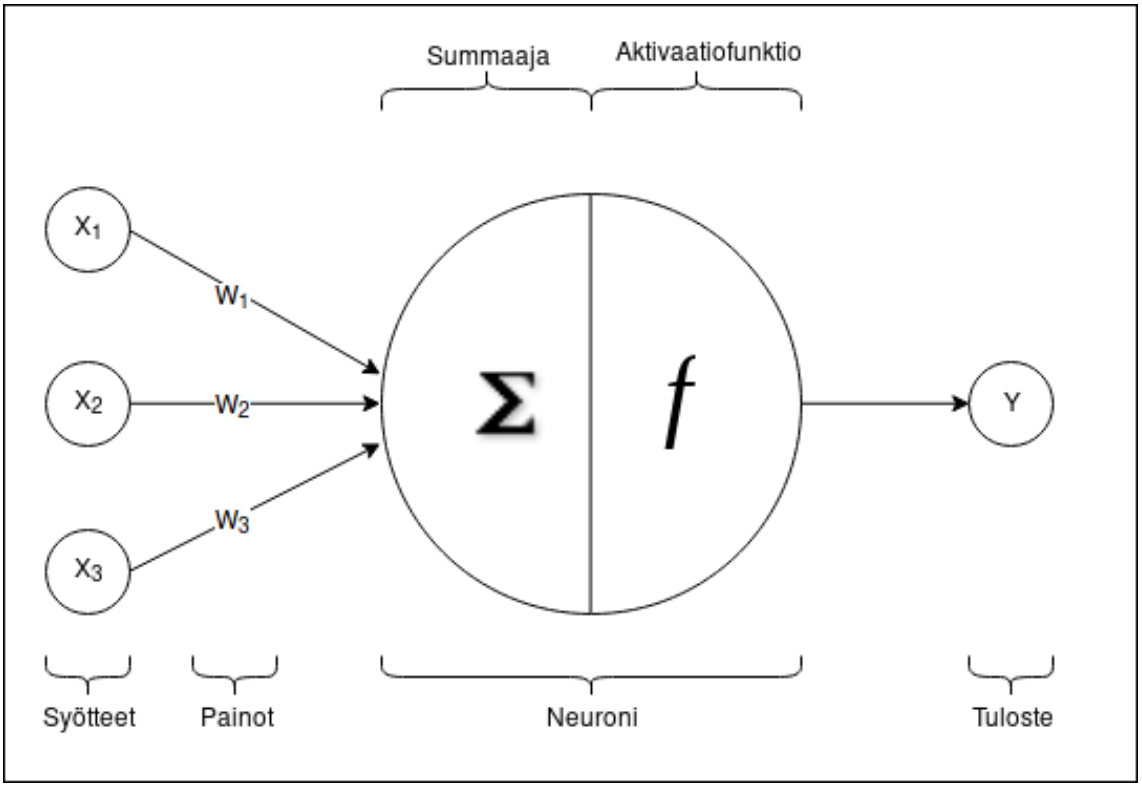
Tämän kandidaatintutkielman tavoitteena on kirjallisuuskatsauksen avulla luoda kuva neuroevoluutiosta koneoppimismenetelmänä. Koneoppiminen on tekoälyn osa-alue, jolla tutkitaan ohjelmistoja, jotka kykenevät itsenäisesti oppimaan malleja niille syötetystä datasta. Tutkielmassa kartoitetaan neuroevoluution vahvuuksia, heikkouksia ja erityispiirteitä muihin menetelmiin verrattuna. Lopputuloksena esitetään kuva menetelmän käyttökelpoisuudesta ja sille parhaiten sopivista sovelluskohteista.

Ennen kuin neuroevoluutiota määritellään täsmällisemmin, tulee ymmärtää osat, joista se koostuu: neuroverkot ja evoluutioalgoritmit. Luvuissa 2 ja 3 määritellään ja esitellään edellä mainitut käsitteet. Sekä neuroverkot että evoluutioalgoritmit ovat biologisesti inspiroituneita menetelmiä. Tässä tutkielmassa niiden esittely kuitenkin tehdään tietojenkäsittelytieteellisestä näkökulmasta ja niiden biologiset vastineet jätetään vain maininnan tasolle. Luvussa

4 käsitellään neuroevoluutiota ja sen soveltamista eri ongelmiin samalla muodostaen kuvaa menetelmän käyttökelpoisuudesta yleisesti. Luvussa 5 muodostetaan yhteenveto tutkielmassa tehdyistä huomioista ja johtopäätöksistä.

2 Neuroverkot

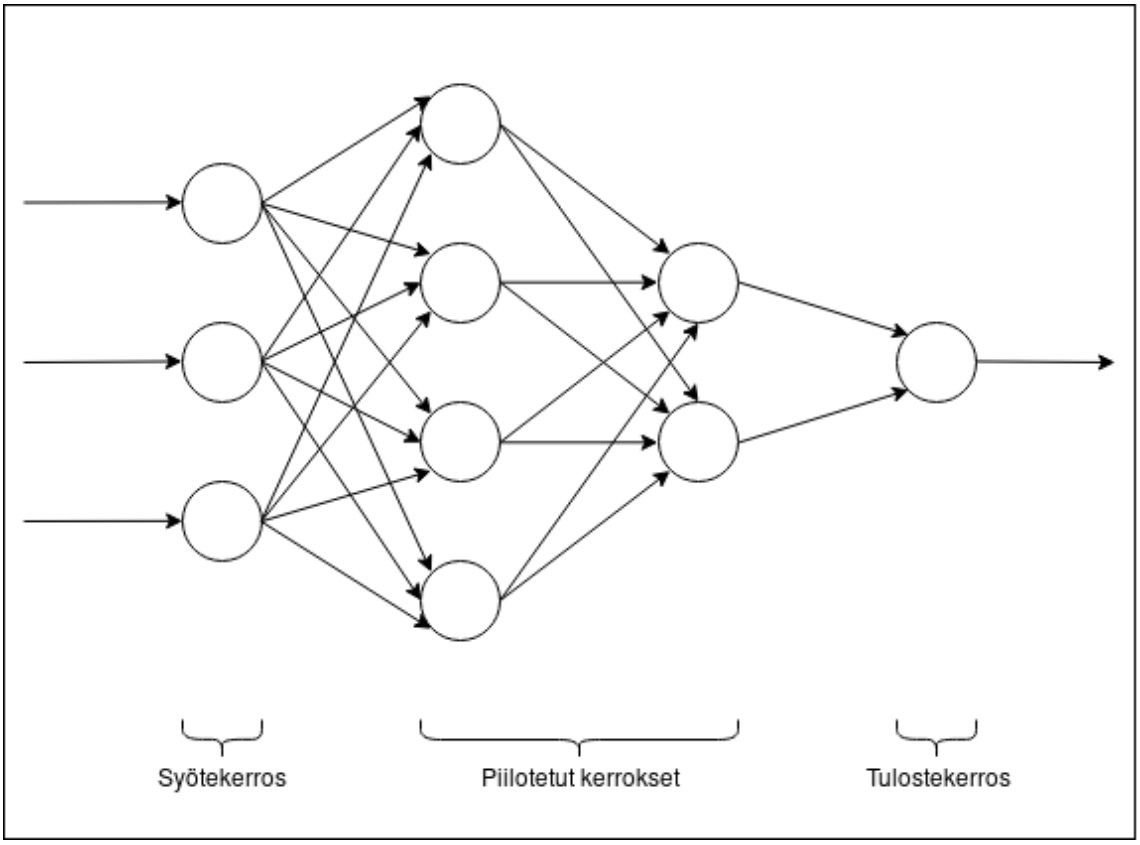
Tietojenkäsittelytieteessä neuroverkoilla tarkoitetaan rinnakkaislaskentaa hyödyntävää laskennallista mallia, jolla on kyky oppia, yleistää tai lajitella dataa (Kröse ja Van der Smagt 1996, s. 13). Neuroverkot ovat biologisesti inspiroituneita, eli niiden toimintatapa on johdettu biologisesta ilmiöstä. Niiden toiminta mallintaa tapaa, jolla hermosto prosessoi tietoa (Rojas 1996, s. 3). Tämän luvun tavoitteena on muodostaa lukijalle perustiedot neuroverkkojen rakenteesta ja toiminnasta. Toimintaan liittyy erityisesti kolme seuraavaa elementtiä: neuronien rakenne, verkon topologia ja oppimisalgoritmi, jolla neuroneita yhdistävät painoarvot määritellään (Rojas 1996, s. 24).



Kuvio 1. Neuronin rakenne

Neuroverkon perusyksikköä, joka vastaa verkossa tapahtuvasta laskennasta, kutsutaan neuroniksi. Neuronit ovat yksinkertaisia prosessointiyksiköitä, jotka kommunikoivat toistensa kanssa signaaleilla, jotka lähetetään niiden välisten painotettujen yhteyksien yli (Kröse ja Van der Smagt 1996, s. 15). Kuvio 1 esittää neuronin rakennetta. Kuten kuviosta käy ilmi,

neuronien ytimessä toimii kaksi funktiota: summaaja ja aktivaatiofunktio. Summaaja laskee summan neuronin vastaanottamien syöte-painoparien tuloista. Aktivaatiofunktio puolestaan muuntaa luodun summan neuronin tulosteeksi. Aktivaatiofunktion voi toteuttaa monin eri tavoin, ja sen rakenne riippuu neuroverkon käyttötarpeista. Riippuen neuronien sijainnista verkossa, ne voidaan jakaa kolmeen eri tyyppiin: syöteneuroneihin, piilotettuihin neuroneihin ja tulosteneuroneihin. Näistä syöteneuronit vastaanottavat syötteen verkon ulkopuolelta, piilotettujen neuronien syöte ja tuloste pysyvät verkon sisällä, ja tulosteneuronien tuloste poistuu verkosta (Kröse ja Van der Smagt 1996, s. 15–16). Samaa tyyppiä olevat neuronit muodostavat verkkoon kerroksia, jotka on nimetty neuronien tyypin mukaan. Esimerkiksi kuvio 2 esittää neuroverkkoa, jossa on kaksi piilotettua kerrosta. Jos verkossa on paljon piilotettuja kerroksia, kutsutaan sitä syväksi neuroverkoksi.



Kuvio 2. Neuroverkon topologia koostuu sen neuroneista ja niiden välisista yhteyksistä.

Neuroverkot voidaan yleisesti ottaen jakaa kahteen kategoriaan riippuen niiden topologiasta, eli mallista, joiden mukaan neuronien väliset liitokset on luotu (Yao 1999). Käytettävä

arkkitehtuuri riippuu verkon soveltamisalueen tarpeista. Eteenpäinsyöttävissä (engl. *feed-forward*) verkoissa data liikkuu suoraviivaisesti syöteneuroneista, mahdollisten piilotettujen neuronien kautta, tulosteneuroneihin (Kröse ja Van der Smagt 1996, s. 17). Eteenpäinsyöttävät verkot ovat rakenteeltaan ja toiminnaltaan yksinkertaisia, minkä vuoksi ne soveltuvat lähinnä yksinkertaisimpien ongelmien ratkaisuun. Takaisinkytketyissä (engl. *recurrent*) neuroverkoissa tasojen välillä ilmenee takaisinkytkentöjä, jotka muodostavat silmukoita, joiden avulla dataa voidaan kierrättää verkon sisällä (Rojas 1996, s. 42). Takaisinkytketyt verkot sopivat erityisen hyvin peräkkäisen datan käsittelyyn, sillä takaisinkytkentöjen ansiosta verkko kykenee muistamaan piirteitä edellisistä syötteistä. Kuvio 2 esittää eteenpäinsyöttävää neuroverkkoa. Kuvassa esiintyvien nuolten suunta esittää tiedon liikkumista verkossa. Verkko on eteenpäinsyöttävä, sillä nuolet etenevät suoraviivaisesti syötekerroksesta tulostekerrokseen. Jos kyseessä olisi takaisinkytketty verkko, osa nuolista muodostaisi silmukoita edelliseen, tai samaan, kerrokseen.

Oppiminen neuroverkoissa tuotetaan oppimisalgoritmin avulla. Oppimisalgoritmilla tarkoitetaan menetelmää, joka muokkaa neuronien välisiä painoarvoja siten, että verkko tuottaa ratkaisun ongelmaan (Rojas 1996, s. 78). Oppiminen voidaan jakaa kahteen kategoriaan: ohjattuun ja ohjaamattomaan. Ohjatussa oppimisessa neuroverkko koulutetaan käyttäen tunnettuja syöte-tulostepareja, joiden avulla verkko saadaan vastaamaan syötteisiin halutulla tavalla. Eräs suosittu ohjatun oppimisen algoritmi on niin sanottu vastavirta-algoritmi (engl. *backpropagation*), joka hyödyntää gradienttimenetelmää painoarvojen optimoinnissa (Rumelhart, Hinton ja Williams 1986). Ohjaamattomassa oppimisessa esimerkkidataa ei ole tarjolla ja verkko yrittää toiminnallaan löytää ja klusteroida syötteestä löytyviä kuvioita (Kröse ja Van der Smagt 1996, s. 18). Itsenäisen luonteensa vuoksi ohjaamatonta oppimista hyödyntävien menetelmien tulokset eivät välttämättä aina ole toivotunlaisia. Menetelmien menestys riippuu niille syötetyn datan laadusta: parhaat tulokset saavutetaan, kun data on helposti luokiteltavissa.

3 Evoluutioalgoritmit

Evoluutiolaskenta on tietojenkäsittelyn alue, joka tutkii ja tuottaa algoritmeja, joiden toiminta on inspiroitunut luonnossa tapahtuvasta evoluutiosta. Näitä algoritmeja kutsutaan evoluutioalgoritmeiksi ja ne yleisesti jaetaan neljään eri menetelmään: evoluutio-ohjelmointi (engl. *evolutionary programming*), evoluutiostrategiat (engl. *evolution strategies*), geneettiseet algoritmit (engl. *genetic algorithms*) ja geneettinen ohjelmointi (engl. *genetic programming*) (Eiben ja Schoenauer 2002). Eri menetelmät ovat toiminnaltaan hyvin samankaltaisia ja niiden erot liittyvät lähinnä teknisiin yksityiskohtiin (Eiben ja Smith 2015, s. 27). Tyypillisesti evoluutioalgoritmeja käytetään koneoppimisessa ja optimoinnissa, mutta niitä hyödynnetään myös laitteiston suunnittelussa ja robotiikassa (Floreano ja Mattiussi 2008, s. 1). Tässä luvussa keskitytään esittelemään evoluutioalgoritmien yleistä toimintatapaa ja eri menetelmien välisiä eroja.

Seuraavaksi esitellään evoluutioalgoritmien yleinen kulku. Kappaleen sisältö perustuu Eibenin ja Smithin (2015, s. 25–27) esimerkkiin. Evoluutioalgoritmien toiminta ongelmienratkaisussa seuraa tyypillisesti yleistä mallia, joka voidaan jakaa kahteen päävaiheeseen. Ensimmäisessä päävaiheessa luodaan populaatio, joka sisältää sattumanvaraisesti luotuja ratkaisuehdokkaita. Luomisen jälkeen jokainen ratkaisuehdokkaista evaluoidaan kelpoisuusfunktion avulla, eli kokeillaan kuinka hyvän ratkaisun ongelmaan se tarjoaa. Ensimmäisen populaation ratkaisut ovat todennäköisesti varsin heikkoja ja testaaminen tässä vaiheessa vain luo kuvan lähtötilanteesta. Toinen päävaihe muodostaa silmukan, jota toistetaan kunnes löydetään tarpeet täyttävä ratkaisu. Silmukan kukin iteraatio koostuu viidestä vaiheesta. Ensimmäisessä vaiheessa muodostetaan populaatiosta sattumanvaraisesti pareja, joita kutsutaan vanhemmiksi. Toisessa vaiheessa valitut vanhemmat risteytetään, jolloin syntyy uusia ratkaisuehdokkaita, joita kutsutaan lapsiksi. Lapset perivät ominaisuuksia molemmilta vanhemmiltaan risteytysalgoritmin määrämällä tavalla. Ominaisuudet, jotka kultakin vanhemmalta lapselle siirretään, valitaan satunnaisesti. Kolmannessa vaiheessa mutatoidaan syntyneitä lapsia erilaisten variaatio-operaattorien avulla. Mutatoinnin tavoite on luoda lisää vaihtelua populaatioon. Neljännessä vaiheessa evaluoidaan syntyneet ratkaisuehdokkaat kelpoisuusfunktion avulla. Mikäli jokin ehdokkaista täyttää alussa ratkaisulle asetetut kriteerit,

keskeytetään algoritmin kulku tähän. Viidennessä, eli viimeisessä, vaiheessa valitaan populaatiosta yksilöt, jotka muodostavat seuraavan sukupolven. Valintamenetelmiä on erilaisia, ja niistä valitaan parhaiten käyttötarvetta palveleva. Menetelmä voi esimerkiksi suosia vahvimpia ehdokkaita, tai sisältää sattumanvaraisuutta. Valinnan jälkeen palataan silmukan alkuun ja aloitetaan risteytys ja mutatointi uudelle sukupolvelle. Algoritmi 1 esittää evoluutioalgoritmien kulun pseudokoodina.

Algoritmi 1 Evoluutioalgoritmien kulku (Eiben ja Smith 2015, s. 26)

```
1: Aloita
2:     Luo populaatio ratkaisuehdokkaita satunnaisilla parametreilla
3:     Evaluoi ratkaisuehdokkaat
4:     Toista
5:         Valitse vanhemmat, muodosta pareja
6:         Risteytä muodostetut parit
7:         Mutatoi risteytyksessä syntyneet lapset
8:         Evaluoi syntyneet ehdokkaat
9:         Valitse populaatiosta yksilöt, jotka muodostavat seuraavan sukupolven
10:    Kunnes tarpeet täyttävä ratkaisu löytynyt
11: Lopeta
```

Eroavaisuudet eri evoluutioalgoritmien välillä liittyvät niiden tekniseen toteutukseen. Tällä tarkoitetaan erityisesti valittuja tietorakenteita, joihin ratkaisuehdokkaat algoritmissa koodataan. Esimerkiksi geneettiset algoritmit hyödyntävät merkkijonoja, evoluutiostrategiat reaaliarvoisia vektoreita ja geneettinen ohjelmointi puurakenteita (Eiben ja Smith 2015, s. 28). Käytettävä evoluutioalgoritmi valitaan tyypillisesti sen käyttämän tietorakenteen perusteella, jotta ratkaisuehdokkaiden käsittely on mahdollisimman sujuvaa. Sopivan tietorakenteen valinta on tärkeää, sillä jotta kelpoisuutta voidaan mitata, täytyy ratkaisuehdokkaat dekoodata evoluutioalgoritmin käyttämästä muodosta. Jos tietorakenne on tähän huonosti soveltuva, voi jatkuva dekoodaaminen käydä laskennallisesti raskaaksi.

4 Neuroevoluutio ja sen soveltaminen

Neuroevoluutiolla tarkoitetaan koneoppimismenetelmää, jossa neuroverkkoja rakennetaan evoluutioalgoritmien avulla (Lehman ja Miikkulainen 2013). Käytännössä tämä tarkoittaa sitä, että neuroevoluutiossa evoluutioalgoritmin ratkaisuehdokkaat ovat neuroverkkoja. Ratkaisuehdokkaiden ominaisuudet, joita eri sukupolvien välillä muokataan, liittyvät neuroverkon toimintaan ja rakenteeseen. Evoluution avulla voidaan muokata esimerkiksi neuronien välisiä painoja, verkon topologiaa ja sen oppimissääntöjä (Yao 1999). Tyypillisesti muokkaus kuitenkin keskittyy painoihin ja topologiaan. Neuroevoluutioalgoritmin kulun aikana ratkaisuehdokkaita evaluoidaan, risteytetään ja mutatoidaan toistuvasti, kunnes tarpeet täyttävä neuroverkko on valmis. Muihin koneoppimismenetelmiin verrattuna neuroevoluutio on hyvin yleiskäyttöinen, sillä toimiakseen se vaatii vain, että kehitettävän neuroverkon suorituskykyä voidaan evaluoida ja sen käytöstä muuttaa evoluution avulla (Miikkulainen 2010).

Neuroevoluutioalgoritmeja voidaan erotella sen mukaan, millainen topologia algoritmin tuottamalla verkolla on. Yksinkertaisimmissa tapauksissa verkon topologia on ennalta määrätty, jolloin evoluution avulla optimoidaan arkkitehtuuriltaan samanlaisten verkkojen neuronien välisten yhteyksien painoja (Whiteson 2012). Topologian ennalta määräminen toteutetaan yrityksen ja erehdyksen avulla ihmiskäyttäjän toimesta (Yao 1999). Useimmissa tapauksissa topologian määrittäminen on kuitenkin hyvin vaikeaa, jopa mahdotonta, vaikka käyttäjä tuntisikin sovellusalueen hyvin (Whiteson 2012). Sen vuoksi mielenkiintoisempia ovat menetelmät, joissa painojen lisäksi myös verkon topologiaa, eli neuronien määrää ja niiden välisiä yhteyksiä, optimoidaan evoluution avulla. Tällaisissa menetelmissä evoluutio luonnollisesti pyrkii kohti optimaalista rakennetta ongelman ratkaisemiseksi ja ihmiskäyttäjän vaikutuksesta mahdollisesti syntyneet vääristymät vähenevät. Hyvä esimerkki algoritmista, joka kehittää sekä verkon painoja että topologiaa on Stanleyn ja Miikkulaisen (2002) suosittu NEAT-algoritmi. Toimivan topologian löytämisen lisäksi evoluution avulla voidaan myös monimutkaistaa jo suunnitellun neuroverkon rakennetta. Stanleyn ja Miikkulaisen (2004) mukaan topologian monimutkaistaminen evoluution avulla johtaa parempiin ratkaisuihin.

Neuroevoluutio soveltuu erityisen hyvin vahvistusoppimisongelmien ratkaisuun. Vahvistumisoppimisella tarkoitetaan koneoppimismenetelmää, jossa itsenäinen, oppiva agentti etsii

kokeilemalla millä toimilla se saavuttaa parhaan tuloksen ratkaistavaan ongelmaan (Sutton ja Barto 1998, s.1). Vahvistusoppimisessa agentti liikkuu valitsemillaan toimilla arvotetusta tilasta toiseen, tavoitteena maksimoida tilojen arvoista kasautuva palkinto (Woergoetter ja Porr 2008). Muihin vahvistusoppimisongelmien ratkaisumenetelmiin verrattuna neuroevoluutio hyötyy siitä, että se kykenee esittämään agentin tiloja jatkuvasti (Miikkulainen 2010). Neuroevoluutiossa käytetyt evoluutioalgoritmit tekevät agentin mahdollisten tilojen mallintamisesta helpompaa ja tarjoavat yksinkertaisen tavan ratkaista jatkuvia ja kooltaan suuria tilakokonaisuuksia (Whiteson 2012). Evoluutiota hyödyntäviä menetelmiä on kuitenkin kritisoitu niiden holistisesta näkemyksestä vahvistusoppimisongelmiin. Suttonin ja Barton (1998, s.9) mukaan evolutionaaristen menetelmien tapa keskittyä vain lopputulokseen, siivuttaen agentin valitsemat toimet ja sen kulkemat tilat, voi olla harhaanjohtava. He kuitenkin toteavat evolutionaaristen menetelmien toimivan muita menetelmiä paremmin tilanteissa, joissa agentti ei kykene havainnoimaan ympäristöään. Miikkulainen (2010) tukee tätä ajatusta ja toteaa neuroevoluution olevan vahva vaihtoehto tilanteissa, joissa kaikkea tietoa agentin ympäristöstä ei ole saatavilla, sillä se kykenee toimimaan vähäisellä vahvistuspalautteella.

Neuroevoluution soveltamisessa eri ongelmiin avainasemaan nousee sen yleiskäyttöisyys ja erityisen hyvä soveltuvuus vahvistusoppimisongelmien ratkaisuun. Yleiskäyttöinen luonne mahdollistaa myös evoluution ilman varsinaista tavoitetta. Luvussa 4.1 käsitellään evoluution suuntaamista ja neuroevoluution luovia ominaisuuksia. Neuroevoluution soveltuvuus erityisesti jatkuvien ja vain osittain havaitsettavissa olevien vahvistusoppimisongelmien ratkaisuun johtaa siihen, että menetelmä on hyödyllinen monissa reaalimaailman sovelluskohteissa. Menetelmää hyödynnetäänkin esimerkiksi fyysisten laitteiden ajo- ja hallintatehtävissä, sekä pelejä pelaavan tekoälyn ja keinoelämän luonnissa (Lehman ja Miikkulainen 2013). Luvuissa 4.2–4.4 annetaan esimerkkejä neuroevoluution hyödyntämisestä kolmessa eri sovelluskohteessa: syvien neuroverkkojen optimoinnissa, arvopapereiden kaupassa ja robotiikassa. Lopuksi luvussa 4.5 muodostetaan vielä yhteenveto neuroevoluutiosta ja sen soveltamisesta eri sovelluskohteisiin.

4.1 Evoluution ohjaaminen

Perinteisesti evoluutiolaskennassa eri ratkaisuehdokkaita evaluoidaan niin sanotun kelpoisuusfunktion avulla. Kelpoisuusfunktiolla pyritään mittaamaan ehdokkaan suoritusta sille annetun tehtävän ratkaisemisessa. Kun ratkaisuehdokkaita valitaan tällä tavoin, voidaan evoluutioprosessia kutsua tehtäväkeskeiseksi, sillä prosessi on suunnattu keskittymään nimenomaan tietyn tehtävän ratkaisuun. Evoluution suuntaamiselle on kuitenkin olemassa vaihtoehtoisia tapoja, joista yksi on ainutlaatuisuuden tavoittelu.

Lehman ja Stanley (2013) esittelevät tavan suunnata evoluutioprosessi tavoittelemaan ainutlaatuisuutta. Tämän kappaleen tiedot pohjautuvat edellä mainittuun artikkeliin. Esittelemässään menetelmässä Lehman ja Stanley korvaavat NEAT-algoritmissa kelpoisuusfunktion niin sanotulla ”novelty search” -algoritmilla, joka mittaa eri ratkaisuehdokkaiden käytöksen ainutlaatuisuutta. Evoluutioprosessin aikana syntyvien uusien ehdokkaiden käytöksen ainutlaatuisuutta mitataan vertaamalla sitä vanhempiin sukupolviin. Aivan kuten kelpoistuusfunktion mittaama kelpoisuus, myös ainutlaatuisuus tulee määritellä käyttötapauskohtaisesti. Esimerkiksi, jos tehtävänä on navigoida sokkelossa, ainutlaatuisena käytöksenä voidaan pitää sitä, että navigoiva agentti liikkuu sokkelossa uusille alueille.

Lehman ja Stanley (2013) väittävät ainutlaatuisuutta tavoittelevien menetelmien menestyvän tehtäväkeskeisiä paremmin tilanteissa, joissa tehtävä on hämäävä. Tehtävien hämäävyyteen liittyy olennaisesti tehtäväkeskeisten menetelmien tapa jumiutua lokaaliin optimiin. Lehman ja Stanley (2013) antavat esimerkin menetelmänsä menestyksestä kaksijalkaisen liikkumisen kehittämisessä evoluution avulla: ainutlaatuisuutta tavoitteleva menetelmä etsii aktiivisesti erilaisia tapoja tuottaa liikettä, kun tehtäväkeskeinen menetelmä jumiutuu lokaaliin optimiin ja palkitsee ratkaisuehdokkaita, jotka vain kaatuvat pisimmälle. Ainutlaatuisuutta tavoittelevien menetelmien teho perustuukin juuri siihen, että ne tutkivat toimialueensa laajemmin, kehittäen näin monipuolisempia ratkaisuja ongelmaan. Lehman ja Stanley (2013) myöntävät, ettei menetelmä kuitenkaan aina tarjoa optimaaleja ratkaisuja, ja että sen kykyä löytää monipuolisia ratkaisuja voidaan hyödyntää yhteistyössä perinteisten, tehtäväkeskeisten menetelmien kanssa. Samaa ajatusta tukevat Cuccu ja Gomez (2011), joiden mukaan suurilla toimialueilla ainutlaatuisuutta tavoittelevat menetelmät eivät yksinään menesty tehtäväkeskeisiä paremmin, mutta niitä voidaan hyödyntää diversiteettiä lisäävinä tekijöinä kelpoisuut-

ta mittaavien menetelmien apuna.

Esimerkiksi evoluutiorobotiikan (engl. *evolutionary robotics*) ja keinoelämän tutkimuksessa hyödynnetään ainutlaatuisuutta tavoittelevia menetelmiä (Doncieux ym. 2015; Lehman ja Stanley 2011). Näillä aloilla hyödynnetään avointa evoluutiota (engl. *open-ended evolution*), eli evoluutioprosessia jatketaan pitkään, ilman varsinaista määränpäätä. Koska ratkaistavaa tehtävää ei ole, on luonnollista keskittyä maksimoimaan diversiteettiä populaatioissa ja luomaan variaatiota ratkaisuehdokkaiden käytöksessä. Evoluution kehittämät ratkaisut ovat monesti odottamattomia laajuudessaan ja monimutkaisuudessaan (Lehman ym. 2018). Näiden ratkaisujen tutkimuksesta voidaan saada arvokkaita ideoita, joita ilman evoluution apua ei koskaan löydettäisi.

4.2 Syvien neuroverkkojen optimointi

Kuten aiemmin on todettu, neuroverkkojen tehokkuuteen vaikuttaa käyttäjän määrittämä verkon arkkitehtuuri, jonka määrittely voi olla hyvin vaikeaa. Ongelmat korostuvat erityisesti syvissä neuroverkoissa, joissa useat piilotetut kerrokset tekevät verkon topologiasta monimutkaisen ja vaikeasti määritettävän. Miikkulainen et al. (2017) ovat luoneet automatisoidun CoDeepNEAT-menetelmän, jonka avulla syvien neuroverkkojen arkkitehtuuri voidaan optimoida evoluution avulla. Tässä luvussa esiintyvät asiat pohjautuvat edellä viitattuun artikkeliin.

Menetelmä perustuu aiemmin mainittuun NEAT-algoritmiin, josta on luotu syville neuroverkoille paremmin soveltuva variaatio: DeepNEAT. NEAT-suvun algoritmeissa ratkaisuehdokkaita kutsutaan kromosomeiksi. Kuhunkin noodeista koostuvaan kromosomiin on koodattu yhden neuroverkon tiedot. Alkuperäisessä algoritmissa kromosomin yksittäinen noodi viittaa verkon yhteen neuroniin. DeepNEAT-variaatiossa yksi noodi kuitenkin viittaa yhteen verkon kerrokseen, mikä mahdollistaa evoluution syville verkoille. Kelpoisuusmittauksia varten jokainen kromosomi muutetaan neuroverkoksi, jota koulutetaan tietty aika. Koulutuksen jälkeen tieto neuroverkon suorituskyvystä palautetaan takaisin algoritmille, jotta sitä voidaan hyödyntää uuden sukupolven valinnassa. Yksin DeepNEAT-algoritmin luomat neuroverkot eivät kuitenkaan saavuta huomiota herättäviä tuloksia. Tämän vuoksi menetelmää

on jatkettu, ottaen mallia kilpailussa menestyneiden syvien neuroverkkojen rakenteesta. Tällaisten verkkojen, kuten GoogleNetin, rakenne pohjautuu toistuviin moduuleihin (Szegedy ym. 2015). Jatkettu menetelmä on nimeltään CoDeepNEAT. Siinä DeepNEAT-menetelmään lisätään ominaisuus kehittää moduuleja ja verkon "pohjapiirroksia", mikä mahdollistaa monipuolisempien ja syvempien arkkitehtuurien luomisen.

Miikkulaisen et al. (2017) mukaan menetelmän avulla on mahdollista muodostaa monimutkaisempia verkon arkkitehtuureja kuin käsin. Artikkelissa esitellään menetelmän avulla luotujen verkkojen menestystä kuvantunnistustehtävissä, ja saavutetut tulokset ovat verrattavissa parhaimpiin käsin suunniteltuihin verkkoihin. Syvien neuroverkkojen opettaminen on kuitenkin laskennallisesti raskasta ja evoluutioprosessin voidaan joutua kouluttamaan tuhansia verkkoja. Ryhmä toteaakin, että erityisesti syvien neuroverkkojen koulutus evoluution avulla hyötyy tulevaisuudessa kasvavasta laskentatehosta ja arvioi, että laskentatehon kasvaessa evoluution avulla kehitetyt neuroverkot ohittavat käsin suunnitellut kilpakumppanit.

4.3 Arvopaperikauppa

Arvopaperikauppa on suosittu sovellusala tekoälyn saralla, sillä kaupankäynnissä menestyvä tekoäly voi johtaa mittaviin taloudellisiin tuottoihin. Azzini ja Tettamanzi (2008) esittelevät menetelmän, jossa neuroevoluutioalgoritmin avulla voidaan toteuttaa automatisoitua kaupankäyntiä. Tässä luvussa esitellyt asiat perustuvat lähinnä edellä viitattuun artikkeliin.

Azzinin ja Tettamanzin esittelemä menetelmä muistuttaa rakenteeltaan Yaon ja Liun (1997) EPNET-algoritmia, joka kehittää sekä verkon painoja että topologiaa evoluution avulla. Menetelmässä luodaan evoluution avulla topologia ja parametrit neuroverkolle, joka määrittää halutaanko tiettyä osaketta myydä vai ostaa pörssin auetessa. Verkon syötteenä käytetään osakkeen edellispäivän hintatietoja (avaus-, korkein-, matalin ja lopetushinta) ja osakkeiden teknisessä analyysissä käytettyjä teknisiä indikaattoreita. Menetelmää testattiin osakekauppa simulaattorilla, joka sisälsi aitoa dataa valituista osakkeista.

Azzini ja Tettamanzi korostavat menetelmänsä minimalisuutta, sillä toimiakseen se vaati verrattain vähän dataa: vain edellä mainitut syötteet. Heidän mukaansa, varovasti toteuttettuna menetelmällä voidaan saavuttaa merkittävä tuotto, vaikkakin siihen sisältyvät riskit ovat suu-

rempia kuin mitä keskiverto sijoittaja on valmis ottamaan. Riskiä voidaan kuitenkin vähentää ajamalla algoritmi useita kertoja, mahdollisesti eri asetuksilla, ja valita suoritettava operaatio tulosten moodin perusteella. Testeissä menetelmän saavuttamat tulokset vastasivat noin 39.8 % vuosittaista tuottoa.

4.4 Robotiikka

Robotiikka on monipuolinen ala, jolle löytyy sovelluskohteita elämän jokaiselta osa-alueelta. Tässä esimerkissä käsitellään robottikäden toimintaa, josta on hyötyä esimerkiksi automaatiossa ja tilanteissa, joissa vaaditaan ihmismäistä näppäryyttä ympäristössä, joka on ihmiselle haitallinen. Huang, Lehman, Mok, Miikkulainen ja Sentis (2014) esittelevät menetelmän, jossa robottikäsi opetetetaan tarttumaan erilaisiin esineisiin neuroevoluution avulla. Tässä luvussa esitellyt asiat perustuvat edellä mainittuun artikkeliin.

Huangin et al. menetelmässä robotti määrittelee itsenäisesti tarttumistehtävään vaadittavat asetukset käden asennolle, orientaatiolle ja sijainnille käyttäen Kinect-sensorista saatua kolmiulotteista dataa. Robotin toimintaa helpotetaan siten, että ennen kuin robotti visualisoi tilanteen sensorinsa avulla, määrittelee ihmisohjaaja rajoittavan laatikon tartuttavan esineen ympärille, jotta robotin on helpompi kohdistaa esineeseen. Robottikättä ohjaa neuroverkko, joka kehitetään NEAT-algoritmin avulla. Aluksi populaatio koostuu neuroverkoista, joiden kaikki syöteneuroneit on yhdistetty yhteen piilotettuun neuroniin, joka on yhdistetty verkon seitsemään tulosteneuroniin. Evoluutioprosessin aikana, verkkojen mutatoituesssa, verkon topologia voi muuttua uusien yhteyksien ja neuronien muodossa. Neuroverkon syötteet koostuvat Kinect-sensorin tuottamasta syvyysdatasta, josta jokaiselle pikselille on määrätty tietty syöteneuroni. Tulosteena verkko palauttaa kolmiulotteista dataa käden asennosta, orientaatiosta ja sijainnista. Neuroverkon kelpoisuutta mitataan sen ohjaaman otteen perusteella: mitä paremman otteen käsi ottaa esineestä, sitä parempi ratkaisu on. Otteen laadun mittaamiseen käytettiin Millerin ja Allenin (1999) kuvaamaa menetelmää, joka perustuu kämmenen ja esineen välisiin kosketuspintoihin.

Neuroevoluution ohjaamaa robottikättä testattiin simulaatiossa ja fyysisellä robottikädellä. Tarttumista testattiin useita kertoja eri muotoisilla esineillä, sekä ihmisohjaajan määrämän

rajoittavan laatikon kanssa että ilman sitä. Simulaatiossa, ilman ohjaavaa laatikkoa, robottikäsi onnistui tarttumaan esineestä keskimäärin noin 63 % tarkkuudella. Ohjaavan laatikon kanssa käden onnistusmisprosentti simulatiossa oli keskimäärin noin 81 %. Simulaatiossa opitut taidot siirtyivät myös reaalimaailmaan, sillä simulaatiossa kehitetyllä neuroverkolla kyettiin tarttumaan uudenlaisiin esineisiin myös fyysisellä robottikädellä. Otteen laatua ei voitu kuitenkaan mitata samalla tavoin kuin simulaatiossa, sillä fyysisen käden kosketuksen painetta ei kyetty vielä mittaamaan.

4.5 Yhteenveto

Edellä mainitut esimerkit muodostavat kuvan neuroevoluutiosta monipuolisena ratkaisuna erilaisiin ongelmiin. Erityisen mielenkiintoinen on esimerkki syvien neuroverkkojen optimoinnista. Syvät neuroverkot ovat olleet viime vuosina suosittu puheenaihe ja suosio on levinnyt myös neuroevoluutiotutkimukseen. Alan tuoreimmat tutkimustulokset liittyvät erityisesti evoluutioalgoritmien soveltamiseen syviin neuroverkkoihin (Miikkulainen ym. 2017; Such ym. 2017). Laskentatehon kasvaessa, syvien neuroverkkojen arkkitehtuurin kehittämisestä evoluution avulla voi muodostua merkittävä tutkimusala, sillä se poistaa neuroverkkojen kouluttamisesta topologian määrittelyvaiheen, joka on usein erittäin haastava. Miikkulaisen et al. (2017) mukaan evoluution avulla luodut verkot tulevat todennäköisesti menestymään käsinsuunniteltuja paremmin. Lisäksi evoluutioalgoritmit hyötyvät erityisen paljon laskentatehon kasvusta. Esimerkiksi evoluutiostrategiat soveltuvat erityisen hyvin rinnakkaislaskentaan ja sitä hyödyntävät menetelmät ovatkin helposti skaalattavissa (Salimans ym. 2017).

Esimerkeissä korostuu myös neuroevoluutioalgoritmien yleiskäyttöisyys. Kolmessa esimerkissä ratkaisut perustuivat samaan NEAT-algoritmiin, vaikka niiden sovellusalat olivat hyvin erilaiset. Yleiskäyttöisyyteen tulee kuitenkin suhtautua maltillisesti. Neuroevoluutiomenetelmät menestyvät parhaiten, kun evoluutioprosessin tuomat hyödyt ovat suurempia kuin sen laskentatehovaatimuksiin liittyvät haitat. Esimerkiksi yksinkertaisissa ohjatun oppimisen ongelmissa neuroverkojen painojen määrittely on nopeampaa toteuttaa vastavirta-algoritmin avulla. Toisaalta, erityisesti vahvistusoppimisongelmissa neuroevoluutiolla on selvä etu muihin menetelmiin verrattuna.

5 Johtopäätökset

Tutkielman tavoitteena oli muodostaa kuva neuroevoluutiosta ja sen käyttökelpoisuudesta. Selvisi, että vaikka neuroevoluutio on menetelmänä monipuolinen ja yleiskäyttöinen, sen käyttö jokaikisen ongelman ratkaisussa ei ole tarkoituksenmukaista. Mikäli ratkaistava ongelmana on luonteeltaan yksinkertainen ja sen ratkaisuun käytettävä neuroverkko on helposti suunniteltavissa käsin, on epätodennäköistä, että evoluution avulla tuotettu ratkaisu menestyisi paremmin. Neuroevoluutio menestyy erityisesti vahvistusoppimisongelmissa, joissa tarvitaan neuroevoluution kykyä jatkuvaan esitykseen ja oppimiseen vajaavaisesta datasta. Tämän vuoksi menetelmä sopii erityisesti reaalimaailman ajo- ja hallintatehtäviin.

Neuroevoluution heikkoutena voidaan pitää sen tarvitsemaa laskentatehoa, joka korostuu erityisesti syvien neuroverkkojen optimoinnissa. Yhden syvän neuroverkon opettamisessa voi kestää tunneista päiviin, riippuen käytössä olevasta laitteistosta. Evoluutioprosessin aikana voidaan kouluttaa jopa tuhansia neuroverkkoja, mikä voi koitua kalliiksi. Tilannetta kuitenkin pehmentää alati kasvava laskentateho, sekä mahdollisuus käyttää pilvilaskentaa ja muita hajautettuja laskentatapoja. Lisäksi eräät evoluutioalgoritmit soveltuvat erinomaisesti rinnakkaislaskentaan, jolloin laskentatehon kasvu hyödyttää niitä suoraviivaisesti.

Tulevaisuudessa on todennäköistä, että neuroevoluution käyttö yleistyy erilaisissa hybridimenetelmissä. Kuten aiemmin todettiin, neuroverkon arkkitehtuurin määrittäminen on erittäin haastava tehtävä. Jos tehokkaan arkkitehtuurin voi määrittää automaattisesti evoluution avulla, säästetään paljon aikaa. Tämä korostuu erityisesti syvissä neuroverkoissa, joiden rakenne voi olla hyvinkin monimutkainen ja tehokkaan ratkaisun löytäminen käsin on vaikeaa. Useimmissa tapauksissa neuronien välisten painoarvojen määrittelyyn on kuitenkin olemassa suoraviivaisempia keinoja, jotka toimivat evoluutiomenetelmiä tehokkaammin. Siksi on todennäköistä, että tulevaisuudessa yleistyvät menetelmät, joissa verkkojen rakenne optimoidaan evoluution avulla, mutta niiden varsinainen kouluttaminen tapahtuu suoraviivaisempien menetelmien kautta.

Neuroevoluutiolla on myös luovia ominaisuuksia, jotka tulevat erityisesti esiin avointa evoluutiota käytettäessä. Ainutlaatuisuutta tavoitteleva evoluutioprosessi on kerännyt suosiota

esimerkiksi evoluutiorobotiikan ja keinoelämän tutkimuksessa. Evoluution tuottamat, monimutkaisuudessaan ja laajudessaan ainutlaatuiset, ratkaisut antavat tutkijoille uusia ideoita. Näiden ideoiden avulla voidaan oppia ymmärtämään biologista elämää ja oppimista paremmin.

Kirjallisuuskatsausta tehdessä on käynyt selväksi, että neuroevoluutiota on tutkittu pitkään pienen, mutta aktiivisen tutkijayhteisön toimesta. Yhteisön koko tulee ilmi kirjallisuuteen tutustuessa, sillä samat nimet ja instituutiot toistuvat useissa julkaisuissa. Viime aikoina tekoälytutkimus on liittynyt erityisesti syväoppimismenetelmiin. Tämä näkyy myös neuroevoluutiotutkimuksessa, sillä alan tuoreimmat tutkimustulokset liittyvät erityisesti syviin neuroverkkoihin. Onkin todennäköistä, että kasvava panostus tekoälytutkimukseen näkyy tulevaisuudessa myös tällä tutkimuksenalalla yhteisön ja rahoituksen kasvuna. Ilmiö tulee todennäköisesti vain kiihtymään, mikäli neuroevoluutiota edelleen menestyksekkäästi yhdistetään syväoppimiseen.

Lähteet

Azzini, A., ja A. G. B. Tettamanzi. 2008. “Evolving Neural Networks for Static Single-position Automated Trading”. *Journal of Artificial Evolution and Applications* 2008:1–17. doi:10.1155/2008/184286.

Cuccu, G., ja F. Gomez. 2011. “When novelty is not enough”. Teoksessa *European Conference on the Applications of Evolutionary Computation,* 234–243. Springer.

Doncieux, S., N. Bredeche, J.-B. Mouret ja A. E. Eiben. 2015. “Evolutionary robotics: what, why, and where to”. *Frontiers in Robotics and AI* 2 (4). doi:10.3389/frobt.2015.00004.

Eiben, A. E., ja M. Schoenauer. 2002. “Evolutionary computing”. *Information Processing Letters* 82:1–6.

Eiben, A. E., ja J. E. Smith. 2015. *Introduction to Evolutionary Computing*. Berlin: Springer.

Floreano, D., ja C. Mattiussi. 2008. *Bio-inspired artificial intelligence: theories, methods, and technologies.* Cambridge, MA: MIT press.

Huang, P.-C., J. Lehman, A. K. Mok, R. Miikkulainen ja L. Sentis. 2014. “Grasping novel objects with a dexterous robotic hand through neuroevolution”. Teoksessa *Computational Intelligence in Control and Automation (CICA), 2014 IEEE Symposium on,* 1–8. IEEE.

Kröse, B., ja P. Van der Smagt. 1996. *An introduction to neural networks.* Amsterdam: University of Amsterdam.

Lehman, J., J. Clune, D. Misevic, C. Adami, J. Beaulieu, P. J. Bentley, S. Bernard, G. Belson, D. M. Bryson, N. Cheney ym. 2018. “The Surprising Creativity of Digital Evolution: A Collection of Anecdotes from the Evolutionary Computation and Artificial Life Research Communities”. *arXiv preprint arXiv:1803.03453.*

Lehman, J., ja R. Miikkulainen. 2013. “Neuroevolution”. Revision #133684, *Scholarpedia* 8 (6): 30977. doi:10.4249/scholarpedia.30977.

Lehman, J., ja K. O. Stanley. 2011. “Abandoning Objectives: Evolution Through the Search for Novelty Alone”. *Evol. Comput.* 19:189–223. doi:10.1162/EVCO_a_00025.

Miikkulainen, R. 2010. “Neuroevolution”. Teoksessa *Encyclopedia of Machine Learning.* New York: Springer.

Miikkulainen, R., J. Liang, E. Meyerson, A. Rawal, D. Fink, O. Francon, B. Raju, A. Navruzyan, N. Duffy ja B. Hodjat. 2017. “Evolving deep neural networks”. *arXiv preprint arXiv:1703.00548.*

Miller, A. T, ja P. K. Allen. 1999. “Examples of 3D grasp quality computations”. Teoksessa *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on,* 2:1240–1246. IEEE.

Mozur, P. 2017. “Beijing wants A.I. to be Made in China by 2030”. *New York Times* (20. heinäkuuta). https://nyti.ms/2tjoNlY.

Rabesandratana, T. 2018. “Emmanuel Macron wants France to become a leader in AI and avoid ’dystopia’”. *Science* (30. maaliskuuta). doi:10.1126/science.aat7491.

Rojas, R. 1996. *Neural Networks: A Systematic Introduction.* Berlin: Springer.

Rumelhart, D. E., G. E. Hinton ja R. J. Williams. 1986. “Learning representations by back-propagating errors”. *Nature* 323:533–536.

Salimans, T., J. Ho, X. Chen ja I. Sutskever. 2017. “Evolution strategies as a scalable alternative to reinforcement learning”. *arXiv preprint arXiv:1703.03864.*

Stanley, K. O., ja R. Miikkulainen. 2002. “Evolving Neural Networks Through Augmenting Topologies”. *Evolutionary Computation* 10:99–127.

———. 2004. “Competitive Coevolution through Evolutionary Complexification”. *Journal of Artificial Intelligence Research* 21:63–100.

Such, F. P., V. Madhavan, E. Conti, J. Lehman, K. O. Stanley ja J. Clune. 2017. “Deep Neuroevolution: Genetic Algorithms Are a Competitive Alternative for Training Deep Neural Networks for Reinforcement Learning”. *arXiv preprint arXiv:1712.06567.*

Sutton, R. S., ja A. G. Barto. 1998. *Reinforcement learning: An introduction.* Cambridge, MA: MIT press.

Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke ja A. Rabinovich. 2015. “Going Deeper With Convolutions”. Teoksessa *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Whiteson, S. 2012. “Evolutionary Computation for Reinforcement Learning”. Teoksessa *Reinforcement Learning: State-of-the-Art,* toimittanut M. Wiering ja M. van Otterlo, 325–355. Berlin: Springer. doi:10.1007/978-3-642-27645-3_10.

Woergoetter, F., ja B. Porr. 2008. “Reinforcement learning”. Revision #91704, *Scholarpedia* 3. doi:10.4249/scholarpedia.1448.

Yao, X. 1999. “Evolving artificial neural networks”. *Proceedings of the IEEE* 87:1423–1447. doi:10.1109/5.784219.

Yao, X., ja Y. Liu. 1997. “A new evolutionary system for evolving artificial neural networks”. *IEEE transactions on neural networks* 8:694–713.