Vahvistusoppimisen käyttö kaupankäyntialgoritmeissa ja näiden testaus

Turun yliopisto
Tietotekniikan laitos
TkK-tutkielma
Tietotekniikka
Kesäkuu 2025
Sisu Vepsäläinen

TURUN YLIOPISTO
Tietotekniikan laitos

Sisu Vepsäläinen: Vahvistusoppimisen käyttö kaupankäyntialgoritmeissa ja näiden testaus

TkK-tutkielma, 21 s.
Tietotekniikka
Kesäkuu 2025

Tutkielmassa tutkitaan algoritmista osakekaupankäyntiä, joka on laajasti käytetty menetelmä rahoitusalalla. Algoritmisella osakekaupankäynnillä pyritään maksimoimaan tuotot ja minimoimaan riskit. Tutkielmassa selvitetään, miten algoritminen kaupankäynti eroaa ihmisen tekemästä osakekaupankäynnistä.
Tutkielmassa selvitetään eri vahvistusoppimismenetelmien hyödyntämistä osakekaupankäynnissä sekä testaamista ja mittareita. Vahvistusoppimisen perusperiaatteet käydään läpi, ja tutustutaan keskeisiin menetelmiin. Näitä ovat Soft Actor-Critic (SAC), Proximal Policy Optimization (PPO), Deep Deterministic Policy Gradient (DDPG), Advantage Actor-Critic (A2C) ja Q-oppiminen. Näiden toimintaa ja soveltuvuutta osakekaupankäyntiin tarkastellaan.
Tulosten perusteella vahvistusoppimista voidaan hyödyntää osakekaupankäynnissä kouluttamaan agentteja, jotka toimivat portfolion optimoinnissa, kaupankäynnissä, toimeksiantojen toteutuksessa sekä yhdistelmästrategiassa. Testauksessa voidaan käyttää toteumatestausta ja eri mittareita, kuten Sharpe-lukua, joilla voidaan mitata eri tavalla agentin toimivuutta. Mittareiden avulla voidaan testata agentin eri ominaisuuksia esimerkiksi volatiliteettia tai maksimilaskua. Toteumatestauksessa agentteja voidaan testata ilman pelkoa rahan menettämisestä.

Asiasanat: Q-oppiminen, Deep Deterministic Policy Gradient, Advantage Actor-Critic, Proximal Policy Optimization, Vahvistusoppiminen, Algoritminen kaupankäynti

Sisällys

1 Johdanto

Tässä luvussa esitellään tutkimuksen tausta, tutkimuskysymykset ja tavoitteet.

1.1 Tutkimuksen rajaus

Tässä kandidaatintutkielmassa tarkastellaan vahvistusoppimisen hyödyntämistä osakekaupankäynnissä ja osakekaupankäynnin testaamista. Mallit, joita tarkastellaan, ovat Soft Actor-Critic (SAC), Proximal Policy Optimization (PPO), Deep Deterministic Policy Gradient (DDPG), Advantage Actor-Critic (A2C) ja Q-oppiminen. Vahvistusoppimismalleja on olemassa muitakin, mutta tässä tutkielmassa keskitytään näihin.

1.2 Tutkimuskysymykset

Tutkimuksessa käsitellään kahta keskeistä tutkimuskysymystä osakekaupankäynnin algoritmeihin liittyen:

1. Miten vahvistusoppimismenetelmiä voidaan hyödyntää sijoittamisessa?
2. Millaisia menetelmiä ja mittareita voidaan käyttää osakekaupankäynnin agenttien testauksessa ja arvioinnissa?

1.3 Tutkimusmenetelmät ja lähteet

Tieteelliset lähteet on haettu IEEE:n ja ACM:n tietokannoista. Haussa on käytetty hakulauseketta "algorithm* trad* AND autom* AND 'stock market' AND reinforcement learning"ja hakutulokset on rajattu vuosina 2019-2024 tehtyihin julkaisuihin, jotta saadaan uusimmat tiedot. Hakutuloksista lähteet on valittu otsikoiden ja tiivistelmien perusteella. Lähteinä on myös käytetty löydetyissä lähteissä käytettyjä lähteitä. Täydentäviä lähteitä on haettu hakusanoilla: Soft Actor-Critic, Proximal Policy Optimization, Advantage Actor-Critic, Q-learning, Deep Deterministic Policy Gradient ja backtesting.

2 Algoritminen kaupankäynti

Algoritminen kaupankäynti on yksi viimeaikaisista merkittävistä rahoitusalan osa-alueista. Sitä voidaan pitää strategiana, joka automaattisesti tekee peräkkäisiä kaupankäyntipäätöksiä. [1] Algoritmisessa kaupankäynnissä tietokoneohjelma tekee kaupankäyntiä noudattaen ennalta määrättyjä sääntöjä [2] [3]. Algoritminen kaupankäynti on laajalti käytetty menetelmä rahoitusmarkkinoilla, ja sitä voidaan käyttää automatisoimaan osakkeiden, optioiden, futuurien ja kryptovaluuttojen osto- ja myyntitoimeksiannot. [4] Algoritmista kaupankäyntiä voidaan myös kutsua automatisoiduksi kaupankäynniksi [2] tai agenttipohjaiseksi kaupankäynniksi [1]. Algoritmisen kaupankäynnin tavoitteena on samanaikaisesti maksimoida tuotot ja vähentää riskejä [1][5].

Algoritmista kaupankäyntiä on esitetty ratkaisuksi osakemarkkinoiden epävarmuuden ja arvojen nopean vaihtelun aiheuttamiin haasteisiin. Epävarmuus osaketrendeissä tekee tuottojen saavuttamisesta vaikeaa ihmisten luomien sääntöjen avulla. Osakearvojen nopeat ja jatkuvat vaihtelut tekevät reaaliaikaisen reagoinnin haastavaksi ihmiselle, ja lisäksi tunteiden aiheuttamat virheet voivat johtaa tappioihin. [2] Algoritminen kaupankäynti toimii ilman tunteita ja ennakkoluuloja, hyödyntäen mahdollisuuksia välittömästi niiden ilmetessä [3]. Algoritminen kaupankäynti päihittää ihmisten tekemän kaupankäynnin monella tasolla, kuten kaupankäynnin vakaudessa, tunteiden vaikutuksen vähenemisessä sekä luovemmassa ja sopeutuvammassa käyttäytymisessä [1].

1980-luvulla alkanut osakemarkkinoiden automaatio mahdollisti reaaliaikaisen tiedon hankinnan ja automatisoidun toimeksiantojen tekemisen. Nämä loivat pohjan algoritmiselle kaupankäynnille, jota alettiin tehdä 1990-luvulla. [6] Algoritmiset kaupankäyntijärjestelmät kattavat laajan kirjon, aina perusteknisiin indikaattoreihin pohjautuvista yksinkertaisista järjestelmistä monimutkaisempiin järjestelmiin, jotka analysoivat laajoja tietoaineistoja ja hyödyntävät tekoälyä sekä koneoppimisalgoritmeja ennustaakseen tulevia hinnan muutoksia [3]. Klassiset algoritmiset strategiat sisältävät muun muassa saapumishintastrategian (engl. arrival price strategy), volyymipainotetun keskihinnan strategian (engl. volume-weighted average price strategy), aikapainotetun keskihinnan strategian (engl. time-weighted average price strategy), toteutusvajestrategian (engl. implementation shortfall strategy) ja sissistrategian (engl. guerrilla strategy)[4]. Olemassa olevassa kirjallisuudessa on esitetty monia kaupankäyntistrategioita, kuten esimerkiksi keskiarvon palautumiseen perustuva strategia (engl. mean reversion). Osa tutkijoista on kehittänyt menetelmiä, joiden avulla kaupankäynnin sääntöjä voidaan luoda. Ennustusmenetelmät, kuten yhdistelmämallioppiminen (engl. ensemble learning), ovat puolestaan suunniteltu ennustamaan ensin tulevia hintasuuntauksia, minkä jälkeen kaupankäyntipäätökset tehdään näiden ennusteiden perusteella. [6]

3 Vahvistusoppiminen kaupankäyntialgoritmeissa

Tässä luvussa käsitellään vahvistusoppimisen periaatteita ja niiden soveltamista kaupankäyntialgoritmeihin. Vahvistusoppiminen on koneoppimisen alalaji, jossa agentti oppii ihmisten tavoin aikaisemmista tapahtumista eri ympäristöissä [7].

3.1 Vahvistusoppimisen perusperiaatteet

Vahvistusoppiminen oppii kokeilun ja epäonnistumisen kautta samoin kuin ihmiset [8]. Vahvistusoppiminen tekee päätöksiä ja saa lopputuloksen perusteella palkkion, jonka perusteella se muuttaa toimintaansa [7]. Huonosta päätöksestä tulee negatiivinen palkkio ja hyvästä positiivinen. Palkkio vaikuttaa seuraavaan päätökseen. [9] Vahvistusoppimisessa agentti siirtyy tekemään enemmän päätöksiä, jotka ovat johtaneet positiiviseen palkkioon ja vähemmän päätöksiä, jotka ovat johtaneet negatiiviseen palkkioon. Vahvistusoppimiseen liittyy nämä 8 käsitettä: agentti, tila, toiminta, ympäristö, palkkio, arvo, politiikka ja malli [8].

Agentti (engl. agent) on toimija, joka toimii tietyssä ympäristössä pyrkien saavuttamaan asetetun tavoitteen [8] [10]. Agentti oppii aikaisemmista kokemuksista ja niiden avulla päivittää politiikkaa tulevaisuuden toimintoja varten [7].

Tila (engl. state, s) on osa ympäristöä, jonka agentti tuntee [10]. Tila on joukko numeraalisia tai symbolisia ominaisuuksia, jotka määrittävät agentin nykyisen

kontekstin [8]. Agentti tekee toiminnot tässä kontekstissa [7].

Toiminto (engl. action, a) on joukko kaikesta toiminnasta mitä agentti voi suorittaa missä tahansa tilassa [8]. Toimintojen avulla agentti vuorovaikuttaa ympäristön kanssa [11].

Ympäristö (engl. environment) sisältää tilat, agentin, ongelman ja lopullisen tavoitteen, jonka agentin tulisi saavuttaa. Ympäristö myös palkitsee agentin ja antaa ehdotuksen seuraavasta tilasta. [8]

Palkkio (engl. reward, r) on välitön numeerinen vastaus ympäristöltä agentille jokaisen toiminnon jälkeen. Positiivinen palkkio merkitsee, että toiminto oli hyvä ja negatiivinen, että toiminto oli huono. [8] Palkinto mahdollistaa tehokkaampien strategioiden oppimisen agentille [11].

Arvo (engl. value) on numeerinen määrä, jonka agentti pitää tallessa ja pyrkii maksimoimaan. Toisin kun palkinto, arvo ei ole vain yhden toiminnon vastaus vaan se ulottuu useiden toimintojen yli. [8]

Politiikka (engl. policy) on looginen kehys, jonka perusteella agentti toimii. Se rakentuu ajan myötä, eikä ole tiedossa etukäteen. [8]

Malli (engl. model) on käytössä vain muutamassa vahvistusoppimismenetelmässä. Malli on vahvistusoppimisen ymmärrys ympäristöstä. [8]

3.2 Vahvistusoppimisen tekninen toiminta

Useimmat vahvistusoppimisongelmat kunnioittavat Markovin ominaisuutta ja täten niihin voi hyödyntää Markovin päätösprosessin (engl. Markov decision process, MDP) matemaattista rakennetta

$$M = (S, A, P, R). \tag{3.1}$$

Yhtälössä (3.1) S on tila-avaruus ja A on joukko toimintoja. $P : S \times A \times S \to [0, 1]$ on siirtymäfunktio, joka määrittää todennäköisyyden siirtyä tilaan s', kun agentti on tilassa s ja valitsee toiminnon a.$R : S \times A \to \mathbb{R}$ on palkintofunktio, joka määrittää palkinnon arvon, jonka agentti saa valittuaan toiminnan a ollessaan tilassa s. [9] Matemaattisessa rakenteessa voi olla myös diskonttauskerroin γ [12]. Diskonttauskerroin määrittää kompromissin pitkän aikavälin palkintojen ja välittömien palkintojen välillä [2]. Diskonttauskerroin on tärkeä konvergenssin kannalta [13]. MDP:n ratkaisu on politiikka, joka valitsee toiminnan jokaiselle tilalle. Politiikan π arvo $v^{\pi}(s)$ tilassa s saadaan, kun lasketaan summa palkinnoista, jonka agentti saa seuratessaan politiikkaa π tilasta s alkaen. Jokaisella MDP:llä on olemassa optimaalinen politiikka π^*. Optimaalinen politiikka maksimoi odotetun summan jokaiselle ensimmäiselle tilalle. [9]

Osakemarkkinaa voi kuvata MDP:n avulla seuraavasti. Tila $s = [\mathbf{p}, \mathbf{h}, b]$ vektorina, jossa $\mathbf{p} \in \mathbb{R}+^D$ on osakkeiden hinnat, $\mathbf{h} \in \mathbb{Z}+^D$ on osakkeiden määrät ja $b \in \mathbb{R}+$ on jäljellä oleva saldo. D merkitsee kuinka monta eri osaketta on ja $\mathbb{Z}+$ on ei-negatiiviset kokonaisluvut. Toiminto a on vektori toiminnoista D:lle eri osakkeelle. Toiminnot ovat myynti, osto ja pitäminen. Myynti vähentää osakkeen määrää h, osto lisää osakkeiden määrää h ja pitäminen pitää osakkeiden määrän ennallaan. Palkkio $r(s, a, s')$ on välitön palkkio, kun tehdään toiminto a tilassa s ja päädytään uuteen tilaan s'. Politiikka $\pi(s)$ on kaupankäyntistrategia tilassa s eli toimintojen todennäköisyysjakauma tilassa s. Q-arvo $Q_{\pi}(s, a)$ on odotettu palkkio toiminnosta a tilassa s, kun seurataan politiikka π. [13] Vahvistusoppimisen avulla pyritään löytämään paras politiikka eli kaupankäyntistrategia, joka tuo parhaat tuotot.

Koulutus Q-oppimista sovellettaessa

Aseta kouluttamismäärä ja iteraatiomäärä

Aseta tavoitetila $s_{tavoite}$

Alusta ympäristö

Alusta mallin parametrit:

Alusta Q-taulukko Q

Alusta tutkimuskerroin α

Alusta diskonttauskerroin γ

Alusta epsilon ϵ

while t on pienempi kuin kouluttamismäärä **do**

Alusta alkutila s_t

while i on pienempi kuin iteraatiomäärä **do**

Valitse toiminto a_t:

$a_t \leftarrow$ Epsilon-ahne strategia (3.2)

Suorita toiminto a_t ympäristössä

Saa palkinto r_t ja seuraava tila s_{t+1}

if $s_{t+1} = s_{\text{tavoite}}$ **then**

break

Päivitä Q-taulukkoa:

$$Q(s_t, a_t) = (1 - \alpha)Q(s_t, a_t) + \alpha((R_{t+1} + \gamma * \ maxQ(s_{t+1}, a)))$$

Päivitä tila:

$s_t \leftarrow s_{t+1}$

i++

t++

return Q-taulukko

[14]

Q-oppiminen (engl. Q-learning)

Q-oppiminen on mallivapaa vahvistusoppimisalgoritmi. Algoritmi oppii oikean käyttäytymisen toistuvan agentin ja ympäristön välisen vuorovaikutuksen myötä. [14]

Algoritmin tarkoitus on iteratiivisesti hienosäätää Q-arvoa, joka on arvio toiminnan a_t oikeellisuudesta, jokaisessa tilassa s_t ajassa t. Joka tilassa on monta toimintoa ja näistä optimaalisin on se, jonka Q-arvo on suurin. [8] Kouluttautumisvaiheessa voidaan käyttää epsilon-ahne (engl. epsilon-greedy) strategiaa, jolloin satunnaisesti valitaan muu toiminto kuin se, jolla on suurin Q-arvo. Epsilon-ahneella strategialla pyritään välttämään jumiutuminen paikalliseen optimiin. [14] Strategia voidaan kuvata yhtälöllä (3.2)

$$a_t = \begin{cases} \arg\max Q(s,a), & \text{jos } \sigma > \varepsilon \\ a_{\text{rand}}, & \text{jos } \sigma \leq \varepsilon \end{cases}, \tag{3.2}$$

missä σ on tutkimuskerroin väliltä 0-1. Kun ε on pienempi kuin σ niin valitaan toiminto, jolla suurin Q-arvo ja muuten valitaan satunnainen toiminto [14].

Q-arvoa päivitetään Bellman-yhtälön (3.3) mukaan jokaisella ajanhetkellä [8].

Bellman-yhtälö:

$$Q\left(s_t, a_t\right) = (1-\alpha)Q\left(s_t, a_t\right) + \alpha\left(\left(R_{t+1} + \gamma * \ maxQ\left(s_{t+1}, a\right)\right)\right) \tag{3.3}$$

[8] missä α on oppimisnopeus, γ on diskonttauskerroin, s_t on tila ajassa t, a_t on tehty toiminta ajassa t, R_{t+1} on palkinto, joka on saatu tehdystä toiminnosta a_t ajassa t, maxQ(s_{t+1}, a) on maksimi Q-arvo, jonka seuraavasta tilasta s_{t+1} voi saada millä tahansa toiminnolla. Q(s_t, a_t) on Q-arvo toiminnolle a_t tilassa s_t ajassa t. [8] Oppimisnopeus saa arvon väliltä 0-1 ja se määrittää kuinka paljon Q-arvoja muokataan toiminnon jälkeen [15]. Suurempi oppimisnopeus johtaa algoritmin nopeampaan konvergoitumiseen, mutta pienentää todennäköisyyttä optimaaliselle ratkaisulle [14]. Diskonttauskerroin on arvo 0-1 väliltä, joka määrittää tulevaisuuden palkkioiden merkityksen Q-arvojen päivityksessä. Pienemällä arvolla agentti keskittyy lähitulevaisuuden tuottoihin ja suuremmalla arvolla pidemmällä tulevaisuudessa

oleviin tuottoihin. [15]

Etu-toimijakriitikko (engl. Advantage Actor-Critic, A2C)

A2C yhdistää toimijakriitikko-mallin ja etuustiedon hyödyntämisen neuroverkkojen avulla [3]. A2C on yleisesti käytetty toimijakriitikkopohjainen vahvistusoppimisalgoritmi, joka tehostaa politiikan päivityksiä gradienttimenetelmän avulla [7]. A2C:ssä toimija valitsee toiminnon tilan perusteella, ja kriitikko arvioi valitut toiminnot maksimoidakseen kertyvät tuotot. A2C käyttää etuustietoa vähentääkseen politiikkagradientin varianssia ja lisätäkseen vakautta. [3] A2C onkin hyvä malli osakekaupankäyntiin vakautensa vuoksi [13]. Etuustiedon $A(s_t, a_t)$ tehtävä on mitata toiminnon ylivoimaisuutta odotetuissa palkkioissa ja se voidaan laskea kahdella tavalla. [3] Laskukaavat ovat esiteltyinä yhtälössä (3.5) ja (3.6).

A2C-algoritmin tavoitefunktio on:

$$\nabla J_\theta(\theta) = \mathbb{E}\left[\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) A(s_t, a_t)\right] \tag{3.4}$$

missä $\pi_\theta(a_t|s_t)$ on politiikan verkko, ja etuustieto $A(s_t, a_t)$. [13]

$$A(s_t, a_t) = Q(s_t, a_t) - V(s_t) \tag{3.5}$$

Yhtälössä (3.5) (s_t, a_t) edustaa arvioitua toiminto-arvofunktiota. [3]

$$A(s_t, a_t) = r(s_t, a_t, s_{t+1}) + \gamma V(s_{t+1}) - V(s_t) \tag{3.6}$$

Yhtälössä (3.6) $r(s_t, a_t, s_{t+1})$ on välitön palkkio, jonka agentti saa tehdessään toiminnon a_t tilassa s_t ja siirtymällä tilaan s_{t+1}. [3]

A2C algoritmissa monet agentit samanaikaisesti vuorovaikuttavat itsenäisesti ympäristön kanssa ja laskevat gradientteja käyttäen eri datanäytteitä. Gradientit keskiarvoistetaan koordinaattorin toimesta ja laitetaan globaaliin verkostoon, joka

päivittää toimija- ja kriitikkoverkostot. Usean agentin käyttäminen monipuolistaa koulutusaineistoa ja mahdollistaa vakaampaa ja tehokkaampaa oppimista. [3] [13] A2C:ssä odotetaan, että kaikki agentit ovat valmiita, minkä jälkeen verkostot päivitetään ja lähetetään agenteille [16].

Syvä Deterministinen Politiikkagradientti (engl. Deep Deterministic Policy Gradient, DDPG)

DDPG voidaan jakaa nimensä perusteella kolmeen osaan, jotka myös kertovat sen ominaisuudet. Syvä (engl. deep) kertoo neuraaliverkkojen käytöstä, deterministinen (engl. deterministic) kertoo, että DDPG:n ulostulo on deterministinen toiminto, jota voidaan käyttää jatkuvassa ympäristössä ja politiikkagradientti (engl. policy gradient) kertoo, että se käyttää politiikkaverkkoa. [17] DDPG on vahvistusoppimismalli, joka yhdistää Q-oppimisen ja politiikan gradientin [3] [7] [13]. DDPG oppii suoraan havaintojen perusteella politiikkagradientin avulla. Menetelmässä pyritään kartoittamaan tilat deterministisesti toimintoihin, mikä tukee paremmin jatkuvan toimintotilan ympäristöjä. [13] DDPG optimoi tuotot jatkuvan toiminnon ympäristössä [3] ja on tehokas käsittelemään jatkuvan toiminnon tilaa, jonka takia se soveltuu hyvin osakekaupankäyntiin [13].DDPG pystyy suoraan käsittelemään jatkuvia toimintoja agentin tilavuorovaikutusten pohjalta ja tallentaa nämä toistopuskuriin (engl. replay buffer) R [3]. Tarkemmin jokaisella ajanhetkellä t, DDPG-agentti tekee toiminnon a_t tilassa s_t, josta saa palkinnon r_t ja siirtyy tilaan s_{t+1}. Nämä siirtymät (s_t, a_t, s_{t+1}, r_t) tallennetaan toistopuskuriin R. Toistopuskurista R valitaan satunnaisesti N siirtymää, joiden avulla päivitetään Q-arvo y_i yhtälöllä (3.7). [13]

$$y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1} \mid \theta^{\mu'}, \theta^{Q'})), \quad i = 1, \ldots, N. \tag{3.7}$$

Yhtälössä (3.7) γ on diskonttaustekijä, $\theta^{\mu'}$ kuvaa kohdepolitiikkaverkon μ' parametreja ja $\theta^{Q'}$ kuvaa kohde-kritiikkiverkon Q' parametreja. Kohde Q-arvo y_i arvioi odotetun kumulatiivisen palkkion nykyiselle tila-toimintoparille. [3] Kriitikkoverkkoa päivitetään minimoimalla häviöfunktio $L(\theta^Q)$, joka on arvioitu ero kohdekriitikkiverkon Q' ja kritiikkiverkon Q ulostulojen välillä [13].

Proksimaalinen Politiikan Optimointi (engl. Proximal Policy Optimization, PPO)

PPO on politiikkasidonnainen vahvistusoppimisalgoritmi, joka tarjoaa vakautta ja luotettavuutta [18]. Se on myös toimijakriitikkorakenteen mukainen. Toimijaverkko antaa toiminnon tietyssä tilassa ja kriitikkoverkko arvioi tilan arvon, mikä vaikuttaa toimijaverkon suorituskykyyn. [19] PPO vakauttaa politiikan kouluttamista rajoittamalla politiikan päivitystä, joka kouluttautumisaskeleella [7] [13]. Rajoittamiseen PPO käyttää luottamusalueen menetelmää, joka varmistaa, että muutokset politiikkaan eivät ole liian suuria [3] [20]. Todennäköisyyssuhde vanhan ja uuden politiikan välillä voidaan ilmaista yhtälöllä (3.8) [13].

$$r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}. \tag{3.8}$$

Yhtälö (3.9) on PPO:n leikattu approksimoitu tavoitefunktio [13].

$$J^{\text{CLIP}}(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta)\hat{A}(s_t, a_t), \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right) \hat{A}(s_t, a_t) \right) \right]. \tag{3.9}$$

Yhtälössä (3.9) $r_t(\theta)\hat{A}(s_t, a_t)$ on tavallinen politiikkagradientin tavoitefunktio, ja $\hat{A}(s_t, a_t)$ on arvioitu hyötyfunktio. Funktio $\text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)$ leikkaa suhdeluvun $r_t(\theta)$ arvojen $[1-\epsilon, 1+\epsilon]$ välille. PPO:n tavoitefunktio valitsee minimoidun arvon leikatun ja tavallisen tavoitteen välillä. [13]

PPO:n etu on helpossa toteutuksessa ja tehokkaassa näytteiden hyödyntämisessä [20].

Pehmeä toimijakriitikko (engl. Soft Actor-Critic, SAC)

SAC on off-policy-algoritmi, jota on ehdotettu ratkaisemaan todelliseen maailmaan soveltamisen rajoituksia [18]. Se optimoi stokastisia politiikkoja sekamuotoisille toimintotiloille, mikä tekee siitä ihanteellisen tutkimusintensiivisiin tehtäviin, kuten robotiikkaan [3]. Stokastisen luonteensa vuoksi SAC voi tehdä parempaa tutkimista ympäristössään, minkä vuoksi se ei jumitu paikalliseen maksimiin [21]. SAC yhdistää toimijakriitikko- ja maksimientropiaoppimisen maksimoidakseen sekä kumulatiivisen palkkion, että politiikan entropian edistääkseen monipuolista toimintaetsintää [3]. Tämä myös edistää häiriönkestävyyttä [12]. SAC:n tavoitefunktio voidaan ilmaista yhtälöllä (3.10) [3].

$$J(\pi) = \mathbb{E}\left[\sum_{t=i}^{T} \gamma^t \left(r(s_t, a_t) + \alpha H(\pi(s_t))\right)\right]. \tag{3.10}$$

Yhtälössä π edustaa politiikkaa, $r(s_t, a_t)$ on hetkellinen palkkio, α on lämpötilaparametri, joka tasapainottaa palkkion ja entropian maksimoinnin, ja $H(\pi(s_t))$ on politiikan todennäköisyysjakauman entropia. [3]

SAC hyödyntää kahta Q-verkkoa tilanne-toimintoarvojen estimointiin. Aktoriverkko päivitetään uudelleenparametrisoinnilla, mikä mahdollistaa tehokkaan gradienttilaskennan. Kriitikkoverkot koulutetaan minimoimaan keskineliöllinen Bellmanvirhe, kuten muissa toimijakriitikko-menetelmissä. Lämpötilaparametri α säädetään dynaamisesti hallitsemaan tutkimisen ja hyödyntämisen tasapainoa. Sitä voidaan päivittää esimerkiksi entropian lämpötilan säädöllä, jossa α pienenee ajan myötä painottaen enemmän palkkion maksimointia [3].

SAC:n parametrit päivitetään vuorottelemalla kahden vaiheen välillä: pehmeä politiikan arviointi ja pehmeä politiikan parantaminen. Politiikan arviointivaihees-

sa pehmeä Q-funktio, joka mallinnetaan neuroverkolla, päivitetään minimoimalla pehmeä Bellman-jäännös. Jatkuvien toimintatilojen käsittelemiseksi politiikka mallinnetaan gaussilaisena jakaumana, jonka keskiarvo ja kovarianssi määräytyvät neuroverkoista [12].

SAC käyttää toistopuskuria off-policyn muotoiluun. Toistopuskuriin tallennetaan havaintojoukot muodossa $(S_t, A_t, R_t + 1, S_t + 1)$, missä S_t on nykyinen tila, A_t suoritettava toiminto tilassa S_t, R_t+1 toiminnosta saatu palkinto ja S_t+1 saavutettu tila, kun toiminto A_t on tehty. Mallin kouluttaminen alkaa vasta, kun toistopuskurissa on riittävästi dataa. [21]

3.3 Agenttien hyödyntäminen

Kaupankäynti

Agentti koulutetaan tekemään kaupankäyntipäätöksiä eli osta, pidä ja myy [9]. Kaupankäynti eroaa toimeksiantojen toteuttamisesta siten, että kaupankäynnissä agentti tekee päätökset siitä, että mitä tehdä, kun toimeksiantojen toteuttamisessa agentti pyrkii vain saamaan ihmisen asettaman toimeksiannon läpi parhaalla hinnalla.

Kaupankäynti voidaan erotella kauppojen pituuden perusteella viiteen kategoriaan, jotka ovat positiokaupankäynti (engl. position trading), swing-kaupankäynti (engl. swing trading), päiväkaupankäynti (engl. day trading), skalppauskaupankäynti (engl. scalp trading) ja suurtaajuuskaupankäynti (engl. high-frequency trading, HFT). Positiokaupankäynnissä osaketta pidetään kuukausista vuosiin. Swing-kaupankäynnissä pyritään hyödyntämään trendejä ja osakkeita pidetään päivistä viikkoihin. Päiväkaupankäynnissä pyritään hyödyntämään päivän sisäisiä liikkeitä ja positiot suljetaan päivän päätteeksi yön yli -riskin välttämiseksi. Skalppauksessa osakkeita pidetään sekunneista minuutteihin ja HFT:ssä mikrosekunneista sekunteihin. [12]

Kaupankäynnin tavoitteena on maksimoida odotettu tuotto. Odotetun tuoton

maksimoimisella pyritään kehittämään voitollinen strategia ja päihittämään vaihtoehtoiset strategiat. [3]

Portfoliohallinta ja optimointi

Portfoliohallinnassa sijoittajat omistavat useita rahoitusinstrumentteja ja jakavat niitä uudelleen säännöllisesti maksimoidakseen pitkäaikaisen tuoton. Tätä kutsutaan myös portfolio-optimoinniksi, portfolion valinnaksi ja portfolion allokoinniksi. [12]

Portfolio-optimisoinnilla pyritään minimoimaan riskit ja maksimoimaan tuotot [9]. Portfoliohallinnassa täytyy ottaa huomioon monia tekijöitä, kuten esimerkiksi riski, tuotot ja likviditeetti [22]. Portfoliohallintaan liittyvät usean osakkeen osta-, pidä- ja myy-päätösten hallinta [8].

Vahvistusoppimisen hyödyntämistä portfoliohallinnassa tutkineet ovat keskittyneet käyttämään toimijakriitikkomenetelmiä, sekä niiden paranneltua versiota: DDPG-menetelmää. Nämä tutkimukset ovat kaikki tuottaneet tuloksia, jotka päihittävät markkinan. [8]. Markkinan päihittäminen tarkoittaa, että menetelmä saavuttaa paremman tuoton kuin laajasti käytetty vertailuindeksi, kuten SP 500 - indeksi, joka koostuu Yhdysvaltojen 500 suurimmasta pörssiyhtiöstä.

Toimeksiantojen toteuttaminen

Toimeksiantojen toteuttamisen tavoitteet on täyttää koko toimeksianto, sekä pyrkiä taloudellisempaan toteutukseen maksimoimalla tuotot tai minimoimalla kustannukset [12]. Toimeksiantojen toteuttamisessa päätöksenteko pysyy ihmisellä, mutta toimeksiannon teko on automatisoitu, jotta suuret sijoitukset jaetaan pieniin osiin, jotta osto tai myyntihintaan ei vaikuteta. [6]

Yhdistelmästrategia

Yhdistelmästrategiassa käytetään useaa agenttia vakaan kaupankäyntistrategian rakentamiseen. [13] Yhdistelmästrategia on suunniteltu mukautumaan erilaisiin markkinaolosuhteisiin sekä tehostamaan agentin yleistä tehokkuutta. [3] Yhdistelmästrategiaan voidaan valita halutut vahvistusoppimismenetelmät. Yhdistelmästrategia koostuu 3:sta vaiheesta.

Ensimmäisessä vaiheessa agentit koulutetaan [3] [13]. Agentit koulutetaan n ajanjakson datalla, missä n on ajanjakso kuukausina. Agentit uudelleenkoulutetaan halutuin aikavälein, jotta varmistettaisiinniiden sopeutuminen muuttuviin markkinadynamiikkoihin ja jotta uusin data sisällytettäisiin koulutukseen. [3]

Toisessa vaiheessa agentit validoidaan ja valitaan parhaiten menestyvä. Agentteja voidaan verrata laskemalla jokaiselle Sharpe-luku. Korkeimman Sharpe-luvun omaava valitaan. [3] [13] Riskin välttämistä hienosäädetään ottamalla validointivaiheessa huomioon turbulenssi-indeksi, joka mittaa markkinoiden vaihtelua ja auttaa arvioimaan agenttien suorituskykyä eri markkinaskenaarioissa. [3]

Kolmannessa vaiheessa annetaan parhaan agentin käydä osakekauppaa [3] [13]. Yhdistelmästrategia perustuu siihen, että eri agentit menestyvät eri markkinatrendeissä [3]. Eräs agentti voi toimia hyvin härkäisessä trendissä, eli markkinnassa, jossa hinnat nousevia, mutta toimia huonosti karhutrendissä, eli markkinan laskiessa pidempään. Toinen agentti voi olla parempi volatiilisessa markkinassa, jossa hinnan vaihtelut ovat suuret lyhyellä aikavälillä. [13] Yhdistelmästrategia pyrkii hyödyntämään hajauttamisen etuja, samalla hyödyntäen kunkin agentin vahvuuksia. Strategia pyrkii optimoimaan tuotot huomioimalla riskit, jotka liittyvät korkeimpiin tuottoihin. [3]

4 Kaupankäyntialgoritmien testaus ja suorituskyvyn arviointi

Tässä luvussa tarkastellaan vahvistusoppimismenetelmillä koulutettujen agenttien suorituskyvyn arviointiin käytettäviä mittareita ja toteumatestausta, jolla agentteja voidaan testata historiallisella datalla. Alaluvussa 4.1 käydään läpi eri mittareita, mitä ne mittaavat ja kaavat, joilla ne voidaan laskea. Alaluvussa 4.2 käydään läpi mitä toteumatestaus on.

4.1 Mittarit

Mittarit ovat agenttejen testaamista varten. Mittareita voidaan käyttää toteumatestauksessa mittaamaan agentin suoriutumista eri tavalla. Esimerkiksi volatiliteetin avulla voidaan arvioida kuinka paljon salkun arvo heittelee, kokonaistuoton avulla kuinka paljon nousua voidaan odottaa ja maksimilaskun avulla, mikä on suurin mahdollinen tappio, kun agentti tekee osakekauppaa. Alla käsitellään joitain mittareita ja niiden laskukaavat.

Kokonaistuotto (engl. Total return, TR) ilmaisee tuoton tietyllä aikavälillä ja voidaan laskea kaavalla (4.1).

$$TR = (Q_{[T]} - Q_{[0]})/Q_{[0]}, \tag{4.1}$$

missä $Q_{[0]}$ on alkusijoitus ja $Q_{[T]}$ portfolion arvo sijoitusjakson lopussa. [4]

Riskiarvo (engl. Value at risk, VaR) arvioi portfolion maksimihäviön tietyllä ajanjaksolla määritetyllä luottamustasolla.

Sharpe-luku (engl. Sharpe ratio, SR1) on yleisesti käytetty mittari, joka yhdistää riskin ja tuoton. [13] Se on mittari riskikorjatulle tuotolle ja kertoo kuinka paljon tuotto riskiyksikköä kohden ylittää riskittömän tuoton. Positiivinen Sharpe-luku kertoo, että salkku kasvaa enemmän, kuin riskitön korkotuotto. Sharpe-luku voidaan laskea kaavalla(4.2). [4]

$$SR = (E[R_p] - R_f)/\omega_p, \tag{4.2}$$

missä $E[R_p]$ on portfolion odotettu tuotto, R_f on riskitön korkotuotto ja ω_p portfolion tuoton keskihajonta. Korkeampi Sharpe-luku tarkoittaa pienempää riskiä [23].

Sortino-luku (engl. Sortino ratio, SR2) on riskikorjattu mittari määrittämään lisätuotto jokaiselle lisätylle riskiyksikölle. Toisin kuin Sharpe-luvussa, Sortino-luvussa käytetään keskihajonnan sijaan alapuolista osittaishajontaa erottamaan suotuisat ja haitalliset vaihtelut toisistaan. Sortinon luku voidaan laskea kaavalla(4.3). [4]

$$SR = (E[R_p] - R_f)/DR, \tag{4.3}$$

missä DR on salkun tuoton alapuolinen osittaishajonta. [4] Korkeampi Sortino-luku tarkoittaa pienempää riskiä [23].

Maksimi lasku (engl. Maximum Drawdown, MDD) laskee suurimman laskun kaupankäyntiajanjakson aikana. Sen avulla voidaan mitata huonoin skenaario. [12] Korkeampi MDD tarkoittaa korkeampaa riskiä [23]. MDD:n määritelmä on

$$MDD = \max_{\tau \in (0,t)} \left[\max_{t \in (0,\tau)} \left(\frac{n_t - n_\tau}{n_t} \right) \right], \tag{4.4}$$

missä n on kumulatiivinen tuotto [23].

Calmar-suhde (engl. Calmar ratio, CR) on eräs toinen riskikorjatun tuoton mittari. Siinä käytetään MDD:tä riskinä. [12] Calmar-suhde voidaan laskea kaavalla(4.5)

$$CR = (E[R_p] - R_f)/MDD \tag{4.5}$$

Volatiliteetti (engl. Volatility, VOL) on tuottovektori r:n varianssi. Sitä käytetään mittamaan tuottojen epävarmuutta ja se kuvaa strategian riskitasoa. [12] VOL:n määritelmä on

$$VOL = \sigma[r] \tag{4.6}$$

Tuotto-tappiosuhde (engl. Gain-Loss ratio GLR) on tappioriskimittari. Se esittää voitollisten ja tappiollisten kauppojen suhdetta. [12] GLR määritelmä on

$$GLR = \frac{\mathbb{E}[r \mid r > 0]}{\mathbb{E}[-r \mid r < 0]}, \tag{4.7}$$

missä r on tuotto tai tappio.

4.2 Toteumatestaus

Toteumatestaus (engl. Backtesting) on menetelmä, jossa algoritmia testataan historiallisella datalla ja mitataan tulokset [8]. Toteumatestauksessa strategiaa voidaan testata ilman, että käytetään oikeaa rahaa sijoittamiseen. Tärkein olettamus toteumatestauksen taustalla on, että historiallisella datalla hyvin suoriutunut strategia suoriutuu hyvin myös tulevaisuudessa.[24] Toteumatestauksen avulla voidaan verrata eri malleja niiden tuottaman voiton kautta koulutus- ja testidatalla [8]. Toteumatestauksessa käytetään Sharpe-lukua ja muita mittareita [11].

Toteumatestausta voidaan tehdä kahdella eri tavalla. Nämä ovat tapahtumapohjainen toteumatestaus, joka on samankaltainen kuin aito kaupankäynti ja huomioi

mm. kulut, sekä vektorisoitu toteumatestaus, joka approksimoi aitoa kaupankäyntiä. Vektorisoitu toteumatestaus on nopeampaa, mutta heikkoutena siinä on liikaa muuttujia huomioitavina ja mahdolliset harhat. [7]

Toteumatestauksen avulla voidaan testata strategian toimivuutta eri markkinaolosuhteissa, strategian tehokkuutta ja mahdollisia heikkouksia. [25] Hyvin toteutettu toteumatestaus hyvillä tuloksilla luo luottamusta käyttää strategiaa jatkossa. Huonojen tulosten tapauksessa on mahdollista hylätä strategia tai tehdä siihen muutoksia. [24]

5 Johtopäätös

Vahvistusoppimismenetelmiä on useita erilaisia, ja niitä voidaan hyödyntää sijoittamisessa eri tehtäviin. Vahvistusoppimismenetelmillä voidaan kouluttaa agentteja tekemään portfolion hallintaa, toimeksiantojen toteutusta ja kaupankäyntiä. Lisäksi on vielä yhdistelmästrategia, jossa koulutetaan useita agentteja ja näistä valitaan testien perusteella paras, joka käy kauppaa tietyn ajanjakson ajan.

Toteumatestaus on tärkeä työkalu testaamaan koulutettujen agenttien toimivuutta markkinoilla ilman riskejä. Toteumatestauksessa voidaan käyttää valittuja mittareita mittaamaan analyysissä olennaisia muuttujia. Kokonaistuotto varmasti kiinnostaa joka testauskerralla, mutta lisäksi voidaan esimerkiksi testata maksimilaskua tai volatiliteettia tietyn jakson aikana. Toteumatestauksen avulla voidaan testata agentin suoriutumista erilaisissa tilanteissa

Mielenkiintoista olisi tehdä jatkotutkimusta, jossa koulutettaisiin useita agentteja samalla datalla eri tehtäviin eri vahvistusoppimismenetelmiä hyödyntäen. Toteumatestauksella ja eri mittareilla agentteja voisi sitten vertailla eri markkinaolosuhteissa ja eri tehtävissä sen selvittämiseksi, mikä menetelmä toimii parhaiten minkäkin tehtävän oppimisessa ja millaisessa markkinassa.

Lähdeluettelo

[1] B. Hirchoua, I. Mountasser, B. Ouhbi ja B. Frikh, ”Evolutionary Deep Reinforcement Learning Environment: Transfer Learning-Based Genetic Algorithm”, teoksessa *The 23rd International Conference on Information Integration and Web Intelligence*, New York, NY, Yhdysvallat: Association for Computing Machinery, 2022, s. 242–249. DOI: 10.1145/3487664.3487698.

[2] Y. Ansari, S. Yasmin, S. Naz ym., ”A Deep Reinforcement Learning-Based Decision Support System for Automated Stock Market Trading”, 2022. url: https://ieeexplore.ieee.org/document/9969608 (viitattu 18. 11. 2024).

[3] I. Varshini Devi, B. Natarajan, S. Prabu, R. A. Praba, K. Ushanandhini ja K. S. Guruprakash, ”Automated Stock Trading using Reinforcement Learning”, teoksessa *2023 International Conference on Integrated Intelligence and Communication Systems (ICIICS)*, Kalaburagi, Intia: IEEE, 2023, s. 1–6. DOI: 10.1109/ICIICS59993.2023.10421071.

[4] B. Jin, ”A Mean-VaR Based Deep Reinforcement Learning Framework for Practical Algorithmic Trading”, 2023. url: https://ieeexplore.ieee.org/document/10076454 (viitattu 18. 11. 2024).

[5] A. Bhakar, P. S. Deori, Y. V. Gautam ja S. KG, ”Maximizing Returns with Reinforcement Learning: A Paradigm Shift in Stock Market Portfolio Management”, teoksessa *TENCON 2023 - 2023 IEEE Region 10 Conference (TENCON)*,

Chiang Mai, Thaimaa: IEEE, 2023, s. 393–398. DOI: 10.1109/TENCON58879.2023.10322511.

[6] B. Itri, Y. Mohamed, Q. Mohammed, B. Omar ja T. Mohamed, ”Deep reinforcement learning strategy in automated trading systems”, teoksessa *2023 3rd International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*, Mohammedia, Marokko: IEEE, 2023, s. 1–8. DOI: 10.1109/IRASET57153.2023.10152925.

[7] J. Garza, ”Reinforcement Learning for Stock Option Trading”, teoksessa *2023 31st Irish Conference on Artificial Intelligence and Cognitive Science (AICS)*, Letterkenny, Irlanti: IEEE, 2023, s. 1–4. DOI: 10.1109/AICS60730.2023.10470596.

[8] G. S. Bharadwaj, D. Pratap ja N. Darapaneni, ”Optimized Automated Stock Trading using DQN and Double DQN”, teoksessa *2024 International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS)*, Hassan, Intia: IEEE, 2024, s. 1–9. DOI: 10.1109/IACIS61494.2024.10721810.

[9] G. D. Santos ja K. R. Lima, ”Impact of combinatorial optimization on reinforcement learning for stock trading in financial markets”, teoksessa *Proceedings of the 20th Brazilian Symposium on Information Systems*, New York, NY, Yhdysvallat: Association for Computing Machinery, 2024, s. 1–8. DOI: 10.1145/3658271.3658282.

[10] Z. BELDI ja M. Bessedik, ”Innovative Integration of Q-learning in BSO Algorithm for Adaptive Community Detection in Social Networks”, 2024. url: https://doi.org/10.1145/3707702 (viitattu 22. 03. 2025).

[11] J. GE, Y. QIN, Y. Li, y. Huang ja H. Hu, ”Single stock trading with deep reinforcement learning: A comparative study”, teoksessa *Proceedings of the 2022*

14th International Conference on Machine Learning and Computing, Guangzhou, Kiina: Association for Computing Machinery, 2022, s. 34–43.

[12] S. Sun, R. Wang ja B. An, "Reinforcement Learning for Quantitative Trading", 2023. url: https://doi.org/10.1145/3582560 (viitattu 30. 10. 2024).

[13] H. Yang, X.-Y. Liu, S. Zhong ja A. Walid, "Deep reinforcement learning for automated stock trading: an ensemble strategy", 2021. url: https://doi.org/10.1145/3383455.3422540 (viitattu 28. 10. 2024).

[14] Y. Gao, Y. Li ja Z. Guo, "A Q-learning based UAV Path Planning Method with Awareness of Risk Avoidance", teoksessa *2021 China Automation Congress (CAC)*, Bejing, Kiina: IEEE, 2021, s. 669–673.

[15] J. Schilperoort, I. Mak, M. M. Drugan ja M. A. Wiering, "Learning to Play Pac-Xon with Q-Learning and Two Double Q-Learning Variants", teoksessa *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, Bangalore, Intia: IEEE, 2018, s. 1151–1158.

[16] G. Kyriakides ja K. G. Margaritis, "Neural Architecture Search with Synchronous Advantage Actor-Critic Methods and Partial Training", teoksessa *Proceedings of the 10th Hellenic Conference on Artificial Intelligence*, Patras, Kreikka: Association for Computing Machinery, 2018, s. 1–7.

[17] X. Zhang ja Y. Xu, "Improvement of prioritized experience replay mechanism based on deep deterministic policy gradient algorithm", teoksessa *Proceedings of the 2023 6th International Conference on Artificial Intelligence and Pattern Recognition*, Xiamen, Kiina: Association for Computing Machinery, 2024, s. 1310–1316.

[18] N. Lee ja J. Moon, "Offline Reinforcement Learning for Automated Stock Trading", 2023. url: https://ieeexplore.ieee.org/document/10285085 (viitattu 28. 10. 2024).

[19] Y. Sun, X. Yuan, W. Liu ja C. Sun, ”Model-Based Reinforcement Learning via Proximal Policy Optimization”, teoksessa *2019 Chinese Automation Congress (CAC)*, Hangzhou, Kiina: IEEE, 2019, s. 4736–4740.

[20] J. Wang, X. Wang, W. Du, Y. Li, C. Tang ja G. Liu, ”Proximal Policy Optimization Algorithm for Integrated Energy System Operation with Adaptive Learning Rate Decay Strategy”, teoksessa *2023 4th International Conference on Smart Grid and Energy Engineering (SGEE)*, Zhengzhou, Kiina: IEEE, 2023, s. 545–549.

[21] M. Shil, G. N. Pillai ja M. K. Gupta, ”Improved Soft Actor-Critic: Reducing Bias and Estimation Error for Fast Learning”, teoksessa *2023 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, Bhopal, Intia: IEEE, 2023, s. 1–6.

[22] J. Yu, X. Lu, A. Pan ja J. Shi, ”Stock trading strategy based on reinforcement learning with GRU network”, teoksessa *Proceedings of the 2024 3rd International Conference on Cyber Security, Artificial Intelligence and Digital Economy*, Nanjing, Kiina: Association for Computing Machinery, 2024, s. 480–485. DOI: 10.1145/3672919.3673005.

[23] Y.-F. Chen, W.-Y. Shih, H.-C. Lai, H.-C. Chang ja J.-L. Huang, ”Pairs Trading Strategy Optimization Using Proximal Policy Optimization Algorithms”, teoksessa *2023 IEEE International Conference on Big Data and Smart Computing (BigComp)*, Jeju, Etelä-Korea: IEEE, 2023, s. 40–47.

[24] V. S. P. Bhamidipati ja D. Saisanthiya, ”Stock Price Prediction using Random Forest Classifier and Backtesting”, teoksessa *2023 International Conference on Computer Communication and Informatics (ICCCI)*, Coimbatore, Intia: IEEE, 2023, s. 1–8.

[25] G. Carrascal, P. Hernamperez, G. Botella ja A. d. Barrio, ”Backtesting Quantum Computing Algorithms for Portfolio Optimization”, 2024. url: `https://ieeexplore.ieee.org/document/10329473` (viitattu 26. 03. 2025).