Kanditutkielma

Tietojenkäsittelytieteen kandiohjelma

Transformer-mallit ja niiden soveltaminen aikasarjojen analyysiin

Teemu Ruokokoski

15.12.2024

Matemaattis-luonnontieteellinen tiedekunta

Helsingin yliopisto

Yhteystiedot

PL 68 (Pietari Kalmin katu 5)
00014 Helsingin yliopisto

Sähkopostiosoite: info@cs.helsinki.fi
URL: http://www.cs.helsinki.fi/

HELSINGIN YLIOPISTO – HELSINGFORS UNIVERSITET – UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Koulutusohjelma — Utbildningsprogram — Study programme
Matemaattis-luonnontieteellinen tiedekunta	Tietojenkäsittelytieteen kandiohjelma

Tekijä — Författare — Author

Teemu Ruokokoski

Työn nimi — Arbetets titel — Title

Transformer-mallit ja niiden soveltaminen aikasarjojen analyysiin

Ohjaajat — Handledare — Supervisors

toht. P. Mikkola, apulaisprof. A. Klami

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Kanditutkielma	15.12.2024	24 sivua

Tiivistelmä — Referat — Abstract

Transformer-arkkitehtuuri on mullistanut syväoppimisen monilla sovellusalueilla, kuten luonnollisen kielen ja kuvien käsittelyssä. Sen vahvuus perinteisiin neuroverkkoratkaisuihin verrattuna perustuu tehokkaaseen huomiomekanismiin, joka mahdollistaa tärkeiden riippuvuuksien tunnistamisen syötteessä ilman tarvetta peräkkäiselle tiedonkäsittelylle. Transformer hyödyntää rinnakkaislaskentaa ja soveltuu erinomaisesti monimutkaisten ja pitkäkestoisten riippuvuuksien mallintamiseen, mikä antaa sille merkittävän edun muihin neuroverkkoratkaisuihin verrattuna. Nämä ominaisuudet tekevät Transformer-malleista kiinnostavan vaihtoehdon myös aikasarjojen analyysissä, jossa datan erityispiirteet, kuten trendit ja kausivaihtelut, asettavat omat vaatimuksensa.

Tutkielmassa tarkastellaan ensin Transformer-arkkitehtuurin perusrakennetta ja sen keskeisiä ominaisuuksia, kuten huomiomekanismia ja rinnakkaislaskennan mahdollistavia tekijöitä. Tämän jälkeen siirrytään aikasarjojen analyysiin, esittelemällä ensin aikasarjojen ominaispiirteet sekä perinteiset menetelmät, joita on käytetty aikasarjojen mallintamisessa. Lisäksi käsitellään erilaisia Transformer-pohjaisia ratkaisuja, joita on kehitetty aikasarjojen analysoimiseksi. Aikasarjat asettavat monia haasteita malleille ja arkkitehtuuria täytyy muokata näiden haasteiden ratkaisemiseksi. Vaikka Transformer-mallien potentiaali aikasarja-analyysissä on käynyt ilmi, niiden mahdollisuuksia ja rajoitteita ei vielä täysin ymmärretä. Transformer-mallien joustavuus ja muokattavuus tarjoavat kuitenkin vahvan pohjan jatkotutkimukselle.

ACM Computing Classification System (CCS)
General and reference → Document types → Surveys and overviews
Computing methodologies → Machine learning → Machine learning algorithms
Computing methodologies → Artificial intelligence → Planning and Scheduling → Planning under uncertainty

Avainsanat — Nyckelord — Keywords

transformers, neuroverkko, aikasarja

Säilytyspaikka — Förvaringsställe — Where deposited

Helsingin yliopiston kirjasto

Muita tietoja — övriga uppgifter — Additional information

Sisällys

1 Johdanto

Syväoppiminen (engl. *deep learning*) on koneoppimisen osa-alue, jossa monikerroksiset neuroverkot oppivat tunnistamaan ja mallintamaan monimutkaisia piirteitä suuresta määrästä dataa. Vaikka neuroverkkojen perusajatus on ollut olemassa jo vuosikymmeniä, vasta viimeaikainen tietokoneiden laskentatehon huomattava kasvu sekä suurten tietoaineistojen saatavuus ovat mahdollistaneet syväoppimisen läpimurron (Murphy, 2022). Tämä kehitys on johtanut siihen, että neuroverkkoja hyödynnetään laajalti eri sovellusalueilla, kuten luonnollisen kielen käsittelyssä, kuvan- ja puheentunnistuksessa, lääketieteellisessä diagnostiikassa ja itseajavien autojen ohjauksessa. Neuroverkkojen avulla on mahdollista ratkaista ongelmia, jotka olivat aiemmin liian haastavia tai aikaa vieviä.

Myös itse neuroverkkoarkkitehtuurit ovat kehittyneet huomattavasti viime vuosina. Perinteisten syväoppimismallien rinnalle on noussut uusia arkkitehtuureja, jotka ovat ratkaisseet monia aiempien mallien rajoitteita. Yksi merkittävimmistä läpimurroista on Googlen tutkijoiden vuonna 2017 esittelemä Transformer-arkkitehtuuri, joka julkaistiin artikkelissa "Attention Is All You Need" (Vaswani et al., 2017). Tämä arkkitehtuuri on muuttanut radikaalisti luonnollisen kielen käsittelyä (engl. *natural language processing*, NLP) ja se on muodostunut perustaksi monille nykyisille suurille kielimalleille (engl. *large language model*, LLM). Ennen Transformer-mallia luonnollisen kielen käsittelyssä käytettiin laajalti toistuvia neuroverkkoja[†] (engl. *recurrent neural network*, RNN), pitkiä lyhytkestomuisteja (engl. *long short-term memory*, LSTM) ja konvoluutioneuroverkkoja (engl. *convolutional neural network*, CNN) (Bishop ja Bishop, 2024). Näissä malleissa tietoa käsitellään askel kerrallaan, mikä hidastaa tietojen prosessointia. Tämä vaiheittainen lähestymistapa rajoittaa myös mallien kykyä käsitellä tilanteita, joissa nykyinen tieto (esimerkiksi sana tekstissä) on vahvasti sidoksissa aiempiin tietoihin.

Toisin kuin aikaisemmat neuroverkkomallit, Transformer-mallit käsittelevät tietoa rinnakkain. Tämä tekee niistä tehokkaita kielenkäsittelyssä ja kuvien analysoinnissa, joissa on saavutettu merkittäviä läpimurtoja. Mallin joustavuus on mahdollistanut sen soveltamisen myös monilla muilla aloilla. Viime aikoina Transformerin käyttöä on tutkittu myös aika-

[†]Suomenkielisenä käännöksenä käytetään toisinaan myös "takaisinkytkettyä neuroverkkoa", mutta tässä tutkielmassa käytetään "toistuvaa neuroverkkoa", koska esimerkiksi sähkötekniikan alalla vakiintunut takaisinkytkennän englanninkielinen vastinpari on "feedback".

sarjojen analysoinnissa, jossa sen kyky mallintaa pitkiä ja monimutkaisia riippuvuuksia tarjoaa uusia mahdollisuuksia. Transformer-mallia voidaan hyödyntää esimerkiksi talous- ja terveystietojen analyysissä, sääennusteiden laatimisessa sekä liikenteen ja sähkönkulutuksen ennakoimisessa.

Tutkielman tavoitteena oli syventää kirjoittajan ymmärrystä Transformer-arkkitehtuurista sekä selvittää, miten sitä voidaan soveltaa aikasarjojen analyysiin. Kirjallisuuskatsauksen laatimiseksi suoritettiin laaja hakuprosessi, jossa hyödynnettiin useita tieteellisiä tietokantoja, kuten ACM Digital Library, Google Scholar, IEEE Xplore ja Helka. Hakusanoina käytettiin monipuolista valikoimaa termejä ja niiden variaatioita, kuten "transformers", "attention", "time series analysis", "deep learning for time series", sekä muita aiheeseen liittyviä yhdistelmiä, joiden avulla löydettiin relevantteja artikkeleita ja tutkimuksia. Joitain tuoreita tutkimuksia jätettiin pois tutkielmasta, koska ne olivat vielä esijulkaisuvaiheessa.

Tutkielmassa tarkastellaan Transformer-mallin perusrakennetta, sen etuja verrattuna perinteisiin neuroverkkomalleihin, sekä mallin kykyä käsitellä pitkiä riippuvuuksia huomiomekanismin avulla. Pyrkimyksenä on selvittää, miten Transformer-arkkitehtuuria on muokattava vastaamaan aikasarjojen erityisvaatimuksia ja millaisia haasteita sen soveltaminen aikasarjoihin tuo mukanaan.

Tutkielma on jaettu viiteen lukuun. Johdannon jälkeen luvussa 2 esitellään Transformer-arkkitehtuuri, keskittyen sen keskeisiin piirteisiin ja etuihin perinteisiin neuroverkkoratkaisuihin verrattuna. Luvussa esitellään Transformer-mallin perusrakenne, joka koostuu kooderista ja dekooderista sekä mallin olennaisimmasta osasta eli huomiomekanismista.

Kolmannessa luvussa käsitellään aikasarjojen erityispiirteitä sekä perinteisiä menetelmiä ajallisen datan mallintamiseen. Tilastotieteellisistä malleista tarkastellaan ARIMA-mallia (Box ja Jenkins, 1970) sekä muita merkittäviä tilastollisia menetelmiä. Lisäksi esitellään lyhyesti koneoppimisen menetelmiä, joita on hyödynnetty aikasarja-analyysissä. Luvun lopussa syvennytään Transformer-mallin eri muunnelmiin ja niiden soveltamiseen aikasarjojen analyysissä.

Neljännessä luvussa tarkastellaan aikasarjojen analyysin keskeisiä haasteita, kuten datan muutoksista aiheutuvia ongelmia, laskennallisen monimutkaisuuden hallintaa sekä pitkien sekvenssien muistiongelmia. Lisäksi vertaillaan Transformer-mallien suorituskykyä muihin koneoppimisen ja tilastotieteen menetelmiin. Lopuksi pohditaan tutkimusmahdollisuuksia ja Transformer-mallien kehityssuuntia aikasarjojen ennustamisessa. Viidennessä luvussa esitetään yhteenveto tutkielman tärkeimmistä havainnoista.

2 Transformer-arkkitehtuuri

Alunperin luonnollisen kielen käsittelyyn kehitetty Transformer-arkkitehtuuri on mullistanut syväoppimisen kentän tarjoten tehokkaita ratkaisuja monen eri sovellusalueen haasteisiin. Tämä arkkitehtuuri on ollut suuri menestys edistyksellisissä luonnollisen kielen käsittelyn malleissa, kuten GPT, BERT ja T5. GPT (engl. *Generative Pre-trained Transformer*) käyttää Transformer-arkkitehtuuria tekstin generoimiseen (Radford, 2018) ja BERT (engl. *Bidirectional Encoder Representations from Transformers*) parantaa tekstin ymmärtämistä kaksisuuntaisen huomiomekanismin avulla (Devlin, 2018). T5 (engl. *Text-To-Text Transfer Transformer*) puolestaan yksinkertaistaa kielen käsittelyn tehtäviä muuntamalla ne tekstipohjaiseen muotoon (Raffel et al., 2020).

Kielimallien menestys on herättänyt laajaa kiinnostusta Transformer-arkkitehtuuria kohtaan ja sen soveltamiseen monilla eri aloilla. Viime aikoina Transformer-mallin käyttö on laajentunut esimerkiksi kuvantunnistukseen, äänenkäsittelyyn, videoanalyysiin, kuvien generointiin, biologisten sekvenssien analyysiin ja aikasarjojen ennustamiseen (Bishop ja Bishop, 2024). Nämä sovelluskohteet osoittavat Transformer-mallin joustavuuden ja soveltuvuuden moniin eri tietotyyppeihin.

2.1 Transformer-mallin edut

Toistuvat neuroverkot (RNN) ja pitkät lyhytkestomuistit (LSTM) käsittelevät syötteitä askel kerrallaan, mikä hidastaa laskentaa. Sarjamuotoinen tietojenkäsittely ei myöskään hyödynnä tehokkaasti moderneja grafiikkaprosessoreita (engl. *graphic processing unit*, GPU), joita käytetään neuroverkkojen laskentatehtävissä. Transformer-malli puolestaan käsittelee koko syötteen rinnakkain ja hyödyntää siten paremmin grafiikkaprosessorien tarjoamaa laskentatehoa (Bishop ja Bishop, 2024). Rinnakkaislaskenta tehostaa oppimista erityisesti suurten datamäärien käsittelyssä. Se parantaa myös mallin skaalautuvuutta, joka mahdollistaa erittäin suurten mallien käytön ja niiden suorituskyvyn entistä paremman optimoinnin.

Transformer-mallin vahvuus on sen huomiomekanismi (engl. *attention*, Bahdanau et al., 2015), joka mahdollistaa kaikkien syötteen osien samanaikaisen tarkastelun. Huomiomekanismin avulla malli arvioi eri tietoelementtien (esimerkiksi sanojen tai kuvien osien)

merkitystä suhteessa toisiinsa ja etsii samankaltaisuuksia syötteen eri osien välillä. Eri osia painotetaan niiden tärkeyden mukaan, minkä avulla tunnistetaan, mitkä osat syötteestä ovat olennaisimpia ymmärtämisen kannalta. RNN- ja CNN-malleilla on tunnetusti vaikeuksia käsitellä pitkiä riippuvuuksia, koska ne perustuvat peräkkäiseen tietojenkäsittelyyn ja paikalliseen tietoon. RNN-malleissa ongelmaksi muodostuu gradienttien katoaminen tai hallitsematon kasvu, joka ilmenee etenkin silloin, kun malli yrittää oppia yhteyksiä kaukana toisistaan olevien syötteen osien välillä (Murphy, 2022). CNN-mallit puolestaan keskittyvät pääasiassa lähiympäristön piirteiden tunnistamiseen, jolloin ne eivät välttämättä huomioi kauempana olevia yhteyksiä. Toisin kuin nämä perinteisemmät menetelmät, Transformer-malli ei tee oletuksia syötteen rakenteesta, vaan pystyy joustavasti käsittelemään pitkiäkin riippuvuuksia (Lin et al., 2022). Transformer-malli onkin syrjäyttänyt perinteiset RNN- ja CNN-mallit monissa sovelluksissa (Murphy, 2022).

Mallin yksi merkittävä etu on myös sen tehokkuus siirto-oppimisessa (engl. *transfer learning*). Malli voidaan aluksi kouluttaa suurilla datamäärillä itseohjautuvasti, jonka jälkeen malli voidaan hienosäätää valituille erityistehtäville ohjatun oppimisen avulla (Bishop ja Bishop, 2024). Siirtovaikutus on erityisen hyödyllistä tilanteissa, joissa käytettävissä oleva datamäärä on rajallinen. Tällöin malli voidaan opettaa ensin jollain muulla vain osittain vastaavalla tiedolla. Siirtovaikutuksen ansiosta malli voi silti saavuttaa korkean suorituskyvyn hyödyntämällä aiemmin esikoulutuksessa oppimiaan tietoja.

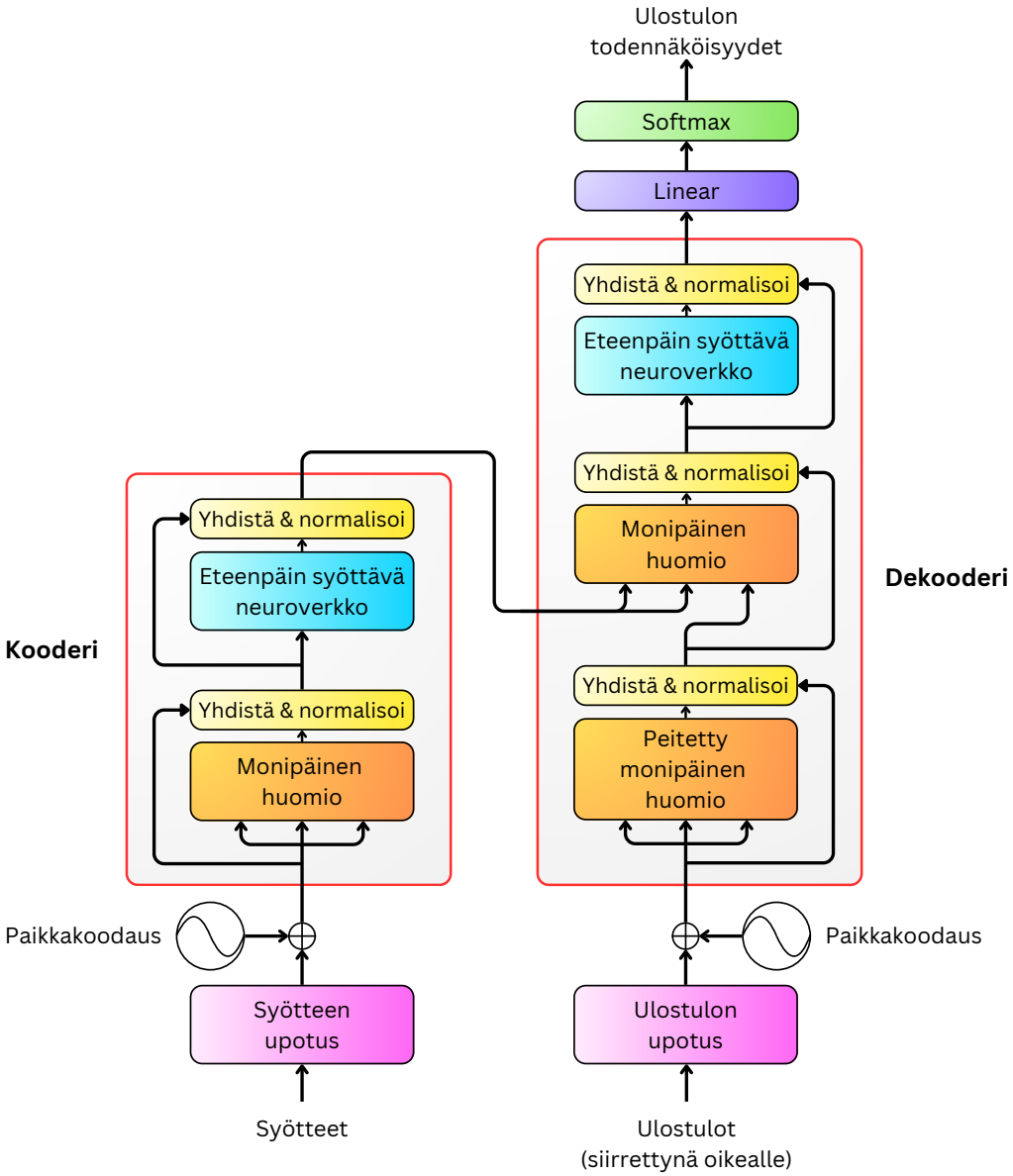
Transformer-malli on osoittautunut monipuoliseksi eri sovellusalueilla ilman suuria rakenteellisia muutoksia. Huomiomekanismi mahdollistaa erilaisten sekvenssityyppien käsittelyn, koska syötteet voivat olla epäsäännöllisiä vektorijoukkoja, järjestettyjä sekvenssejä tai muita yleisempiä esitysmuotoja (Bishop ja Bishop, 2024). Tämä tekee mallista soveltuvan laajaan valikoimaan tehtäviä.

2.2 Perusrakenne ja toiminta

Alkuperäinen Transformer-malli (Vaswani et al., 2017) kehitettiin konekäännöksiä varten. Malli koostuu kahdesta pääkomponentista: kooderista (engl. *encoder*), joka käsittelee syötteen ja dekooderista (engl. *decoder*), joka tuottaa käännöksen. Kooderi-dekooderi-rakenne (Cho et al., 2014) oli yleistynyt vain muutama vuosi ennen Transformerin esittelyä.

Kuvassa 2.1 esitetään alkuperäisen Transformer-arkkitehtuurin rakenne. Kielen kääntämisessä merkkijonot pilkotaan ensin pienempiin yksiköihin, kuten sanoihin, alisanoihin tai merkkeihin (tokenisointi). Nämä tokenit muutetaan vektorimuotoisiksi esityksiksi, joita

kutsutaan sanaupotuksiksi (engl. *word embeddings*). Sanaupotuksiin lisätään paikkakoodaus (engl. *positional encoding*), minkä jälkeen ne syötetään kooderiin, joka tuottaa tokeneille kontekstuaaliset esitykset. Toisin sanoen nämä esitykset sisältävät tietoa sanojen merkityksistä ja suhteista toisiinsa huomioiden samalla kontekstin, jossa sanat esiintyvät. Dekooderi puolestaan käyttää näitä esityksiä sekä aiemmin tuotettuja sanoja tuottaakseen ulostulon yksi sana kerrallaan. Malli laskee kullekin sanalle todennäköisyydet koko sanaston osalta ja valitsee seuraavan sanan todennäköisimpien vaihtoehtojen joukosta.



Kuva 2.1: Alkuperäinen Transformer-arkkitehtuuri, joka sisältää kooderin ja dekooderin (muokattu lähteestä Vaswani et al., 2017).

Eri sovelluksissa voidaan hyödyntää erilaisia arkkitehtuurin kokoonpanoja, riippuen siitä millaisesta tehtävästä on kyse. Alkuperäinen kooderi-dekooderi-malli soveltuu etenkin sekvenssistä sekvenssiin (engl. *sequence-to-sequence*) tehtäviin, kuten kielen kääntämiseen (Vaswani et al., 2017). Pelkkä kooderi on puolestaan tehokas sekvenssin ymmärtämistehtävissä, kuten luokittelussa ja tunnistuksessa. Esimerkkinä tästä on BERT-malli (Devlin, 2018), jota käytetään erityisesti kysymysten ymmärtämisessä ja tekstin luokittelussa. Vastaavasti pelkkä dekooderi soveltuu sekvenssien luontiin, kuten kielen mallintamiseen. Esimerkiksi hyvin suosittuja GPT-malleja käytetään tekstin luomisessa ja ne toimivat pelkästään dekoodereina (Radford, 2018).

2.2.1 Kooderi ja dekooderi

Transformer-arkkitehtuurissa kooderi ja dekooderi koostuvat useista identtisistä kerroksista, joita voidaan pinota. Alkuperäisessä mallissa sekä kooderilla että dekooderilla oli kuusi kerrosta, mutta käytännössä kerroksia voidaan lisätä tarpeen mukaan. Jokaisessa kerroksessa hyödynnetään monipäistä itsehuomiota (engl. *multi-head self-attention*) ja eteenpäin syöttävää neuroverkkoa (engl. *feedforward neural network*, FFNN).

Kooderi vastaa syötteen käsittelystä ja sen muuntamisesta. Vaikka kooderikerrokset ovat muuten identtisiä, jokaisella kerroksella on omat opitut parametrit. Kunkin kooderikerroksen rakenne sisältää itsehuomiomekanismin, eteenpäin syöttävän neuroverkon, jäännösyhteyden (engl. *residual connection*) ja kerroksen normalisoinnin (engl. *layer normalization*).

Jäännösyhteys luo oikotien eri kerrosten välille. Yhteyden avulla syötteen alkuperäiset ominaisuudet voidaan yhdistää kunkin kerroksen tuottamaan uuteen tietoon. Tämä vähentää oppimisprosessiin liittyvää gradienttien katoamista ja helpottaa syvien verkkojen koulutusta (He et al., 2016). Kerrosnormalisointi puolestaan parantaa oppimisprosessin tehokkuutta ja lisää mallin vakautta tasapainottamalla kerroksien aktivaatioita (Ba, 2016).

Ennen kuin syöte siirtyy kooderin käsittelyyn, se käy läpi esikäsittelyvaiheet, jotka sisältävät sanaupotukset ja paikkakoodauksen. Sanaupotuksessa syötteen osat muutetaan matemaattiseen muotoon vektoreiksi, joissa samankaltaiset syötteen osat sijaitsevat lähellä toisiaan vektoreiden moniulotteisessa esityksessä. Esimerkiksi sanat "metsä" ja "puu" saattavat olla toisiaan muistuttavia vektoreita, kun taas täysin erilaisten sanojen, kuten "koira" ja "tietokone", vektorietäisyys on suurempi. Sanaupotusten perusidea pätee myös muunlaiseen dataan, kuten kuviin ja ääniin. Myös niissä syötteen osat muutetaan vektoreiksi, joiden piirteet mahdollistavat eri osien vertailun. Paikkakoodaus puolestaan lisää

syötteeseen tietoa osien sijainnista, jolloin malli pystyy tunnistamaan ja hyödyntämään osien suhteellisia sijainteja.

Kooderi luo syötteestä vektoreita, jotka esittävät merkityksellisiä riippuvuuksia syötteen eri osien välillä. Kooderikerroksissa itsehuomiomekanismi tunnistaa ja painottaa tärkeitä suhteita syötteen sisällä, jolloin malli pystyy ymmärtämään sekä lähekkäin että kaukana sijaitsevien osien välisiä yhteyksiä.

Dekooderin päätehtävä Transformer-mallissa on tuottaa sarjamuotoista dataa, mikä tekee siitä sopivan sekvenssien generointitehtäviin, kuten kielenkäännökseen ja tekstin tuottamiseen. Se hyödyntää kooderilta saatuja tietoja ja aiemmin generoitua sisältöä luodakseen seuraavan osan peräkkäisestä jaksosta vaihe vaiheelta.

Dekooderin itsehuomiokerros eroaa hieman kooderin vastaavasta. Dekooderi ei saa nähdä tulevia osia generoitavasta jaksosta, mikä estetään käyttämällä peitettyä itsehuomiomekanismia (engl. *masked self-attention*). Tämä tarkoittaa, että dekooderi voi tarkastella vain aiemmin luotuja osia nykyisen osan tuottamiseen. Jos dekooderi saisi myös ennustettavat osat syötteenä, se voisi huijata oppimisprosessin aikana hyödyntämällä tulevia tietoja (Prince, 2023). Tällöin mallin kyky oppia ennustamaan seuraavia osia heikkenisi merkittävästi.

Dekooderi hyödyntää peitetyn itsehuomiomekanismin lisäksi erillistä huomiomekanismia, joka yhdistää dekooderin tuottaman tiedon kooderin muokkaamaan syötteeseen. Näin dekooderi pystyy tuottamaan johdonmukaista sisältöä. Dekooderin viimeiset kerrokset koostuvat lineaarisesta kerroksesta ja sitä seuraavasta softmax-funktiosta, joka laskee ennustettavien vaihtoehtojen todennäköisyydet (Murphy, 2022). Dekooderi liittää tuottamansa sisällön jatkuvasti kasvavaan syötteeseensä, kunnes malli ennustaa loppumerkin, jolloin prosessi päättyy.

2.2.2 Huomiomekanismi

Huomiomekanismi on Transformer-arkkitehtuurin toiminnan perusta. Alun perin huomiomekanismi esiteltiin RNN-mallien laajennuksena parantamaan niiden suorituskykyä kielenkäännöksessä (Bahdanau et al., 2015), mutta sen todellinen läpimurto tapahtui vasta Transformer-malleissa. Toisin kuin perinteiset neuroverkot, joissa syötteitä kerrotaan kiinteillä painoilla, Transformer käyttää syötteestä riippuvia painokertoimia, jotka mukautuvat dynaamisesti syötteen mukaan (Bishop ja Bishop, 2024).

Huomiomekanismissa kaikki syötevektorit järjestetään riveittäin syötematriisiin $\mathbf{X}$, jonka

muoto on $n \times d$, missä n on syötteen vektorien määrä (esimerkiksi sanojen lukumäärä tekstissä) ja d on kunkin vektorin dimensio (piirteiden määrä). Matriisi $\mathbf{X}$ muunnetaan kolmeksi erilliseksi matriisiksi: kyselyiksi $\mathbf{Q}_{n\times d_q}$, avaimiksi $\mathbf{K}_{n\times d_k}$ ja arvoiksi $\mathbf{V}_{n\times d_v}$. Tämä muunnos tapahtuu kertomalla syötematriisi $\mathbf{X}$ kolmella eri painomatriisilla $\mathbf{W}^{(q)}_{d\times d_q}$, $\mathbf{W}^{(k)}_{d\times d_k}$ ja $\mathbf{W}^{(v)}_{d\times d_v}$, jotka ovat mallin koulutuksen aikana oppimia parametreja. On tärkeää huomata, että matriisien dimensioiden tulee olla yhteensopivia, jotta kertolaskut voidaan suorittaa. Yhtälöinä nämä voidaan esittää seuraavasti (Bishop ja Bishop, 2024):

$$\mathbf{Q} = \mathbf{X}\mathbf{W}^{(q)} \tag{2.1}$$

$$\mathbf{K} = \mathbf{X}\mathbf{W}^{(k)} \tag{2.2}$$

$$\mathbf{V} = \mathbf{X}\mathbf{W}^{(v)} \tag{2.3}$$

Näiden laskutoimitusten tuloksena muodostuvat matriisit $\mathbf{Q}$, $\mathbf{K}$ ja $\mathbf{V}$, jotka sisältävät jokaiselle syötteen osalle vastaavat kysely-, avain- ja arvovektorit. Kyselyt ($\mathbf{Q}$) määrittävät, mitä tietoa malli kulloinkin etsii, kun taas avaimet ($\mathbf{K}$) koodaavat syötteen osien keskeiset ominaisuudet, joita kyselyt käyttävät vertailukohtana. Arvot ($\mathbf{V}$) puolestaan sisältävät sen tiedon, joka lopulta yhdistyy mallin lopulliseen tulokseen (Murphy, 2022). Tämä rakenne mahdollistaa sen, että malli pystyy joustavasti painottamaan eri syöteosien merkitystä.

Syöteosien välisten riippuvuuksien laskeminen tapahtuu kysely- ja avainvektorien pistetulon avulla, joka mittaa vektorien samankaltaisuutta. Pistetulon suuruus ilmaisee, kuinka vahva yhteys kahden syöteosan välillä on. Jokaisen syöteosan kyselyvektori kerrotaan kaikkien muiden syöteosien avainvektoreiden kanssa, jolloin saadaan huomioarvot, jotka kuvaavat kunkin syöteosan merkitystä suhteessa muihin syöteosiin.

Jotta pistetulojen arvot pysyisivät hallinnassa erityisesti silloin, kun vektoreiden ulottuvuus d on suuri, ne skaalataan jakamalla vektoreiden ulottuvuuden neliöjuurella. Ilman skaalausta pistetulojen arvot voivat kasvaa niin suuriksi, että mallin herkkyys kärsii ja mallista tulee vaikea kouluttaa (Prince, 2023). Skaalauksen jälkeen pistetuloille suoritetaan normalisointi softmax-funktion avulla, joka suhteuttaa ne välille $0 - 1$. Nämä huomioarvot ilmaisevat, kuinka paljon painoarvoa kullekin syöteosalle annetaan.

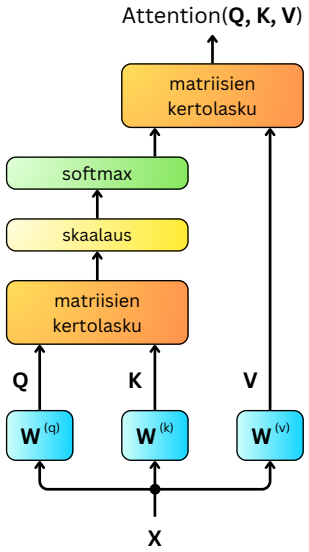
Lopuksi huomioarvot kerrotaan vielä arvovektoreilla, jolloin jokaisen syöteosan merkitys painotetaan sen saamien huomioarvojen perusteella. Lopputuloksena saadaan uusi matriisi, joka huomioi syöteosien väliset riippuvuudet. Tämä mahdollistaa sen, että malli pystyy keskittymään olennaisiin osiin riippumatta siitä, ovatko ne lähekkäin vai kaukana toisistaan.

Skaalatun pistetulohuomion laskeminen voidaan esittää yhtälöllä (Vaswani et al., 2017):

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V} \tag{2.4}$$

Koska syötevektorit kootaan matriisimuotoon, huomion laskeminen voidaan suorittaa rinnakkain. Laskennassa voidaan hyödyntää grafiikkaprosessoreita ja kaikki syöteosat voidaan käsitellä samanaikaisesti. Tämä parantaa merkittävästi laskennan nopeutta ja mahdollistaa suurempien datamäärien käsittelyn.

Kuva 2.2 havainnollistaa skaalatun pistetulohuomion laskentaa. Tämä rakenne muodostaa yksittäisen huomiopään monipäisessä huomiomekanismissa.



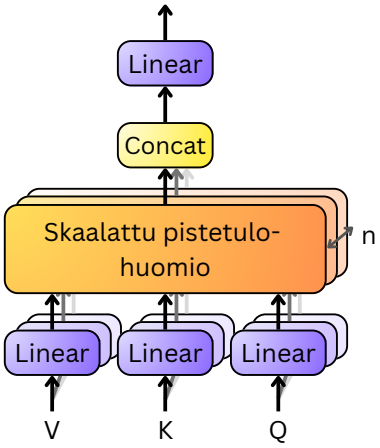
Kuva 2.2: Skaalatun pistetulohuomion laskeminen (muokattu lähteestä Bishop ja Bishop, 2024).

2.2.3 Monipäinen huomio

Transformer-mallit käyttävät usein monipäistä huomiomekanismia, vaikka periaatteessa yksi huomiokerros voisi riittää. Jos käytössä on vain yksi huomiokerros, se voi tunnistaa tietyn riippuvuuden syötteen osien välillä, mutta saattaa sivuuttaa muita tärkeitä suhteita.

Monipäisessä huomiossa huomiomekanismeja lisätään rinnakkain ja se mahdollistaa useiden eri näkökulmien samanaikaisen tutkimisen. Monipäisen huomiomekanismin rakenne on esitetty kuvassa 2.3. Koska eri huomiopäät (engl. *attention heads*) käyttävät omia painomatriisejaan **Q**-, **K**- ja **V**-matriisien laskemiseen, jokainen pää pystyy käsittelemään erityyppisiä riippuvuuksia syötteen osien välillä. Lopullinen huomioarvo muodostetaan

yhdistämällä kaikkien päiden antamat tulokset. Näin malli pystyy ymmärtämään syvällisemmin syötettä ja kykenee monimutkaisempiin tehtäviin.



Kuva 2.3: Monipäinen huomio koostuu useasta rinnakkaisesta huomiokerroksesta (muokattu lähteestä Vaswani et al., 2017).

2.2.4 Eteenpäin syöttävä neuroverkko

Monipäisen huomiomekanismin jälkeen Transformer-arkkitehtuurissa on tavallinen eteenpäin syöttävä neuroverkko (FFNN). Verkko käsittelee jokaisen syötevektorin lisäten syötteeseen epälineaarisia muunnoksia (Lin et al., 2022). Huomiomekanismin tuottamaa informaatiota siis jatkojalostetaan ennen sen siirtämistä seuraavaan kerrokseen. Tämä prosessi parantaa mallin kykyä oppia monimutkaisempia riippuvuuksia syötteiden välillä (Bishop ja Bishop, 2024).

Eteenpäin syöttävä neuroverkko koostuu tyypillisesti kahdesta täysin liitetystä kerroksesta sekä epälineaarisesta aktivointifunktiosta, kuten ReLU (engl. *rectified linear unit*, Glorot et al., 2011). Vaikka FFNN on rakenteeltaan varsin yksinkertainen, se on olennainen osa Transformer-mallin suorituskyvyn kannalta.

2.2.5 Paikkakoodaus

Syötteen osien järjestys on oleellinen tieto useimmissa peräkkäisiä käsittelytehtäviä vaativissa sovelluksissa. Esimerkiksi kielimalleissa lauseen merkitys voi muuttua täysin toiseksi rippuen siitä, missä järjestyksessä sanat esiintyvät. Transformer-mallissa ei kuitenkaan itsessään ole sisäänrakennettua ymmärrystä järjestyksestä. Järjestyksen ja suhteellisten etäisyyksien mallintaminen on ratkaistava lisäämällä erillinen paikkakoodaus syötteeseen

(Vaswani et al., 2017). Ilman tätä mekanismia malli ei voisi tunnistaa järjestystä tai suhteita syötteen eri osien välillä, mikä tarkoittaisi, että se tuottaisi saman tuloksen, vaikka osien järjestys vaihtuisi.

Alkuperäisessä artikkelissa (Vaswani et al., 2017) tutkittiin myös opittuja paikkakoodauksia (engl. *learned positional encodings*), mutta päädyttiin käyttämään sin- ja cos-funktioita. Tällöin jokaiselle syötteen elementille lasketaan paikkakoodaus seuraavasti:

$$\begin{aligned} PE_{(pos,2i)} &= \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right) \\ PE_{(pos,2i+1)} &= \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right) \end{aligned} \tag{2.5}$$

missä

- $PE_{(pos,2i)}$ on paikkakoodaus parillisille indekseille,
- $PE_{(pos,2i+1)}$ on paikkakoodaus parittomille indekseille,
- pos on syötteen elementin sijainti,
- i on dimensioindeksi,
- d_{model} on mallin dimensio.

Yhtälössä lukuarvoa 10000 voidaan tarvittaessa muuttaa syötteen pituudesta riippuen.

Paikkakoodaus perustuu sin- ja cos-funktioiden sarjaan, joiden aallonpituudet kasvavat tasaisesti. Tämä mahdollistaa sen, että paikkakoodaus voidaan laskea minkä tahansa pituisille syötteille. Lisäksi jokaisen sijainnin esitys on lineaarisesti ennustettavissa suhteessa muihin sijainteihin, mikä helpottaa mallin kykyä ymmärtää syötteen elementtien välisiä suhteita ja etäisyyksiä (Murphy, 2022).

Kun paikkakoodaus on laskettu, se yhdistetään alkuperäiseen syötteeseen, jolloin malli saa tarvitsemansa tiedon syöteosien suhteiden tulkitsemiseksi. Vaikka sin- ja cos-funktiot toimivat hyvin kielimalleissa, on kehitetty myös monia muita paikkakoodausmenetelmiä, jotka tarjoavat erilaisia etuja eri käyttötarkoituksiin. Näihin menetelmiin kuuluu esimerkiksi opittujen paikkakoodauksien käyttö, missä paikkatiedot opitaan mallin muiden parametrien kanssa. Lisäksi syötematriiseja voidaan manipuloida tai syötettä esikäsitellä toistuvan neuroverkon (RNN) avulla (Dufter et al., 2022), mikä voi parantaa mallin suorituskykyä.

3 Aikasarjojen analysointi

Aikasarjat ovat ajallisesti järjestettyä dataa, joka sisältää usein kausivaihteluita, trendejä ja pitkän aikavälin riippuvuuksia. Niiden analysoinnin tavoitteena on ymmärtää menneisyyttä ja ennustaa tulevaa, mikä mahdollistaa organisaatioille ja päätöksentekijöille parempien suunnitelmien ja päätösten tekemisen. Aikasarja-analyysia hyödynnetään monissa päivittäisissä operatiivisissa päätöksissä, kuten varastonhallinnassa, sääennusteiden laatimisessa, liikenteen sujuvuuden parantamisessa, sähkönkulutuksen optimoinnissa sekä talouden muutosten ennakoinnissa.

Monilla aloilla tarkat ennusteet ovat taloudellisesti arvokkaita ja voivat vaikuttaa suoraan liiketoiminnan kannattavuuteen. Esimerkiksi osakemarkkinoiden hintojen ennustaminen perustuu aikasarjojen analysointiin. Osakekurssit vaihtelevat jatkuvasti ja niiden arvoon vaikuttavat lukuisat tekijät, kuten talouden yleinen tila, geopoliittiset tapahtumat ja yritysten tulosraportit. Osakkeiden hinnoittelussa on mahdollista havaita toistuvia kaavoja ja sen vuoksi monet yritykset ovat investoineet valtavia summia näitä kaavoja tunnistavien koneoppimisjärjestelmien kehittämiseen (Huyen, 2022).

Perinteisiä tilastotieteeseen perustuvia menetelmiä, kuten liikkuvia keskiarvoja ja autoregressiivisia malleja (Box ja Jenkins, 1970), on käytetty laajasti aikasarjojen analyysissä. Ne soveltuvat yksinkertaisiin tilanteisiin, mutta monimutkaisten riippuvuuksien hallinta voi olla niille haastavaa. Neuroverkkojen kehittyminen on tuonut uusia mahdollisuuksia vaativampien aikasarjojen analysointiin ja ennustamiseen.

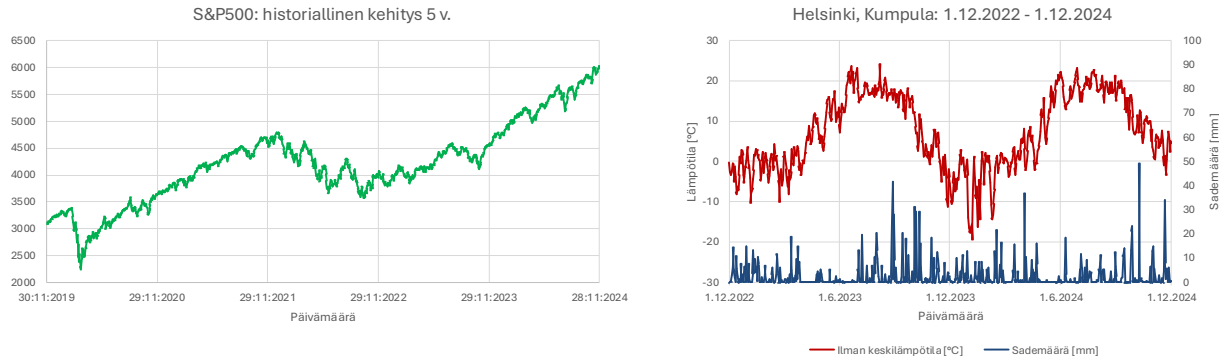
3.1 Aikasarjojen erityispiirteet

Aikasarjat ovat havaintoja ilmiöistä, joita mitataan ja tallennetaan tietyin aikavälein. Toisin kuin muissa sekventiaalisissa datatyypeissä, kuten esimerkiksi kielessä tai geenisekvensseissä, havaintojen ajankohdat ovat olennainen tekijä aikasarjojen analyysissä. Tällaisia tietoja syntyy luonnollisesti monissa arkielämän tilanteissa, kuten osakkeiden hintakehityksessä, säähavainnoissa ja sähkönkulutuksen seurannassa. Aikasarjojen erityinen piirre on se, että peräkkäiset havainnot ovat usein riippuvaisia toisistaan (Box et al., 2015). Aikasarjadatan tyyppi ja ominaisuudet vaihtelevat merkittävästi sovelluksen mukaan ja sen käsittely eroaa monin tavoin luonnollisen kielen käsittelystä. Yksittäisellä havaintopisteel-

lä ei ole samanlaista semanttista merkitystä kuin sanalla lauseessa.

Aikasarjojen kausivaihtelut, syklit ja trendit vaikuttavat niiden analysointiin (Box et al., 2015). Kausivaihtelut tarkoittavat säännöllisiä muutoksia, jotka toistuvat tietyn aikavälin, kuten vuodenaikojen tai kuukausien, mukaan. Esimerkiksi vähittäiskaupan myynti kasvaa tyypillisesti ennen joulua, kun taas kesäkuukausina se saattaa vähentyä. Syklit puolestaan kuvaavat pidempiä, ennustamattomampia vaihteluita, jotka voivat kestää useita vuosia, kuten talouden nousu- ja laskukaudet. Toisin kuin kausivaihtelut, syklit eivät noudata tarkkoja aikarajoja, joten niiden ennustaminen voi olla haastavaa. Trendi puolestaan tarkoittaa pitkäaikaista ja samansuuntaista muutosta datassa, kuten jatkuvaa kasvua tai laskua. Esimerkiksi viime vuosina kasvispohjaisten ruokien suosio on ollut kasvussa, kun yhä useammat ihmiset ovat siirtyneet käyttämään enemmän kasvipohjaisia tuotteita ympäristö- ja terveystietoisuuden lisääntyessä.

Kuvassa 3.1 on esitetty kaksi esimerkkikuvaajaa aikasarjoista. Vasemmalla on S&P500-osakeindeksin historiallinen kehitys viiden vuoden ajalta (Nasdaq, 2024). Alun jyrkkä pudotus johtuu COVID-19-pandemian aiheuttamasta paniikista, mutta sen jälkeen trendi kääntyi nopeasti nousuun. Vuonna 2022 USA:n osakemarkkinoilla nähtiin laskutrendi, mutta vuoden lopulla trendi kääntyi jälleen nousuun. Oikeassa kuvassa esitetään päivittäinen ilman keskilämpötila ja sademäärä kahden vuoden ajalta, mitattuna Helsingin Kumpulassa (Ilmatieteen laitos, 2024). Lämpötilakäyrästä erottuu selkeä kausivaihtelu, joka on tyypillistä Suomen ilmastolle.



Kuva 3.1: Vasemmalla S&P500-osakeindeksin historiallinen kehitys viiden vuoden ajalta (Nasdaq, 2024). Oikealla päivittäinen ilman keskilämpötila ja sademäärä kahden vuoden ajalta, mitattuna Kumpulassa (Ilmatieteen laitos, 2024).

Useimmat aikasarjat ovat stokastisia (Box et al., 2015), mikä tarkoittaa, että niiden kehitys ei ole täysin ennustettavissa aiempien havaintojen perusteella. Vaikka aikaisemmilla arvoilla on merkitystä tulevien havaintojen kannalta, myös satunnaiset tekijät vaikuttavat

tuloksiin. Monet aikasarjat saattavat näyttää trendejä, jotka voivat olla osa pidempiaikaista sykliä tai täysin satunnaisia ilmiöitä (Cowpertwait, 2009). Tällaiset stokastiset trendit ovat yleisiä talouden aikasarjoissa. Aikasarjojen epäsäännöllisyys ja jatkuva kehitys edellyttävät ennustemalleilta jatkuvaa sopeutumista, sillä kulttuurit, muodit ja teknologiat muuttuvat jatkuvasti. Ennustemalli on hyödyllinen vain jos sen oppimat menneisyyden kaavat toistuvat myös tulevassa datassa (Huyen, 2022).

3.2 Analyysimenetelmät

Aikasarja-analyysin historia ulottuu 1920- ja 1930-luvuille, jolloin kehitettiin ensimmäisiä matemaattisia malleja aikasarjojen ennustamiseen. Näihin kuuluvat etenkin autoregressiiviset (AR) ja liukuvan keskiarvon (MA) mallit (Box et al., 2015). Autoregressiivisissa (AR) malleissa prosessin nykyinen arvo ennustetaan edellisten arvojen lineaarisena yhdistelmänä, johon lisätään satunnaistermi, joka kuvaa prosessissa tapahtuvaa satunnaista vaihtelua. Liukuvan keskiarvon (MA) malli arvioi aikasarjan nykyistä arvoa aikaisempien havaintojen virheiden keskiarvon avulla, mikä tasoittaa satunnaista vaihtelua.

ARMA- ja ARIMA-mallit, jotka yhdistävät autoregressiiviset ja liukuvan keskiarvon mallit, vakiintuivat 1970-luvulla Boxin ja Jenkinsin kehitystyön myötä (Box ja Jenkins, 1970). ARIMA-malli kykenee huomioimaan myös trendit ja kausivaihtelut, mikä tekee siitä käyttökelpoisen monimutkaisissa aikasarjoissa. Näitä malleja käytetään edelleen aikasarjojen analysoimiseen, mutta niiden kyky käsitellä epälineaarista dataa on rajallinen. Mallin parametrien säätö edellyttää asiantuntemusta ja manuaalista optimointia, mikä rajoittaa niiden käyttöä suurissa ja monimuotoisissa tietoaineistoissa. ARIMA-mallien kehitys on kuitenkin luonut perustan monille nykyisille tilastollisille ja koneoppimispohjaisille aikasarjamalleille.

Lisäksi tilastotieteessä on käytössä monia muita aikasarjamalleja. Esimerkiksi ETS-malli (engl. *error, trend, seasonal*) soveltuu hyvin kausivaihteluita ja trendejä sisältävien aikasarjojen ennustamiseen, sillä se painottaa tuoreimpia havaintoja enemmän kuin vanhempia (Hyndman, 2018). VAR-malli (engl. *vector autoregression*) mahdollistaa useiden muuttujien samanaikaisen analysoinnin, jolloin jokaisen muuttujan arvo riippuu sekä sen omista aikaisemmista arvoista että muiden muuttujien aikaisemmista arvoista (Box et al., 2015). Facebookin (nykyisin Meta) tutkijaryhmän kehittämä Prophet-malli puolestaan on kehitetty aikasarjojen ennustamiseen, joissa on voimakkaita kausivaihteluita, trendin muutoksia ja puuttuvia havaintoja (Taylor ja Letham, 2018). Mallin helppokäyttöisyys

mahdollistaa ennustamisen ilman syvällistä tilastotieteellistä asiantuntemusta.

Aikasarjojen analysoinnissa on siirrytty yhä useammin koneoppimismallien käyttöön, koska reaalimaailman aikasarjojen vaihtelut ovat usein liian monimutkaisia perinteisten menetelmien käsiteltäväksi. Näitä koneoppimismalleja ovat esimerkiksi tukivektorikoneet (engl. *support vector machine*, SVM), päätöspuut (engl. *decision tree*), satunnaismetsät (engl. *random forest*) ja gradient boosting-menetelmät, kuten XGBoost (Nielsen, 2019). Nämä mallit hyödyntävät dataan perustuvaa oppimista, mikä mahdollistaa monimutkaisempien suhteiden ja epälineaaristen ilmiöiden havaitsemisen. Etenkin XGBoost on tunnettu tehokkuudestaan ja suorituskyvystään. Se on saavuttanut suosiota useissa koneoppimisen kilpailuissa ja käytännön sovelluksissa.

Neuroverkot tarjoavat joustavan menetelmän, joka mahdollistaa aikasarjadatan mallintamisen ilman ennakko-oletuksia datan rakenteesta. Lupaavia tuloksia on saavutettu esimerkiksi RNN- (Rangapuram et al., 2018), LSTM- (Siami-Namini et al., 2018) ja CNN-malleilla (Wu et al., 2022). Lisäksi eteenpäin syöttäviin verkkoihin perustuvat mallit, kuten N-BEATS ja NHITS (Challu et al., 2023), ovat osoittautuneet tehokkaiksi erilaisissa aikasarjaennusteissa.

Suurten kielimallien menestys tekee Transformer-mallista houkuttelevan vaihtoehdon myös aikasarja-analyysiin, sillä se tarjoaa tehokkaan tavan mallintaa pitkiä riippuvuuksia ja monimutkaisia ajallisia suhteita. Seuraavassa luvussa tarkastellaan tarkemmin, miten Transformer-arkkitehtuuria on muokattu ja sovellettu aikasarjojen analyysiin.

3.3 Transformer-variantit aikasarjojen analyysissä

Vaikka alkuperäinen Transformer-arkkitehtuuri soveltuu myös aikasarjojen analyysiin, sen suorituskyky ei ole optimaalinen aikasarjadatan käsittelyssä. Aikasarjojen käsittelyssä on tärkeää huomioida paikalliset piirteet, kuten trendit, kausivaihtelut ja muutospisteet, jotka riippuvat ympäröivästä kontekstista. Alkuperäinen Transformer ei kuitenkaan kykene huomioimaan näitä paikallisia piirteitä riittävän hyvin (Li et al., 2019).

Transformer-mallista on kehitetty lukuisia erilaisia muunnelmia, jotka pyrkivät parantamaan mallin suorituskykyä aikasarjojen analyysissä. Viime vuosina tutkimus Transformerien soveltamisesta aikasarjoihin on ollut runsasta. Seuraavassa esitellään muutamia muunnelmia ja niiden erikoisuuksia.

Varhaisena ratkaisuna LogSparse (Li et al., 2019) hyödynsi logaritmista itsehuomiota pit-

kän aikavälin riippuvuuksien mallintamiseen, mikä vähensi laskennallista monimutkaisuutta ja muistin tarvetta. Logaritmisessa itsehuomiossa kukin havaintopiste huomioi kaikki lähellä olevat pisteet, mutta kauempana olevat pisteet huomioidaan eksponentiaalisin askelin. Lisäksi LogSparse käytti konvoluutiokerroksia itsehuomiossa, mikä parantaa paikallisten piirteiden tunnistamista ja auttaa trendien sekä poikkeamien havaitsemisessa.

Informer (H. Zhou et al., 2021) puolestaan hyödyntää niin kutsuttua ProbSparse- itsehuomiota. Tämän avulla malli valitsee vain syötteen merkittävimmät kyselyvektorit, mikä vähentää laskennallista kuormitusta. Autoformer (Wu et al., 2021) puolestaan käyttää rakennetta, joka jakaa aikasarjadatan kausivaihtelu- ja trendikomponentteihin. Autokorrelaatiomekanismi toimii huomiomoduulina, jolloin malli voi hyödyntää aiempien havaintojen toistuvia rakenteita ennusteissaan. Myös FEDformer-malli (T. Zhou et al., 2022) soveltaa kausivaihtelu- ja trendikomponenttien erottelua. Lisäksi malli hyödyntää Fourier- ja wavelet-muunnoksia ja käyttää itsehuomiomekanismia taajuustasossa.

PatchTST (Nie et al., 2023) jakaa aikasarjadatan kiinteän kokoisiin aikajaksoihin tai lohkoihin. Tämä tarkoittaa jakamista tietyn aikavälin osiin, kuten esimerkiksi 50- tai 100-aikapisteen pätkiin. Kukin lohko (*patch*) sisältää useita peräkkäisiä aikapisteitä ja nämä lohkot käsitellään erillisinä osina. Lohkoihin jakaminen mahdollistaa pidemmän historiallisen syötteen hyödyntämisen ja samalla paikallisten merkityksellisten tietojen säilyttämisen. Tämän lähestymistavan myötä myös laskenta- ja muistivaatimukset vähenevät.

Viime aikoina on myös tutkittu suurten kielimallien käyttöä pohjana aikasarjojen analysointiin. Esimerkkeinä tästä ovat GPT4TS (Zhou et al., 2023) ja TimeLLM (Jin et al., 2023). Näiden mallien ideana on hyödyntää kielimallien laajaa koulutusaineistoa siirtovaikutuksen kautta: valmiiksi koulutettua kielimallia käytetään pohjana ja sen ulostulokerrokset, paikkakoodaukset sekä upotukset koulutetaan uudelleen aikasarjadatan tarpeisiin.

Tuoreissa julkaisuissa on tutkittu myös perusmallien (engl. *foundation model*) kehitystä aikasarjoja varten. Perusmallit ovat malleja, jotka on esikoulutettu suurilla aineistoilla ja joita voidaan myöhemmin hienosäätää erityisiin tehtäviin. Ensimmäinen tällainen malli on TimeGPT (Garza et al., 2023), joka on esikoulutettu hyödyntämällä laajaa aikasarjadataa eri lähteistä, kuten taloudesta, säästä, energiasta ja terveydenhuollosta. Kun malli opetetaan monipuolisella datalla, joka sisältää kohinaa, kausivaihteluja, eripituisia syklejä sekä erilaisia trendejä ja poikkeamia, se parantaa mallin kykyä soveltaa opittua eri tilanteisiin.

4 Aikasarjojen käsittelyn haasteet ja ratkaisut

Aikasarjoissa satunnaiset vaihtelut sekoittuvat ja yhdistyvät mahdollisesti useisiin eripituisiin sykleihin, trendeihin, poikkeaviin havaintoihin ja äkillisiin tasonmuutoksiin. Tämä monimuotoisuus tekee aikasarjamallinnuksesta haastavaa. Perinteiset tilastolliset menetelmät eivät aina kykene vastaamaan näihin haasteisiin riittävän tehokkaasti, mutta neuroverkot tarjoavat lupaavia mahdollisuuksia analyysiin. Vaikka neuroverkkojen kehitys on ollut nopeaa, niiden soveltaminen aikasarjoihin ei ole ongelmatonta.

Tässä luvussa tarkastellaan aikasarjojen analyysin keskeisiä haasteita, arvioidaan Transformer-mallien suorituskykyä ja vertaillaan niitä muihin koneoppimisen malleihin aikasarjojen ennustamisessa. Lisäksi luvussa pohditaan tulevia tutkimusmahdollisuuksia ja mahdollisia ratkaisuja Transformerien hyödyntämisessä aikasarjojen käsittelyssä.

4.1 Datan muutoksiin liittyvät haasteet

Aikasarja-analyysin sovelluksissa joudutaan kohtaamaan tilanteita, jotka johtuvat datan jatkuvista muutoksista. Ajan myötä opetus- ja testidatan jakaumat alkavat erota toisistaan, joka johtaa mallin ennustetarkkuuden heikkenemiseen. Jakauman muutokset (engl. *distribution shifts*) voidaan jakaa kolmeen päätyyppiin (Moreno-Torres et al., 2012):

- **Kovariaattisiirtymä** (engl. *covariate shift*): Kovariaattisiirtymä on tilanne, jossa syötteen x jakauma $P(x)$ muuttuu koulutus- ja käyttötilanteen välillä, mutta syötteen x ja ulostulon y välinen suhde (eli ehdollinen jakauma $P(y|x)$) säilyy ennallaan. Tämäntyyppinen siirtymä voi tapahtua esimerkiksi silloin, kun mallin koulutuksessa käytetty data on kerätty erilaisissa olosuhteissa kuin data, johon mallia sovelletaan myöhemmin. Esimerkiksi jos malli on koulutettu yhden alueen säädatan avulla ja sitä sovelletaan myöhemmin toiselle alueelle, sääparametrien jakaumat voivat erota alueiden välillä. Kuitenkin säämuuttujien ja tulosten välinen perussuhde, kuten esimerkiksi energiankulutuksen ennustaminen, pysyy ennallaan.

- **Prioritodennäköisyyssiirtymä** (engl. *prior probability shift*): Tässä tapauksessa

ulostulon y jakauma $P(y)$ muuttuu koulutus- ja testitilanteen välillä, mutta syötteen x ja lopputuloksen y välinen ehdollinen jakauma $P(x|y)$ pysyy samana. Tämä tarkoittaa, että vaikka syötteen antama informaatio lopputuloksesta säilyy ennallaan, eri lopputulosten yleisyys muuttuu ajan myötä.

- **Konseptisiirtymä** (engl. *concept shift*) tarkoittaa tilannetta, jossa syötteen ja tulosmuuttujan välinen yhteys $P(y|x)$ muuttuu ajan myötä. Konseptisiirtymä voi ilmetä dynaamisissa ympäristöissä, joissa mallit kohtaavat muuttuvia ilmiöitä, kuten kuluttajakäyttäytymisen tai markkinatrendien muutoksia. Tällöin sama syöte voi johtaa eri tulokseen. Usein konseptisiirtymät ovat syklisiä tai kausiluonteisia.

Yhteistä näille siirtymille on se, että ne heikentävät mallin ennustetarkkuutta ja luotettavuutta, jos niitä ei oteta huomioon mallin kehityksessä. Muutokset voivat tapahtua joko asteittain, jolloin ne kehittyvät hitaasti ajan myötä, tai äkillisesti, jolloin ne vaikuttavat tuloksiin välittömästi. Siirtymiä voi esiintyä jo mallin opetuksen aikana, jos koulutusaineisto ei vastaa käyttötilanteen todellista ympäristöä.

Datan muutoksiin liittyvien haasteiden hallintaan on kolme pääasiallista lähestymistapaa: 1) mallien kouluttaminen suurilla aineistoilla, 2) koulutetun mallin mukauttaminen kohdejakaumaan (engl. *domain adaptation*) ja 3) mallin uudelleenkoulutus (Huyen, 2022). Näiden menetelmien avulla voidaan parantaa mallin kykyä sopeutua muuttuviin olosuhteisiin ja säilyttää ennustetarkkuus.

4.2 Transformer-arkkitehtuurin haasteet

Transformer-mallien käytössä on haasteena niiden suuri laskennallinen teho- ja resurssitarve. Suurilla tietomäärillä koulutettaessa mallit vaativat merkittävää laskentakapasiteettia, mikä tekee niiden hyödyntämisestä kallista ja hidasta. Tämä laskennallinen monimutkaisuus johtuu itsehuomiomekanismista, jossa jokainen syöteosa vaikuttaa kaikkiin muihin syöteosiin. Tämän seurauksena sekvenssin pituuden kasvaessa sekä aika- että tilakompleksisuus kasvavat neliöllisesti, mikä tekee pitkien aikasarjojen käsittelystä erittäin raskasta.

Viimeaikaisissa Transformer-malleissa on kokeiltu erilaisia ratkaisuja laskennallisen tehokkuuden parantamiseksi. Itsehuomion laskentakuormaa voidaan vähentää esimerkiksi rajoittamalla huomion laskeminen vain osaan syötteestä (engl. *sparse attention*) tai jakamalla syöte pienempiin lohkoihin, jotka käsitellään erikseen (engl. *patching*). Aikasarjan

voi myös muuntaa taajuustasoon ja suorittaa laskennan vain valituille taajuuskomponenteille, mikä tehostaa laskentaa (T. Zhou et al., 2022).

Aikasarjoissa ajallinen jatkuvuus on säilytettävä koko syötteen läpi, jotta malli voi tunnistaa oikein pitkän aikavälin trendejä ja kausivaihteluita. Transformerin huomiomekanismi ei itsessään ymmärrä syöteosien positiota, vaan paikkatieto lisätään syötteeseen erillisenä paikkakoodauksena. Paikkatieto voi kuitenkin kadota syvissä malleissa, kun syöte kulkee kerroksien läpi (Lin et al., 2022). Tutkimuksissa on ehdotettu erilaisia ratkaisuja paikkatietojen säilyttämiseksi, kuten opittujen paikkakoodauksien hyödyntäminen, aikaleimojen liittäminen syötteeseen ja paikkatiedon lisääminen joka kerrokseen (Lin et al., 2022).

Transformerit ovat erittäin yleiskäyttöisiä, sillä ne eivät tee oletuksia datan rakenteellisista ominaisuuksista. Rakenteellisen ennakkotiedon puute altistaa ne kuitenkin helposti ylisovittumiselle silloin, kun käytössä on vähän dataa (Lin et al., 2022). Neuroverkot hyötyvät suuresta määrästä opetusaineistoa ja niiden suorituskyky paranee usein datan lisääntyessä. Mitä enemmän dataa, sitä paremmin malli voi oppia monimutkaisia riippuvuuksia. Kielimallien kohdalla tämä skaalaushyöty on helpommin saavutettavissa, sillä tekstiä on runsaasti tarjolla. Aikasarjoille ei sen sijaan ole olemassa vastaavan mittakaavan julkisia tietoaineistoja.

4.3 Suorituskykyvertailu ja pohdinta

Eri menetelmien suorituskyvyn vertailuissa on useissa tutkimuksissa hyödynnetty Autoformer -mallin artikkelissa (Wu et al., 2021) esiteltyjä kahdeksaa avointa aikasarjakokoelmaa. Näihin kuuluvat sää, liikenne, sähkö, influenssan kaltaiset sairaudet (ILI) sekä neljä erityyppistä sähkömuuntajista kerättyä aikasarjaa (ETTm1, ETTm2, ETTh1, ETTh2). Autoformer ylitti suorituskyvyssään vertailukohtina käytetyt RNN- ja CNN-mallit, sekä tilastotieteelliset mallit Prophet ja ARIMA.

Artikkelissa "Are Transformers Effective for Time Series Forecasting?" (Zeng et al., 2023) esiteltiin yksinkertainen lineaarinen neuroverkkomalli, DLinear, joka suoriutui paremmin kuin kaikki aiemmat Transformer-mallit useilla vertailudatoilla. Artikkelissa kyseenalaistettiin Transformer-mallien soveltuvuus aikasarjojen ennustamiseen.

Uudemmat Transformer-mallit, kuten PatchTST ja iTransformer (Liu et al., 2023), ovat puolestaan ylittäneet DLinearin suorituskyvyssä. Näissä malleissa hyödynnetty datan jakaminen osiin on osoittautunut tehokkaaksi. Tutkimukset ovat myös osoittaneet, että täy-

den huomion käyttö ei ole välttämätöntä kilpailukykyisten ennustetarkkuuksien saavuttamiseksi (Lin et al., 2022).

Aikasarjojen analyysissä on kuitenkin yhä epävarmuutta Transformerien tehokkuudesta. Useat tutkijat ovat osoittaneet, että yksinkertaisemmat mallit voivat tarjota riittävän tai jopa paremman tarkkuuden pienemmillä kustannuksilla ja monimutkaisuudella (Garza et al., 2023). Etenkin jos käytettävissä oleva data on rajallista, yksinkertaiset mallit voivat olla tehokkaampia kuin monimutkaiset syväoppimismallit.

Vaikka Transformer-arkkitehtuuri on saavuttanut merkittäviä läpimurtoja kielenkäsittelyssä, aikasarjojen analyysin osalta sen hyötyjä ei pidetä yhtä selkeinä. Kritiikki kohdistuu etenkin uusien innovaatioiden käytännön hyödyllisyyteen ja lupauksiin, jotka eivät aina täyty (Garza et al., 2023). On huomioitava, että useimmissa vertailuissa käytetyt aikasarja-aineistot ovat suhteellisen pieniä. Tämä herättää kysymyksen mallien yleistettävyydestä reaalimaailman sovelluksiin, joissa aineistot voivat olla monimutkaisempia ja sisältää enemmän vaihtelua. Se, että malli toimii hyvin laboratorio-olosuhteissa, ei välttämättä tarkoita, että se saavuttaa saman suorituskyvyn aidossa käyttöympäristössä.

Erilaiset aikasarjat tuovat mukanaan vaihtelevia haasteita riippuen sovelluskohteesta ja datan ominaispiirteistä. Esimerkiksi osakemarkkinoiden kurssidata on usein hyvin stokastista, mikä tekee pitkän aikavälin ennusteista haastavia verrattuna esimerkiksi sähkönkulutuksen aikasarjoihin, joissa esiintyy selkeitä toistuvia rakenteita. Tästä syystä menetelmän valinnan tulisi perustua sovelluskohteeseen: joissakin tapauksissa yksinkertaiset mallit voivat riittää selkeisiin rakenteisiin, mutta monimutkaisemmat menetelmät voivat olla tarpeen monimutkaisempien riippuvuuksien tunnistamiseen (Hyndman, 2018).

Aikasarjoihin tarkoitettujen Transformer-perusmallien kehittäminen voi olla yksi lupaava tutkimussuunta. Sen suuri haaste on kuitenkin riittävän aikasarjadata-aineiston saatavuus, joka on huomattavasti rajallisempaa kuin kielimalleilla (Zhou et al., 2023). Siirto-oppiminen voi auttaa tässä, sillä sen avulla voidaan hyödyntää aiemmin koulutettuja malleja ja helpottaa mallin kouluttamista pienemmillä aikasarja-aineistoilla.

Aikasarjojen ennustaminen on edelleen avoin ongelma, eikä mikään yksittäinen malli ole osoittautunut parhaaksi kaikissa sovelluksissa. Vaikka Transformereilla on saatu lupaavia tuloksia, sen soveltaminen ei välttämättä takaa parasta mahdollista suorituskykyä kaikkien aikasarjatyyppien ja sovelluskohteiden osalta. Monimutkaisten kausiluonteisten ja trendipiirteiden mallintaminen vaatii edelleen tutkimuksen jatkamista, jotta voidaan kehittää tehokkaita menetelmiä aikasarjojen moninaisten rakenteiden tunnistamiseen ja hyödyntämiseen.

5 Yhteenveto

Transformer-arkkitehtuuri on mullistanut luonnollisen kielen käsittelyn, mikä tekee siitä kiinnostavan myös aikasarjojen analyysissä. Aikasarjojen käsittely on keskeistä monilla sovellusalueilla, kuten taloudessa, sääennusteissa ja energiasektorin suunnittelussa, joissa tarkat ja luotettavat ennusteet ovat välttämättömiä.

Tutkielmassa tarkasteltiin Transformer-arkkitehtuurin perusominaisuuksia ja sen etuja verrattuna aiempiin neuroverkkoratkaisuihin. Transformerin huomiomekanismi mahdollistaa olennaisten riippuvuuksien tunnistamisen syötteestä sekä rinnakkaislaskennan hyödyntämisen, mikä parantaa mallin suorituskykyä. Arkkitehtuurin rakenteesta käsiteltiin myös yksityiskohtia, kuten kooderin ja dekooderin roolit, huomiomekanismin toimintaperiaatteet sekä paikkakoodauksen merkitys sarjamuotoisen tiedon käsittelyssä.

Arkkitehtuurin esittelyn jälkeen siirryttiin tarkastelemaan aikasarjojen analyysiä. Aikasarjoille ominaisia piirteitä, kuten trendejä, kausivaihteluita ja peräkkäisten havaintojen riippuvuutta toisistaan, käsiteltiin osana niiden analyysin vaatimuksia. Tämän jälkeen esiteltiin sekä perinteisiä että koneoppimiseen perustuvia menetelmiä, joita on käytetty aikasarjojen analyysiin. Lisäksi tarkasteltiin aikasarjoihin kehitettyjä Transformer-arkkitehtuurin muunnelmia.

Aikasarjojen käsittelyyn liittyy useita haasteita, jotka johtuvat sekä datan ominaisuuksista että Transformer-arkkitehtuurin alkuperäisestä rakenteesta. Näiden haasteiden ratkaisemiseksi arkkitehtuuria on muokattu esimerkiksi huomiomekanismia ja paikkakoodausta kehittämällä, uudistamalla rakennetta sekä syötettä lohkomalla tai muuntamalla taajuustasoon. Edistysaskeleista huolimatta lopullista läpimurtoa ei ole vielä saavutettu, mikä tekee jatkotutkimuksesta erittäin kiinnostavaa ja tarpeellista.

Tutkielmaa varten tehdyn laajan kirjallisuuskatsauksen perusteella Transformer-arkkitehtuurin suorituskyky on lupaava aikasarjojen analyysissä, mutta monet muut menetelmät voivat olla kilpailukykyisiä tietyissä tehtävissä. Menetelmän valinta riippuu usein sovelluskohteesta, sillä eri arkkitehtuurit soveltuvat erilaisten aikasarjojen analyysiin. Jos koulutusaineistoa on rajallisesti, myös perinteiset koneoppimismallit voivat osoittautua paremmaksi ratkaisuksi. Transformer-mallien joustavuus ja muokattavuus tarjoavat kuitenkin vahvan pohjan uusille innovaatioille, jotka voivat ratkaista nykyisiä haasteita.

Tekoälyn käyttö tutkielmassa

Tässä tutkielmassa hyödynnettiin ChatGPT 4o-mallia keskustelukumppanina Transformer-arkkitehtuurin tutkimusvaiheessa. Kielimallille esitettiin tarkentavia kysymyksiä niistä arkkitehtuurin osa-alueista, jotka eivät heti avautuneet luetuista artikkeleista.

Kirjoitusprosessissa ei käytetty tekoälytyökaluja.

Lähteet

Ba, J. L. (2016). "Layer normalization". *arXiv preprint arXiv:1607.06450*.

Bahdanau, D., Cho, K. ja Bengio, Y. (2015). "Neural machine translation by jointly learning to align and translate". Teoksessa: *3rd International Conference on Learning Representations, ICLR 2015*.

Bishop, C. M. ja Bishop, H. (2024). *Deep learning: Foundations and concepts*. Springer Nature.

Box, G. E., Jenkins, G. M., Reinsel, G. C. ja Ljung, G. M. (2015). *Time series analysis: forecasting and control*. John Wiley & Sons.

Box, G. ja Jenkins, G. M. (1970). *Time Series Analysis: Forecasting and Control*. Holden-Day.

Challu, C., Olivares, K. G., Oreshkin, B. N., Ramirez, F. G., Canseco, M. M. ja Dubrawski, A. (2023). "Nhits: Neural hierarchical interpolation for time series forecasting". Teoksessa: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 37. 6, s. 6989–6997.

Cho, K., Merriënboer, B. van, Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. ja Bengio, Y. (2014). "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". Teoksessa: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, s. 1724–1734.

Cowpertwait, P. S. (2009). *Introductory time series with R*. Springer Dordrecht Heidelberg.

Devlin, J. (2018). "Bert: Pre-training of deep bidirectional transformers for language understanding". *arXiv preprint arXiv:1810.04805*.

Dufter, P., Schmitt, M. ja Schütze, H. (2022). "Position information in transformers: An overview". *Computational Linguistics*, 48(3), s. 733–763.

Garza, A., Challu, C. ja Mergenthaler-Canseco, M. (2023). "TimeGPT-1". *arXiv preprint arXiv:2310.03589*.

Glorot, X., Bordes, A. ja Bengio, Y. (2011). "Deep sparse rectifier neural networks". Teoksessa: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop ja Conference Proceedings, s. 315–323.

He, K., Zhang, X., Ren, S. ja Sun, J. (2016). "Deep residual learning for image recognition". Teoksessa: *Proceedings of the IEEE conference on computer vision and pattern recognition*, s. 770–778.

Huyen, C. (2022). *Designing machine learning systems*. "O'Reilly Media, Inc."

Hyndman, R. (2018). *Forecasting: principles and practice*. OTexts.

Ilmatieteen laitos (2024). *Avoin data: ilman keskilämpötila ja sademäärä*. https://www.ilmatieteenlaitos.fi/havaintojen-lataus. (CC BY 4.0) Käytetty: 1.12.2024.

Jin, M., Wang, S., Ma, L., Chu, Z., Zhang, J. Y., Shi, X., Chen, P.-Y., Liang, Y., Li, Y.-F., Pan, S. et al. (2023). "Time-llm: Time series forecasting by reprogramming large language models". *arXiv preprint arXiv:2310.01728*.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X. ja Yan, X. (2019). "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting". *Advances in neural information processing systems*, 32.

Lin, T., Wang, Y., Liu, X. ja Qiu, X. (2022). "A survey of transformers". *AI open*, 3, s. 111–132.

Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L. ja Long, M. (2023). "itransformer: Inverted transformers are effective for time series forecasting". *arXiv preprint arXiv:2310.06625*.

Moreno-Torres, J. G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N. V. ja Herrera, F. (2012). "A unifying view on dataset shift in classification". *Pattern recognition*, 45(1), s. 521–530.

Murphy, K. P. (2022). *Probabilistic machine learning: an introduction*. MIT press.

Nasdaq (2024). *Avoin osakemarkkinadata: S&P 500 historiallinen data*. https://www.nasdaq.com/market-activity/index/spx/historical?page=1&rows_per_page=10&timeline=y5. Käytetty: 1.12.2024.

Nie, Y., Nguyen, N. H., Sinthong, P. ja Kalagnanam, J. (2023). *A Time Series is Worth 64 Words: Long-term Forecasting with Transformers*. arXiv: 2211.14730 [cs.LG]. URL: https://arxiv.org/abs/2211.14730.

Nielsen, A. (2019). *Practical time series analysis: Prediction with statistics and machine learning*. O'Reilly Media.

Prince, S. J. (2023). *Understanding deep learning*. MIT press.

Radford, A. (2018). *Improving language understanding by generative pre-training*. Technical Report. OpenAI.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W. ja Liu, P. J. (2020). "Exploring the limits of transfer learning with a unified text-to-text transformer". *Journal of machine learning research*, 21(140), s. 1–67.

Rangapuram, S. S., Seeger, M. W., Gasthaus, J., Stella, L., Wang, Y. ja Januschowski, T. (2018). "Deep state space models for time series forecasting". *Advances in neural information processing systems*, 31.

Siami-Namini, S., Tavakoli, N. ja Namin, A. S. (2018). "A comparison of ARIMA and LSTM in forecasting time series". Teoksessa: *2018 17th IEEE international conference on machine learning and applications (ICMLA)*. Ieee, s. 1394–1401.

Taylor, S. J. ja Letham, B. (2018). "Forecasting at scale". *The American Statistician*, 72(1), s. 37–45.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. ja Polosukhin, I. (2017). "Attention is all you need". *Advances in Neural Information Processing Systems*, s. 5998–6008.

Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J. ja Long, M. (2022). "Timesnet: Temporal 2d-variation modeling for general time series analysis". *arXiv preprint arXiv:2210.02186*.

Wu, H., Xu, J., Wang, J. ja Long, M. (2021). "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting". *Advances in neural information processing systems*, 34, s. 22419–22430.

Zeng, A., Chen, M., Zhang, L. ja Xu, Q. (2023). "Are transformers effective for time series forecasting?" Teoksessa: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 37. 9, s. 11121–11128.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H. ja Zhang, W. (2021). "Informer: Beyond efficient transformer for long sequence time-series forecasting". Teoksessa: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 35. 12, s. 11106–11115.

Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L. ja Jin, R. (2022). "Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting". Teoksessa: *International conference on machine learning*. PMLR, s. 27268–27286.

Zhou, T., Niu, P., Sun, L., Jin, R. et al. (2023). "One fits all: Power general time series analysis by pretrained lm". *Advances in neural information processing systems*, 36, s. 43322–43355.