Juuso Lehrbäck

Tarkkaavaisuusmekanismien kehittyminen neuroverkkokääntämisessä

Tietotekniikan kandidaatintutkielma

20. kesäkuuta 2023

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Juuso Lehrbäck

Yhteystiedot: jueelehr@student.jyu.fi

Ohjaaja: Tuomo Rossi

Työn nimi: Tarkkaavaisuusmekanismien kehittyminen neuroverkkokääntämisessä

Title in English: Evolution of attention mechanisms in neural machine translation

Työ: Kandidaatintutkielma

Opintosuunta: Tietotekniikka

Sivumäärä: 25+0

Tiivistelmä: Tämä kandidaatintutkielma on toteutettu kirjallisuuskatsauksena, jossa tutkitaan tarkkaavaisuusmekanismeja neuroverkkokääntämisessä. Tarkoituksena on selvittää, miten tarkkaavaisuusmekanismit ovat kehittyneet vuodesta 2015. Tarkasteltavina osa-alueina ovat käännösten laadun, arkkitehtuuristen erojen ja tehokkuuden kehittyminen. Tulosten perusteella tarkkaavaisusmekanismieja hyödynnetään laajemmin, ja niiden avulla saadaan entistä parempia käännöksiä. Lisäksi mekanismit ovat monipuolistuneet ja monimutkaistuneet, mutta niitä on mahdollista tehostaa eri keinoin.

Avainsanat: neuroverkkokääntäminen, NMT, neuroverkot, konekääntäminen, tarkkaavaisuusmekanismit

Abstract: This bachelor's thesis is carried out as a literature review and it researches attention mechanisms in neural machine translation. The purpose of this thesis is to find out how attention mechanisms have evolved since 2015. More precisely, this paper concentrates on the progress in translation quality, efficiency and architectural differences. According to the results, attention mechanisms are used more widely within NMT architectures and they have contributed to the improvement in translation quality. Additionally, these mechanisms have become more diverse and complicated, but they can be made more efficient.

Keywords: neural machine translation, NMT, neural networks, machine translation, attention mechanisms

Kuviot

Taulukot

Sisällys

1 Johdanto

Tämä tutkielma käsittelee erilaisia tarkkaavaisuusmekanismeja ja niiden kehittymistä neuroverkkokääntämisessä. Konekääntäminen on luonnollisen kielen kääntämistä toiselle kielelle automaattisesti tietokoneen avulla. Neuroverkot puolestaan ovat ihmisten aivojen hermoverkkoja jäljitteleviä verkkoja, jotka koostuvat tietoja käsittelevistä neuroneista. Tieto liikkuu neuroneista toisiin, ja lopulta viimeisissä neuroneissa on annettuun ongelmaan neuroverkon muodostama ratkaisu. Neuroverkkoja hyödyntävää konekääntämistä kutsutaan neuroverkkokääntämiseksi. Neuroverkkokääntäjät koostuvat enkoodereista, dekoodereista ja tarkkaavaisuusmekanismeista. Tarkkaavaisuusmekanismit ovat ihmisen tarkkaavaisuutta mukailevia tekniikoita, jotka ovat merkittävässä osassa neuroverkkokääntäjissä. Tarkkaavaisuuden avulla neuroverkko saadaan keskittymään niihin sanoihin, jotka vaikuttavat eniten käännettävään sanaan.

Tämän tutkielman tarkoituksena on vertailla erilaisia tarkkaavaisuusmekanismeja ja tuoda esille, miten ne ovat kehittyneet. Erityisesti tässä tutkielmassa keskitytään RNN-pohjaisiin ja Transformer-mallin tarkkaavaisuusmekanismeihin. Vertailu kattaa käännösten laadun, mekanismien arkkitehtuuriset erot sekä niiden tehokkuuden. Näin saadaan yleiskuva siitä, miten tarkkaavaisuusmekanismit ovat kehittyneet vuodesta 2015. Aihe on tärkeä, sillä konekääntäminen on nykymaailmassa paljon käytetty apuväline, ja siihen liittyvä teknologia on kehittynyt paljon viime vuosina. Neuroverkkokääntäminen ja tarkkaavaisuusmekanismit tulivat laajemmin käyttöön konekääntämisessä vasta 2010-luvun puolivälissä, minkä jälkeen kehitys on ollut nopeaa.

Toisessa luvussa käydään läpi aiheeseen keskeisesti liittyviä käsitteitä ja avataan yleisellä tasolla niiden teoriaa. Kolmannessa luvussa perehdytään tarkkaavaisuusmekanismeihin neuroverkkokääntämisessä ja tutustutaan niiden teoreettiseen pohjaan. Tämän ohella kuvaillaan tarkemmin joitakin valittuja tarkkaavaisuusmekanismeja. Tarkoituksena on tuoda esille erityyppisiä mekanismeja. Neljännessä luvussa vertaillaan edeltävässä luvussa esiteltyjä mekanismeja. Vertailua tehdään käännösten laadun, arkkitehtuuristen erojen ja tehokkuuden osalta. Viidennessä luvussa tehdään yhteenveto tutkielmassa tehdyistä johtopäätöksistä ja saaduista tuloksista.

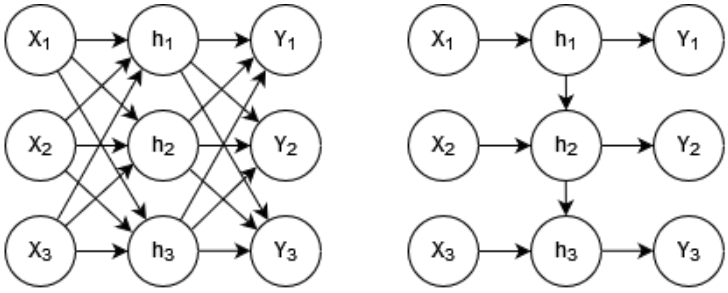
2 Neuroverkot ja neuroverkkokääntäminen

Tässä luvussa esitellään neuroverkot ja neuroverkkokääntäminen, jotka ovat tarkkaavaisuusmekanismien kannalta olennaisimmat käsitteet. Pohjimmiltaan neuroverkot ja konekääntäminen perustuvat koneoppimiseen (engl. *machine learning*), joka on yksi tekoälyn (engl. *artificial intelligence, AI*) toteutustapa. Se on nykyisin laajasti käytössä ja perustuu siihen, että ohjelma oppii sille annetusta datasta riippuvuussuhteita ja muita yhteyksiä. Riippuen siitä, miten koneoppimisohjelmaa opetetaan, oppiminen voi olla ohjattua, ohjaamatonta tai vahvistettua. Kone on ihmistä parempi käymään läpi valtavia aineistoja ja löytämään sieltä yhteyksiä. Koneoppimiseen on monia erilaisia toteutuksia, joista yksi on keinotekoinen neuroverkko (engl. *artificial neural network*). Neuroverkkoja käsitellään seuraavassa alaluvussa.

2.1 Neuroverkot

Neuroverkot ovat neuroneista muodostuvia verkkoja, jotka koostuvat useista kerroksista. Ne jäljittelevät karkeasti ihmisten aivojen toimintaa. Neuroverkon ensimmäinen kerros on syötekerros (engl. *input layer*), johon syötetään tarvittavat tunnetut arvot. Viimeinen kerros on puolestaan ulostulokerros (engl. *output layer*), johon neuroverkon antama tulos tulee. Näiden kahden välissä on piilokerroksia (engl. *hidden layer*) (Koehn 2017, 11–12.). Vähintään kaksi piilokerrosta sisältävää neuroverkkoa kutsutaan syväksi neuroverkoksi (engl. *deep neural network*) (Koehn 2017, 14). Neuroverkkojen arkkitehtuuria on havainnollistettu kuviossa 1. Seuraavaksi esitellään tavallisen eteenpäin kytketyn neuroverkon (engl. *feedforward neural network, FFN*) toimintaa.

Jokainen eteenpäin syöttävän neuroverkon piilo- ja ulostulokerros koostuu neuroneista, joihin tulee syöte edellisen kerroksen neuroneilta. Neuronissa syötteet kerrotaan yhteyksien painokertoimilla (engl. *weight*), ja nämä tulot lasketaan yhteen. Summaan lisätään vielä neuronin vakiotermi (engl. *bias*). Saatu arvo viedään aktivointifunktioon, joka muuttaa summan lineaarisesta muodosta epälineaariseen. Aktivointifunktio voi olla esimerkiksi sigmoidfunktio, hyperbolinen tangentti tai ReLu-funktio (Koehn 2017, 14.). Näin saadaan neuronin lopullinen tulos. Lopulta ulostulokerroksen neuroneihin muodostuu neuroverkon muodosta-



Kuvio 1. Kaksi erilaista neuroverkkoa. Vasemmalla on eteenpäin kytketty ja oikealla takaisinkytketty neuroverkko.

ma vastaus tehtävään (Koehn 2017, 11.).

Jotta neuroverkko kykenisi oppimaan, sitä on opetettava. Tätä varten sille annetaan paljon aineistoa. Neuroverkon tarkoituksena on oppia muodostamaan yhteyksiä aineiston syötteiden ja tulosten väliltä. Tämä tapahtuu minimoimalla virhefunktiota, joka muodostuu vähintään neuroverkon antaman tuloksen ja oikean arvon erotuksesta. Minimointiin voidaan käyttää esimerkiksi vastavirta-algoritmia (engl. *backpropagation*) ja gradienttimenetelmää (engl. *gradient descent*). Vastavirta-algoritmissa neuronien painoja päivitetään viimeisestä kerroksesta lähtien siten, että muutosten suuruudet lasketaan gradienttimenetelmällä. Gradienttimenetelmässä etsitään neuronien painokertoimien funktioiden gradientteja ja muutetaan arvoja gradienteista vastakkaiseen suuntaan. Näiden algoritmien avulla neuronien painokertoimia ja vakiotermejä muutetaan viimeisestä kerroksesta lähtien siten, että virhefunktion arvo pienenee (Koehn 2017, 16.).

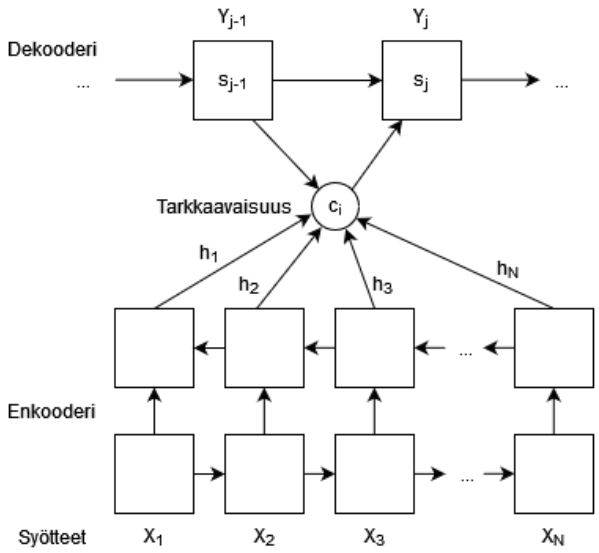
Edellä esitelty eteenpäin kytketty neuroverkko ei ole ainoa neuroverkkoarkkitehtuuri. Muita yleisiä arkkitehtuureja ovat takaisinkytketyt neuroverkot (engl. *recurrent neural network, RNN*) ja konvoluutioneuroverkot (engl. *convolutional neural network*). Erityisesti takaisinkytketyt neuroverkot ovat yleisiä seuraavassa alaluvussa esiteltävissä neuroverkkokääntäjissä. Tällaisten neuroverkkojen suurin ero eteenpäin kytkettyihin neuroverkkoihin on niiden sisältämät piilotilat. Piilotilojen mukana tieto liikkuu aika-askelittain piilokerrosten kautta johonkin suuntaan. Näin edelliset piilotilat vaikuttavat muihin saman kerroksen neuroneihin (Koehn 2017, 44–46.).

2.2 Neuroverkkokääntäminen

Konekääntäminen tarkoittaa luonnollisen kielen kääntämistä toiselle kielelle koneellisesti. Konekääntäjiä on erilaisia, kuten perinteisiä tilastollisia (engl. *statistical machine translation, SMT*), sääntöihin perustuvia (*rule-based machine translation*) ja neuroverkkoihin pohjautuvia. Kun neuroverkkoja käytetään konekääntämiseen, puhutaan neuroverkkokääntämisestä (engl. *neural machine translation, NMT*). Neuroverkkokääntämisen takana oleva teoria kehitettiin jo 1980- ja 1990-luvuilla, mutta silloin tietokoneiden laskentakapasiteetti ei riittänyt niiden toteuttamiseen (Koehn 2017, 10.). Laskentakapasiteetin huiman kasvun myötä neuroverkkokääntäjiä on pystytty ottamaan käyttöön, ja niiden tuottamien käännösten laatu on parantunut nopeasti vuodesta 2015 lähtien (Koehn 2017, 16).

Neuroverkkokääntäjässä yhden lauseen kääntämistä varten lasketaan ehdollinen todennäköisyys $p(y|x)$, jossa x on käännettävä lause ja y käännös (Luong, Pham ja Manning 2015). Luonnollisesti todennäköisin vaihtoehto valitaan käännökseksi (Bahdanau, Cho ja Bengio 2015). Jokainen käytetyn sanaston sana esitetään one-hot-vektorina, jossa on yksi ykkönen ja loput arvot ovat nollia. Lisäksi jokaiselle sanalle annetaan vektorimuotoinen sanaupotus (engl. *word embedding*), jotka ovat samankaltaisia toisiaan semanttisesti muistuttavilla sanoilla. Tavallisesti sanastoon valitaan rajallinen määrä yleisimpiä sanoja, jotta vektorien koko ei paisuisi liian suureksi (Koehn 2017, 38.).

Yleensä neuroverkkokääntäjässä on enkooderi ja dekooderi kuten kuviossa 2 (Luong, Pham ja Manning 2015). Ne koostuvat usein monesta kerroksesta (Maruf, Martins ja Haffari 2019), ja niiden välissä on tarkkaavaisuusmekanismi, jota tarkastellaan seuraavassa luvussa. Enkooderi muuttaa annetun diskreetin lausesyötteen piilokerroksissa jatkuvaan muotoon (Maruf, Martins ja Haffari 2019). Dekooderin tehtävä on puolestaan valita sille annettujen tietojen pohjalta todennäköisin käännös (Koehn 2017, 55). Tämä on niin sanottu sekvenssistä sekvenssiin -malli (engl. *sequence-to-sequence model*) (Koehn 2017, 54). Takaisinkytketty neuroverkko on suoraviivainen valinta neuroverkkokääntäjän enkooderiin ja dekooderiin (Luong, Pham ja Manning (2015), Bahdanau, Cho ja Bengio (2015)), koska tällaisessa neuroverkossa sanaa käännettäessä edellisten sanojen käännösten piilotilat ovat tiedossa (Koehn 2017, 45). Enkooderissa takaisinkytketty neuroverkko on yleensä kaksisuuntainen (Koehn 2017, 55), koska seuraavien syötteiden piilotiloilla on merkitystä. Tämän avulla konekään-



Kuvio 2. Yksinkertaisen tarkkaavaisuutta käyttävän neuroverkkokääntäjän arkkitehtuuri.

täjä tietää käännöksen kontekstin (Koehn 2017, 55). Lisäksi se mahdollistaa mielivaltaisen pitkien lauseiden käsittelyn (Koehn 2017, 44).

Käytännössä enkooderissa ja dekooderissa on yksinkertaisen takaisinkytketyn neuroverkon sijaan joko pitkä lyhytkestomuisti (engl. *long short-term memory, LSTM*) tai sen yksinkertaisempi muoto gated recurrent unit (*GRU*) (Luong, Pham ja Manning 2015). Ne ovat erilaisia RNN-arkkitehtuureja. Nämä porteista koostuvat neuroverkot yhdistävät aiempia tiloja ja uusia syötteitä (Koehn 2017, 46–49.). Niitä käytetään sitä varten, että konekääntäjä osaa paremmin kääntää sellaisia lauseita, joissa kaukana olevat sanat vaikuttavat toisiinsa (Koehn 2017, 46–49). Tämä johtuu siitä, että LSTM:n ja GRU:n avulla voidaan estää suurissa RNN-verkoissa tapahtuvia räjähtävän tai katoavan gradientin ongelmia (Hochreiter ja Schmidhuber 1997) (Chung ym. 2014).

Neuroverkkokääntäjää opetetaan sille annettavalla kieliaineistolla, josta neuroverkko oppii käyttämällä n-grammeja. N-grammit ovat n:n sanan mittaisia jaksoja, joiden avulla neuroverkko päättelee, mitkä sanat seuraavat todennäköisimmin mitäkin. Oppiminen tapahtuu päivittämällä neuronien painokertoimia vastavirta-algoritmilla (Koehn 2017, 40.). Opettamista varten käytetään gradienttimenetelmää, yleensä stokastista gradienttimenetelmää (engl. *stochastic gradient descent, SGD*) (Bahdanau, Cho ja Bengio 2015) tai jotain sen muunnelmaa.

Muunnelma voi olla esimerkiksi Adam (Vaswani ym. 2017). Yksi neuroverkkokääntäjien huono puoli on se, että niiden opettaminen on laskentateholtaan kallista (Koehn 2017, 42). Lisäksi ne tarvitsevat muita konekääntäjiä enemmän pohja-aineistoa, jotta niillä voidaan saada kohtalaisen laadukkaita käännöksiä (Koehn 2017, 105–106).

Konekäännösten laatua mitataan usein BLEU-arvioinnilla. BLEU-asteikko perustuu siihen, kuinka tarkasti tuotettu käännös vastaa ammattikääntäjän tuottamaa käännöstä (Papineni ym. (2002).). Asteikko sisältää välin 0–1, mutta yleensä arvot skaalataan välille 0–100. Mitä suurempi arvo, sitä laadukkaampana käännöstä pidetään (Papineni ym. (2002)). Neuroverkkokääntäjillä saadaan nykyisin parempia käännöksiä kuin muilla konekääntäjillä, minkä vuoksi niitä käytetään laajalti (Stasimioti ym. 2020). Esimerkiksi Google Kääntäjä on hyödyntänyt neuroverkkoja vuodesta 2016 lähtien (Le ja Schuster 2016). DeepL on toinen neuroverkkoihin ja tarkkaavaisuusmekanismeihin perustuva konekääntäjä (“How does DeepL work?” 2021).

3 Tarkkaavaisuusmekanismit

Tarkkaavaisuus tarkoittaa neuroverkkokääntämisessä sellaista mekanismia, joka ottaa huomioon ainakin osan aiemmin käännetystä tekstistä ja käännettävästä syötteestä. Tarkkaavaisuusmekanismilla lasketaan, kuinka suuri yhteys käännettävällä sanalla on muihin syötteen sanoihin. Ensimmäisessä alaluvussa käydään läpi neuroverkkokääntämisessä käytettävien tarkkaavaisuusmekanismien yleinen toimintaperiaate ja matemaattinen pohja. Toisessa alaluvussa laajennetaan ensimmäisessä alaluvussa käsiteltyä tarkkaavaisuusmekanismia tarkastelemalla muita RNN-pohjaisia toteutuksia. Kolmannessa alaluvussa esitellään selvästi muista malleista eroava monipäiseen tarkkaavaisuuteen perustuva Transformer-malli ja sen muunnelmia.

3.1 Yleinen tarkkaavaisuusmekanismi

Tässä alaluvussa käsitellään yksinkertaista tarkkaavaisuusmekanismia. Tarkkaavaisuutta on ensimmäisen kerran käytetty neuroverkkokääntämisessä vuonna 2015 (Bahdanau, Cho ja Bengio 2015). Tarkkaavaisuusmekanismit mahdollistivat sen, että neuroverkkokääntäjillä saatiin muihin konekääntäjätyyppeihin verrattavissa olevia tuloksia (Koehn 2017, 10). Tarkkaavaisuudella on monia hyviä ominaisuuksia neuroverkkokääntämisessä. Niiden ansiosta pitkien lauseiden ja erisnimien kääntämisen laatu on korkeampi kuin muilla mekanismeilla (Luong, Pham ja Manning (2015), Bahdanau, Cho ja Bengio (2015)). Toisaalta tarkkaavaisuusmekanismin lisääminen arkkitehtuuriin monimutkaistaa konekääntäjän toteutusta (Koehn 2017, 54).

Tarkkaavaisuusmekanismi on yleensä neuroverkkokääntäjän dekoooderissa oleva funktio, joka laskee yhteyden jokaisen syötteenä olevan sanan ja dekooderin edeltävän piilotilan välillä. Tarkkaavaisuuskerros toimii kuten neuroverkon eteenpäin kytketty kerros, ja tarkkaavaisuusfunktio suoritetaan jokaisen dekooderin neuronin kohdalla (Koehn 2017, 57.). Tuloksena saadaan kontekstivektori (engl. *context vector*), jota käytetään dekooderissa laskemaan todennäköisin käännös sanalle. Funktio tarvitsee enkooderista sekä edeltävistä että seuraavista piilotiloista koostuvan vektorin, jota kutsutaan tässä tapauksessa annotaatiovektoriksi

(engl. *annotation vector*). Tämän vektorin saadakseen enkooderin takaisinkytketty neuroverkko on usein kaksisuuntainen, kuten edeltävässä luvussa kerrottiin (Bahdanau, Cho ja Bengio (2015).). Tarkkaavaisuuden sijoittuminen yksinkertaisessa neuroverkkokääntäjässä on esitetty kuviossa 2.

Tarkkaavaisuusmekanismin tarkoituksena on tuottaa kontekstivektori c, kun sille annetaan syötteiden kuvauksia vastaava annotaatiovektori h ja neuroverkon edellinen piilotila s. Mekanismissa tuotetaan tarkkaavaisuusvektori a, joka kuvaa käännettävän sanan ja muiden sekvenssissä olevien sanojen välisen yhteyden tärkeyttä (Koehn 2017, 57.). Tarkemmin sanoen arvo a_{ij} kuvaa sitä, kuinka suuri yhteys annotaatiolla h_j on tulevaan käännökseen (Bahdanau, Cho ja Bengio 2015). Kontekstivektorissa annotaatiot on kerrottu näillä tarkkaavaisuusvektorissa olevilla painoilla. Matemaattisesti tämä tarkoittaa yksinkertaisesti seuraavaa, kun w_a sekä u_a ovat tarkkaavaisuuskerrosta vastaavan neuroverkon kerroksen painoja ja b_a on neuronin vakiotermi:

$$e_{ij} = a(s_{i-1}, h_j) = w_a^T s_{i-1} + u_a^T h_j + b_a. \tag{3.1}$$

(Koehn 2017, 57)

Yllä oleva arvo normalisoidaan, jolloin saadaan tarkkaavaisuusarvo a_{ij}:

$$a_{ij} = \frac{exp(e_{ij})}{\sum_{k=1}^{N} exp(e_{ik})}.$$

(Koehn 2017, 57)

Kontekstivektoriksi saadaan:

$$c_i = \sum_{j=1}^{N} a_{ij} h_j.$$

(Koehn 2017, 57)

Vaihtoehto funktiolle 3.1 on

$$a(s_{i-1}, h_j) = v_a^T tanh(W_a s_{i-1} + U_a h_j). \tag{3.2}$$

(Bahdanau, Cho ja Bengio 2015)

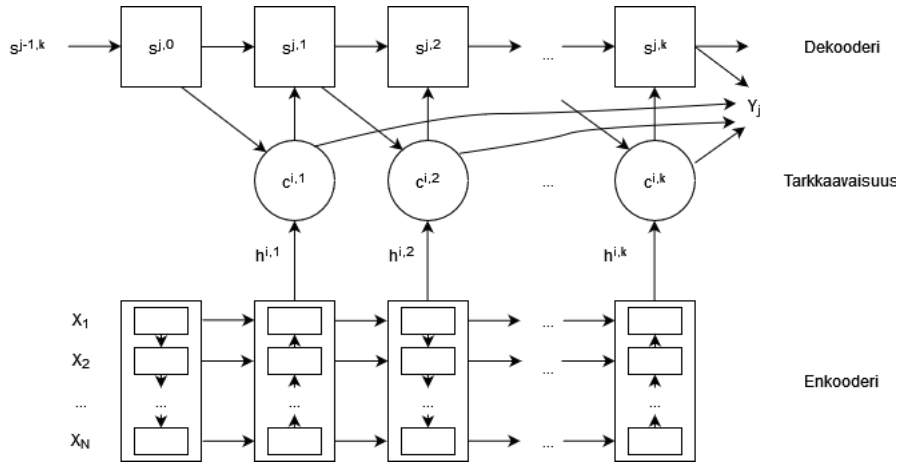
3.2 Erilaisia RNN-pohjaisia malleja

Tässä alaluvussa käsitellään erilaisia takaisinkytkettyihin neuroverkkoihin pohjautuvia tarkkaavaisuusmekanismeja. Esiteltäviä mekanismeja ovat syvä tarkkaavaisuus (engl. *deep attention*), hienorakenteinen tarkkaavaisuus (engl. *fine-grained attention*) ja kaksikielinen tarkkaavaisuus (engl. *bilingual attention*). Kaikki eroavat jollakin tavalla toisistaan ja edellisessä alaluvussa esitellystä tarkkaavaisuudesta.

Choi, Cho ja Bengio (2018) esittelivät hienorakenteisen tarkkaavaisuuden. Siinä jokaiselle annotaatiovektorin alkiolle annetaan yksi skalaariarvoinen kerroin sen sijaan, että koko annotaatiovektorilla h_j olisi vain yksi kerroin. Käytännössä tämä tarkoittaa sitä, että kaavassa 3.1 lasketaan arvo e_{ij} jokaista annotaatiovektorin dimensiota d kohden, jolloin a_{ij} on vektori skalaariarvon sijaan. Kontekstivektori c_i saadaan tarkkaavaisuusvektorin ja annotaatiovektorin alkioittaisena kertolaskuna. Tarkoituksena on mahdollistaa se, että tarkkaavaisuusmekanismi voi keskittyä sanojen erilaisiin tulkintoihin. Lisäksi heidän mallissaan sanaupotus annetaan parametrina tarkkaavaisuusfunktiolle.

Zhang, Xiong ja Su (2018) puolestaan muuttivat konekääntäjän rakennetta laajemmin. He esittelivät syvän tarkkaavaisuuden, joka perustuu useisiin tarkkaavaisuuskerroksiin neuroverkkokääntäjässä. Jokainen tarkkaavaisuuskerros vastaa yhtä enkooderi- ja dekooderikerrosta. Tarkkaavaisuuskerros k saa parametrinsa sitä vastaavalta enkooderikerrokselta ja edeltävältä dekooderikerrokselta. Dekooderi saa parametrinsa vastaavasti tarkkaavaisuuskerrokselta k ja edelliseltä dekooderikerrokselta. Näin enkooderi vaikuttaa dekooderiin kerroksittain. He käyttävät mallissaan tarkkaavaisuusfunktiota 3.2 sillä erotuksella, että kaikki parametrit riippuvat kerroksesta k. Käännös valitaan dekooderin viimeisen piilotilan ja kaikkien tarkkaavaisuuskerrosten tulosten pohjalta. Syvään tarkkaavaisuuteen perustuva arkkitehtuuri esitetään kuviossa 3.

Kang ym. (2022) esittelivät kaksikielisen tarkkaavaisuuden, jossa tarkkaavaisuus huomioi kaikki dekooderin piilotilat vain viimeisimmän sijaan. Arkkitehtuurissa tarkkaavaisuusmekanismi käyttää lähdekielen annotaatiovektorin h lisäksi käännöspuolen vastaava vektoria s, joka muuttuu jokaisen aika-askeleen kohdalla. Vektori s toimii näin ollen dekoodaushistoriana. Kaksikielisessä tarkkaavaisuudessa näille vektoreille annetaan omat painokertoimensa



Kuvio 3. Syvään tarkkaavaisuuteen perustuva arkkitehtuuri.

kontekstivektoria laskettaessa. Heidän mukaansa dekoodaushistorian lisääminen tarkkaavaisuuteen parantaa käännösten tarkkuutta, koska käännöksissä itsessään on paljon lisäinformaatiota verrattuna vain käännettävässä tekstissä olevaan tietoon. Mallissa on yksi tarkkaavaisuuskerros jokaisessa dekooderin kerroksessa. Arkkitehtuuri hyödyntää myös two-hop-tarkkaavaisuutta. Tarkkaavaisuusarvo muodostetaan tavallisen tarkkaavaisuuden ja two-hop-tarkkaavaisuuden painotettuna summana. Two-hop-tarkkaavaisuus perustuu dekooderin piilotilojen välisiin epäsuoriin yhteyksiin, jotka saadaan laskettua selvittämällä suorien yhteyksien merkittävyydet.

Heidän mallissaan käytetään funktion 3.1 sijaan seuraavaa tarkkaavaisuusfunktiota, kun l on nykyinen kerros, d_s on piilotilavektorin pituus ja W_a^l sekä U_a^l ovat vastaavan neuroverkon kerroksen kerroinmatriiseja:

$$e_{ik}^l = \frac{1}{\sqrt{d_s}}(s_i^{l-1}W_a^l)(m_k^{l-1}U_a^l). \tag{3.3}$$

(Kang ym. 2022)

Alkio m_k^{l-1} puolestaan määritellään seuraavasti, kun N on syötevektorin koko:

$$m_k^{l-1} = \begin{cases} h_k^{l-1} & , \text{jos } k \leq N \\ s_{k-N}^{l-1} & , \text{jos } k > N. \end{cases} \tag{3.4}$$

(Kang ym. 2022)

Two-hop-tarkkaavaisuus määritellään seuraavasti, kun λ on hyperparametri ja a^l_{ik} on normalisoitu arvosta e^l_{ik} samaan tapaan kuin yleisessä tarkkaavaisuusmekanismissa:

$$a^l_{ik} = a^l_{ik} + \frac{\lambda}{i-k+1} \sum_{k'=k+1}^{i-1} a^l_{ik'} \cdot a^l_{k'k}. \tag{3.5}$$

(Kang ym. 2022)

Lisäksi tarkkaavaisuusarvot normalisoidaan seuraavasti:

$$a^l_{ik} = \frac{a^l_{ik}}{\sum_{k'=1}^{i} a^l_{ik'}}. \tag{3.6}$$

(Kang ym. 2022)

Kontekstivektori c^l_i saadaan seuraavasti, kun W^l_b sekä U^l_b ovat neuroverkon kerroksen painomatriiseja ja x on välillä $[0,1]$;

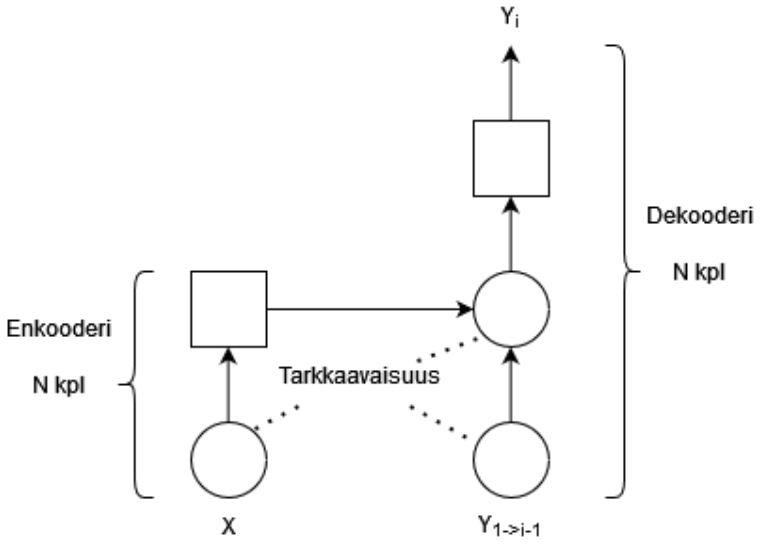
$$c^l_i = x \sum_{j=1}^{N} a^l_{ij}(h^{l-i}_j W^l_b) + (1-x) \sum_{i'=1}^{i} a^l_{ii'}(s^{l-i}_{i'} U^l_b). \tag{3.7}$$

(Kang ym. 2022)

3.3 Transformer-mallit

Kaikki neuroverkkokääntäjät eivät käytä takaisinkytkettyihin neuroverkkoihin ja niihin sisällytettyyn tarkkaavaisuuteen perustuvaa arkkitehtuuria. Tällaisista erilaisista arkkitehtuureista yleisin ja kehittynein on niin sanottu Transformer-malli. Sitä on pidetty usein parhaita käännöksiä tuottavana neuroverkkokääntäjien arkkitehtuurina (You, Sun ja Iyyer 2020). Alkuperäisen Transformer-mallin tarkkaavaisuuden lisäksi tässä alaluvussa esitellään sen muunnelmat valikoiva tarkkaavaisuus (engl. *selective attention*) ja kovakoodattu gaussilainen tarkkaavaisuus (engl. *hard-coded gaussian attention*).

Vaswani ym. (2017) esittelivät Transformer-mallin vuonna 2017. Se eroaa RNN-pohjaisista tarkkaavaisuusmekanismia käyttävistä arkkitehtuureista siinä, että se ei käytä takaisinkytkettyjä neuroverkkoja, vaan tavallisia eteenpäin kytkettyjä neuroverkkoja. Malli hyödyntää monipäistä tarkkaavaisuutta (engl. *multi-head attention*) enkooderin ja dekooderin yhdistävässä kerroksessa. Transformer-arkkitehtuurin enkooderissa ja dekooderissa käytetään lisäksi itseistarkkaavaisuutta (engl. *self-attention*), joka on tässä tapauksessa yksi monipäisen tarkkaavaisuuden muoto.



Kuvio 4. Yksinkertaisesti esitetty Transformer-malli.

Kuten kuviosta 4 voidaan havaita, Transformer-arkkitehtuuri koostuu päällekkäin pinotuista enkooderi- ja dekooderikerroksista. Molempia kerroksia on yhteensä N kappaletta, ja jokaisessa enkooderikerroksessa on kaksi alikerrosta. Ensimmäinen alikerros on monipäinen tarkkaavaisuus ja toinen eteenpäin syöttävä neuroverkko. Dekooderikerros on samanlainen, mutta siinä näiden kahden alikerroksen edellä on verhottu monipäinen tarkkaavaisuus (engl. *masked multi-head attention*), joka varmistaa tarkkaavaisuuden keskittymisen vain edeltäviin käännöksiin (Vaswani ym. (2017).).

Itseistarkkaavaisuus tarkoittaa sellaista tarkkaavaisuutta, jonka kaikki parametrit tulevat samasta paikasta. Parametrit ovat joko syötteiden kuvauksia tai tuotettuja käännöksiä riippuen siitä, onko tarkkaavaisuus enkooderissa vai dekooderissa. Käytännössä itseistarkkaavaisuus laskee jokaisen sanaparin välisen yhteyden. Monipäinen tarkkavaisuus puolestaan on sellainen tarkkaavaisuusmekanismi, jossa tarkkaavaisuusfunktio suoritetaan useita kertoja samoilla syötteillä, mutta niiden toisistaan eroavilla lineaariprojektioilla. Tässä lineaariprojektio tarkoittaa yksinkertaistetusti syötteen ja neuroverkon kerroksen opittujen painojen matriisin pistetuloa. Tarkkaavaisuuden päillä saadaan eri tuloksia, jotka ketjutetaan yhteen. Tämä mahdollistaa sanojen erilaisten mahdollisten kuvausten huomioon ottamisen (Vaswani ym. (2017).).

Heidän mallissaan tarkkaavaisuusfunktio on funktion 3.1 sijaan seuraavanlainen itseistarkkaavaisuuskerroksissa:

$$a(h_j, h_k) = \frac{1}{\sqrt{|h|}} h_j h_k^T. \tag{3.8}$$

(Koehn 2017, 98)

Enkooderin ja dekooderin yhdistävässä tarkkaavaisuuskerroksessa s_i on dekooderin edellisen kerroksen piilotila, kun taas h_k on enkooderin annotaatiovektori:

$$a(s_i, h_k) = \frac{1}{\sqrt{|h|}} s_i h_k^T. \tag{3.9}$$

(Koehn 2017, 99)

Maruf, Martins ja Haffari (2019) esittelivät Transformeriin valikoivan ja kontekstitietoisen tarkkaavaisuuden. Se rajaa kontekstia käyttämällä harvaa tarkkaavaisuutta. Tällainen tarkkaavaisuus huomioi vain ne osat konteksista, jotka ovat oleellisimpia käännettävän lauseen kannalta. Epäoleellisille osille annetaan painoarvoksi 0. Tämän muutoksen ohella he muokkasivat monipäistä tarkkaavaisuutta ja itseistarkkaavaisuutta käsittelemällä lausetasoa sanatason ohella. Kontekstitarkkaavaisuus voidaan sijoittaa joko enkooderi- tai dekooderipinon päälle omana kerroksenaan. Kontekstitarkkaavaisuuskerros sisältää kaksi alikerrosta, jotka ovat kontekstitarkkaavaisuus ja eteenpäin syöttävä neuroverkkokerros.

You, Sun ja Iyyer (2020) esittelivät Transformer-malliin kovakoodatun gaussilaisen tarkkaavaisuuden. Siinä enkooderin ja dekooderin itseistarkkaavaisuus korvataan kovakoodatulla Gaussin jakaumalla, eikä monipäinen tarkkaavaisuus opi uusia parametreja. Tarkkaavaisuusfunktio on käytännössä Gaussin jakauma funktion 3.8 sijaan. Jakauman keskipiste on heidän mallissaan nykyinen, edeltävä tai seuraava sana ja keskihajonta on 1. Tarkkaavaisuuden eri päitä varten annetaan jakaumille omat keskiarvot.

4 Tulokset

Tässä luvussa vertaillaan aiemmin tässä tutkielmassa esiteltyjä tarkkaavaisuusmekanismien toteutuksia ja tuodaan esille erilaisia tuloksia. Vertailu keskittyy käännösten laatuun, tarkkaavaisuusmekanismien arkkitehtuurisiin eroihin ja niiden tehokkuuteen. Kaikenkattavaa vertailua ei ole mahdollista tehdä kandidaatintutkielman puitteissa, joten keskitytään näihin konekääntäjien ja tarkkaavaisuusmekanismien kannalta olennaisiin seikkoihin. Pohjaverrokkeina toimivat Luongin, Phamin ja Manningin (2015) sekä Bahdanaun, Chon ja Bengion (2015) esittelemät varhaiset ja suhteellisen yksinkertaiset tarkkaavaisuusmekanismit. Ne antavat hyvän pohjatason, joihin muita uudempia mekanismeja voidaan verrata.

4.1 Käännösten laatu

Taulukko 1. Neuroverkkoarkkitehtuurien BLEU-tulokset.

			englanti–saksa		englanti–ranska	
	aineisto	algoritmi	sanat	BLEU	sanat	BLEU
Luong, Pham ja Manning (2015)	WMT14	SGD	50	23,0	-	-
Bahdanau, Cho ja Bengio (2015)	WMT14	SGD	-	-	30	28,45
Vaswani ym. (2017)	WMT14	Adam	37	28,4	32	41,0
Choi, Cho ja Bengio (2018)	WMT15	Adam	30	23,74	-	-
Zhang, Xiong ja Su (2018)	WMT14	Adam	30	26,45	40	39,88
Maruf, Martins ja Haffari (2019)	WMT16	Adam	30	24,84*	-	-
You, Sun ja Iyyer (2020)	WMT14	?	?	26,3	?	39,1
Kang ym. (2022)	WMT14	Adam	50	29,5	50	42,2

Sanojen lukumäärät ilmoitettu tuhansissa. * = Paras saatu tulos uutistestiaineistolla.

Taulukosta 1 voidaan huomata, että BLEU-tulosten perusteella kääntäjien taso nousi nopeasti vuoden 2015 jälkeisinä vuosina. Tämä perustuu usein tutkimuksissa esiintyvien kieliparien englanti–saksa ja englanti–ranska tarkasteluun. Jokaisessa tarkastellussa tutkimuksessa oli

mukana ainakin toinen näistä kielipareista. Transformer-mallin itseistarkkaavaisuus ja monipäinen tarkkaavaisuus nostivat tason korkealle, minkä jälkeen laatu ei ole juurikaan parantunut BLEU-tulosten perusteella. Tämä voisi tarkoittaa sitä, että konekääntäjien taso lähestyy parasta saavutettavissa olevaa tasoa. Voidaan myös huomata, että RNN-mallit ovat kehittyneet huomattavasti vuoden 2015 jälkeen ja parhaimmillaan ohittaneet Transformer-malliin perustuvien konekääntäjien tason. Molemmantyylisillä malleilla voidaan saada nykyään hyviä käännöstuloksia.

Esimerkiksi kieliparin englanti–saksa BLEU-tulokset ovat nousseet 7:ssä vuodessa Luongin, Phamin ja Manningin (2015) arkkitehtuurin arvosta 23,0 Kangin ym. (2022) arkkitehtuurin arvoon 29,5. Näiden välissä Transformer-arkkitehtuuri oli saavuttanut arvon 28,4 (Vaswani ym. 2017). RNN-pohjaisissa arkkitehtuureissa kehitys tapahtui hitaammin. Choi, Cho ja Bengio (2018) paransivat arvoon 23,74 WMT15-aineistolla, ja Zhang, Xiong ja Su (2018) nostivat sen edelleen arvoon 26,45. Maruf, Martins ja Haffari (2019) käyttivät edelleen eri opetus- ja testiaineistoa, jonka pohjalta he tuottivat tuloksia väliltä 24–25. Se suoriutui tutkimuksessa Transformer-mallia paremmin. You, Sun ja Iyyer (2020) puolestaan saivat tutkimuksessaan BLEU-arvon 26,3, vaikka heidän arkkitehtuurinsa yksinkertaisti Transformer-mallia.

Kieliparissa englanti–ranska tulokset ovat parantuneet vielä kieliparia englanti-saksa enemmän. Bahdanau, Cho ja Bengio (2015) saivat arkkitehtuurillaan BLEU-arvon 28,45, kun taas Vaswanin ym. (2017) arkkitehtuuri nosti tason arvoon 41,0. Kang ym. (2022) puolestaan kertoivat saaneensa arvon 42,2. Tässä välissä RNN-pohjaiset mallit kehittyivät Zhangin, Xiongin ja Sun (2018) myötä tasolle 39,88. You, Sun ja Iyyer (2020) puolestaan saavuttivat arvon 39,1.

Lauseen pituus vaikuttaa käännöksen laatuun tarkkaavaisuusmekanismista ja muusta arkkitehtuurista riippuen. Kangin ym. (2022) arkkitehtuurissa käännösten laatu on paras silloin, kun lauseen pituus on 10–50 sanaa. Lyhyempien ja pidempien lauseiden kohdalla laatu on BLEU-asteikolla 3–5 pistettä alempi. Luongin, Phamin ja Manningin (2015) arkkitehtuurissa käännösten laatu ei heikkene yli 50:n sanan pituisissa lauseissa, mutta toisaalta sen taso on kauttaaltaan Kangin ym. (2022) mallia heikompi. Kangin ym. (2022) arkkitehtuurissa tarkkaavaisuus osaa joka tapauksessa ottaa huomioon kauempana toisistaan olevien sano-

jen yhteyden. Transformer-malli osaa myös huomioida kaukana toisistaan olevien sanojen väliset yhteydet (Vaswani ym. 2017).

Tulee huomata, että BLEU-asteikkoa ei tulisi käyttää absoluuttisena konekääntäjien tason mittarina. Post (2018) huomauttaa, että BLEU-asteikon tulos riippuu sekä sen omista parametreista että konekääntäjien parametreista. BLEU-tuloksia voidaan hyödyntää suuntaa-antavana apuvälineenä, joka osoittaa tulokset kelvollisiksi. Tässä tutkimuksessa esitellyt tutkimukset tulokset ovat melko vertailukelpoisia, koska niissä on käytetty samaa kieliparia, pääasiassa WMT14-opetusaineistoa, suurimmaksi osaksi samaa optimoijaa ja suunnilleen samankokoista sanastoa. Näin ollen tuloksissa olevat erot johtuvat pääosin erilaisista arkkitehtuureista. Esitellyt arvot eivät ole kuitenkaan täysin vertailukelpoisia, mutta niiden avulla voidaan todeta, että käännösten laatu on uudemmilla arkkitehtuureilla selvästi parempi.

Tässä tutkimuksessa keskityttiin kahteen yleiseen kieliparii, joita varten on paljon pohja-aineistoa ja jotka ovat kielitieteellisesti lähellä toisiaan. Tarkkaavaisuusmekanismien kehittyminen on tutkimuksen perusteella parantanut näiden kielten välisiä käännöksiä. Harvinaisten ja pienempiresurssisten kielten osalta kehityksestä on vähemmän tutkimusta, eikä niitä käsitelty tässä tutkielmassa. Lisäksi on huomioitava, että tarkkaavaisuusmekanismit eivät ole ainoa käännösten laatuun vaikuttava seikka. Käsitellyissä arkkitehtuureissa on muitakin eroja.

4.2 Arkkitehtuurien erot ja tehokkuus

Neuroverkkokääntäjissä olevien tarkkaavaisuuskerrosten määrä on kasvanut. Alkuperäisissä tarkkaavaisuusmekanismeja käyttävissä neuroverkkokääntäjissä on yksi tarkkaavaisuuskerros enkooderin ja dekooderin välissä (Bahdanau, Cho ja Bengio (2015), Luong, Pham ja Manning (2015)). Lähes kaikissa tässä tutkimuksessa tarkastelluissa konekääntäjissä on suurempi määrä tarkkaavaisuuskerroksia kuin alkuperäisissä arkkitehtuureissa. Choin, Chon ja Bengion (2018) arkkitehtuurissa kerroksia on niiden tavoin vain yksi. Esimerkiksi Zhangin, Xiongin ja Sun (2018) syvä tarkkaavaisuus on viisikerroksinen. Lisäksi monessa arkkitehtuurissa kerrokset koostuvat alikerroksista, joilla on eri tehtäviä. Esimerkiksi alkuperäisen Transformer-mallin 6-kerroksisessa arkkitehtuurissa on 6 enkooderi- ja dekooderiker-

rosta, joissa on yhteensä 18 tarkkaavaisuuskerrosta (Vaswani ym. 2017). Kangin ym. (2022) mallissa puolestaan on 6 enkooderi- ja 4 dekooderikerrosta, joista jokaisessa dekooderikerroksessa on tarkkaavaisuusalikerros. Tarkkaavaisuuden laajempi hyödyntäminen on tämän perusteella ollut kehityksen suunta. Suurempi tarkkaavaisuuskerroksien määrä mahdollistaa kontekstin tarkemman käsittelyn, kuten Zhang, Xiong ja Su (2018) toivat esille. Tällöin alempien kerrosten tarkkaavaisuus valikoi ylemmille kerroksille annettavaa tietoa. Toisaalta yhä suuremmalla määrällä tarkkaavaisuuskerroksia ei välttämättä enää voida saavuttaa parempia käännöksiä.

Vuosien varrella on kehittynyt erilaisia tarkkaavaisuusmekanismeja, joissa on vaihtelevia painotuksia. Jotkut niistä keskittyvät kontekstin laajempaan huomioimiseen, kun taas osa pyrkii harventamaan tarkasteltavaa syötettä tärkeimpiin sanojen muodostamaan osajoukkoon (Maruf, Martins ja Haffari 2019). Jotkut mekanismit puolestaan pyrkivät ottamaan huomioon sanojen erilaiset tulkinnat (Choi, Cho ja Bengio 2018). Joissakin arkkitehtuureissa erilaisia tarkkaavaisuusmekanismeja yhdistellään, jotta niistä saataisiin mahdollisimman paljon hyötyä. Esimerkiksi Maruf, Martins ja Haffari (2019) käyttävät mallissaan useita erilaisia mekanismeja. Lisäksi osa mekanismeista pyrkii saamaan kaiken mahdollisen tiedon irti konteksista. Kang ym. (2022) esimerkiksi hyödyntävät koko dekooderihistoriaa.

Tarkkaavaisuusmekanismeissa käytetyt funktiot ovat osittain muuttuneet. Moni tarkastelluista tarkkaavaisuusmekanismeista perustuu tarkkaavaisuusfunktioihin 3.1, 3.2 ja 3.8. Jotkut mekanismit käyttävät muunnelmia näistä funktioista. Yleistä on esimerkiksi uusien ulottuvuuksien lisääminen. Toisaalta Kang ym. (2022) käyttävät hyvin erilaista tarkkaavaisuusfunktiota 3.3, ja You, Sun ja Iyyer (2020) puolestaan kovakoodasivat funktiota 3.8.

Edellisistä päätelmistä voidaan huomata, että kehitetyt tarkkaavaisuusmekanismit muuttavat aiempia mekanismeja vaihtelevan laajasti. Osa tarkkaavaisuusmekanismeista muokkaa vanhempia mekanismeja vain pienessä mittakaavassa. Esimerkiksi Choin, Chon ja Bengion (2018) malli on arkkitehtuuriltaan muuten hyvin samanlainen kuin alkuperäiset arkkitehtuurit, mutta siinä jokaiselle annotaatiovektorin alkiolle annetaan oma painokerroin. Se käyttää myös sanaupotusta funktion parametrina. Toisessa ääripäässä on Vaswanin ym. (2017) Transformer-malli, joka käyttää aiemmista malleista täysin poikkeavia tarkkaavaisuusmekanismeja.

Kaikki tarkkaavaisuusmekanismien kehitys ei ole tehnyt niistä monimutkaisempia, laajempia ja hitaampia. Transformer-malli mahdollisti laajemman rinnakkaisten operaatioiden hyödyntämisen. Lisäksi mallissa olevan itseistarkkaavaisuuden käyttäminen pienentää useimmissa tapauksissa laskennallista monimutkaisuutta (Vaswani ym. (2017).). Toisaalta itseistarkkaavaisuus on mahdollista kovakoodata käännösten laatua juurikaan heikentämättä (You, Sun ja Iyyer 2020). Erityisesti Youn, Sunin ja Iyyerin (2020) esittelemä Transformer-mallia yksinkertaistava arkkitehtuuri tekee konekääntäjästä tehokkaamman. Muutosten myötä se käyttää muistia selvästi aiempia malleja vähemmän ja dekoodaus on entistä nopeampaa. Heidän mukaansa käännösten laatu ei heikkene muutosten myötä huomattavasti verrattuna alkuperäiseen Transformer-malliin, mutta käännökset eivät ole yhtä laadukkaita kuin Marufin, Martinsin ja Haffarin mallin (2019). Yksinkertaisempien tarkkaavaisuusmekanismien kehitys on tärkeää, kun tavoitteena on luoda nopeammin toimivia ja tehokkaampia konekääntäjiä. Erityisen hyödyllistä tällainen kehitys on silloin, kun sitä ei tehdä käännösten laadun kustannuksella.

5 Yhteenveto

Tarkkaavaisuusmekanismit ovat olleet oleellisessa osassa neuroverkkokääntäjissä, ja mekanismit olivat alun perin syy sille, että neuroverkkokääntäjistä tuli kilpailukykyisiä. Tarkkaavaisuus pitää huolen siitä, että käännöskonteksti ja aiemmin tuotetut käännökset otetaan huomioon käännöksissä. Tämän tutkielman tarkoituksena oli löytää jotain tarkkaavaisuusmekanismien kehityssuuntia ja tarkastella mekanismien eroja käännösten laadussa, arkkitehtuureissa sekä tehokkuudessa. Tarkasteltujen mekanismien määrä oli rajallinen, mutta niiden myötä esille tuli joitakin kehityksen yleisiä suuntaviivoja. Tutkielman pohjalta voidaan tehdä huomioita siitä, millaisilla tarkkaavaisuusmekanismeilla neuroverkkokääntäjien käännösten laatua ja tehokkuutta voidaan parantaa.

Tutkimuksen perusteella tarkkaavaisuusmekanismit ovat kehittyneet vuodesta 2015, mikä on osaltaan parantanut neuroverkkokääntäjien käännösten laatua ainakin tarkasteltujen kieliparien välillä. Alkuperäinen Transformer-malli nosti käännösten tason korkeammalle, mutta RNN-pohjaiset mallit ovat nyt saavuttaneet ja ohittaneet sen asettaman tason. Kehityksen myötä konteksti pystytään valikoimaan tarkemmin ja kääntäjässä on mahdollista keskittyä myös kohdekielen kontekstiin. Tällaisia ominaisuuksia varten uudemmat tarkkaavaisuusmekanismit hyödyntävät tarkkaavaisuutta yhä laajemmassa mittakaavassa. Tarkkaavaisuutta voidaan esimerkiksi hyödyntää useassa neuroverkon kerroksessa tai useaa erityyppistä tarkkaavaisuutta yhdistellään. Tämä viittaa siihen, että tarkkaavaisuuden laajempi hyödyntäminen on yksi tapa parantaa neuroverkkokääntäjien tasoa. Toisaalta tarkkaavaisuuden kehitys on mahdollistanut entistä tehokkaammat konekääntäjät. Tarkkaavaisuutta voidaan tehostaa esimerkiksi laskuoperaatioita rinnakkaistamalla ja kovakoodaamalla ominaisuuksia.

Tarkkaavaisuusmekanismit ovat edelleen erittäin keskeisessä osassa neuroverkkokääntäjissä, vaikka neuroverkkokääntäjien arkkitehtuuri ja tarkkaavaisuusmekanismit ovat muuttuneet ja kehittyneet. Keskeisyys näkyy esimerkiksi siinä, että sekä DeepL (“How does DeepL work?” 2021) että Google Kääntäjä (Caswell ja Liang 2020) käyttävät tarkkaavaisuusmekanismeja sisältäviä neuroverkkoarkkitehtuureja. Kehitys ja tutkimus tulevat jatkumaan edelleen.

Lähteet

Bahdanau, Dzmitry, Kyunghyun Cho ja Yoshua Bengio. 2015. “Neural machine translation by jointly learning to align and translate”. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings,* https://doi.org/10.48550/arXiv.1409.0473.

Caswell, Isaac, ja Bowen Liang. 2020. “Recent Advances in Google Translate”. Viitattu 3. huhtikuuta 2023. https://ai.googleblog.com/2020/06/recent-advances-in-google-translate.html.

Choi, Heeyoul, Kyunghyun Cho ja Yoshua Bengio. 2018. “Fine-grained attention mechanism for neural machine translation”. *Neurocomputing* 284:171–176. ISSN: 0925-2312. https://doi.org/10.1016/j.neucom.2018.01.007.

Chung, Junyoung, Caglar Gulcehre, Kyunghyun Cho ja Yoshua Bengio. 2014. “Empirical evaluation of gated recurrent neural networks on sequence modeling”. *NIPS 2014 Deep Learning and Representation Learning Workshop,* https://doi.org/10.48550/arXiv.1412.3555.

Hochreiter, Sepp, ja Jürgen Schmidhuber. 1997. “Long short-term memory”. *Neural computation* 9 (8): 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

“How does DeepL work?” 2021. Viitattu 6. maaliskuuta 2023. https://www.deepl.com/en/blog/how-does-deepl-work.

Kang, Liyan, Shaojie He, Mingxuan Wang, Fei Long ja Jinsong Su. 2022. “Bilingual attention based neural machine translation”. *Applied Intelligence* 53 (4): 4302–4315. https://doi.org/10.1007/s10489-022-03563-8.

Koehn, Philipp. 2017. “Neural Machine Translation”, http://mt-class.org/jhu/assets/nmt-book.pdf.

Le, Quoc V., ja Mike Schuster. 2016. “A Neural Network for Machine Translation, at Production Scale”. Viitattu 6. maaliskuuta 2023. https://ai.googleblog.com/2016/09/a-neural-network-for-machine.html.

Luong, Minh-Thang, Hieu Pham ja Christopher D. Manning. 2015. “Effective approaches to attention-based neural machine translation”. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing:1412–1421. https://doi.org/10.48550/arXiv.1508.04025.

Maruf, Sameen, André F. T. Martins ja Gholamreza Haffari. 2019. “Selective attention for context-aware neural machine translation”. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference* 1:3092–3102. https://doi.org/10.48550/arXiv.1903.08788.

Papineni, Kishore, Salim Roukos, Todd Ward ja Wei-Jing Zhu. 2002. “Bleu: a method for automatic evaluation of machine translation”. Teoksessa *Proceedings of the 40th annual meeting of the Association for Computational Linguistics,* 311–318.

Post, Matt. 2018. “A Call for Clarity in Reporting BLEU Scores”. *WMT 2018 - 3rd Conference on Machine Translation, Proceedings of the Conference* 1:186–191. https://doi.org/10.48550/arXiv.1804.08771.

Stasimioti, Maria, Vilelmini Sosoni, Katia Kermanidis ja Despoina Mouratidis. 2020. “Machine Translation Quality: A comparative evaluation of SMT, NMT and tailored-NMT outputs”. Teoksessa *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation,* 441–450. Lisboa, Portugal: European Association for Machine Translation. https://aclanthology.org/2020.eamt-1.47.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser ja Illia Polosukhin. 2017. “Attention is all you need”. *Advances in neural information processing systems* 30. https://doi.org/10.48550/arXiv.1706.03762.

You, Weiqiu, Simeng Sun ja Mohit Iyyer. 2020. “Hard-coded Gaussian attention for neural machine translation”. *Proceedings of the Annual Meeting of the Association for Computational Linguistics,* 7689–7700. https://doi.org/10.48550/arXiv.2005.00742.

Zhang, Biao, Deyi Xiong ja Jinsong Su. 2018. “Neural machine translation with deep attention”. *IEEE transactions on pattern analysis and machine intelligence* 42 (1): 154–163. https://doi.org/10.1109/TPAMI.2018.2876404.