Jukka Holopainen

ERNIE-GEN - luonnollista kieltä tuottava tekoäly

Metropolia Ammattikorkeakoulu

Tieto- ja viestintätekniikan Insinööri (AMK)

Ohjelmistotuotanto

Insinöörityö

29.5.2022

Tiivistelmä

Tekijä:	Jukka Holopainen
Otsikko:	ERNIE-GEN - luonnollista kieltä tuottava tekoäly
Sivumäärä:	28 sivua
Aika:	29.5.2022
Tutkinto:	Insinööri (AMK)
Tutkinto-ohjelma:	Tieto- ja viestintätekniikka
Ammatillinen pääaine:	Ohjelmistotuotanto
Ohjaajat:	Yliopettaja Auvo Häkkinen

Työssä käydään läpi, miten neuroverkot toimivat sekä miten ERNIE-GEN eroaa muista aikaisemmista luonnollisen kielen prosessointiin tarkoitetuista NLP-neuroverkoista. Työ suoritettiin perehtymällä aiheeseen liittyviin julkaisuihin ja sen pohjalta kerrotaan eroavaisuudet.

Työssä käydään läpi, mitä neuroverkot ovat ja kuinka ne toimivat. Lisäksi käydään läpi, kuinka yksinkertainen neuroverkko tehdään. Kerrotaan myös tarkemmin, miten NLP-sovellukset toimivat ja kuinka ERNIE-GEN eroaa tavanomaisista NLP-sovelluksista. Samalla käydään läpi, kuinka seq2seq-kehys toimii ja kuinka merkittävä se on NLP-sovelluksien kanssa.

ERNIE-GEN on kiinalaisen Baidun julkaisema, valmiiksi koulutettu ja hienosäädetty NLP-neuroverkon kehys. ERNIE-GEN saavuttaa luonnillisen kielen prosessoinnissa edeltäjiinsä verrattuna paremman lopullisen tarkkuuden. Insinöörityössä käydään läpi myös tekoälyjen, neuroverkkojen ja ERNIE-GEN:n mahdollisia heikkouksia.

Työ tehtiin pääasiallisesti tutkimalla julkaisuja, joissa kerrotaan, kuinka ERNIE-GEN eroaa muista vertaisistaan NLP-sovelluksistaan. Työssä kuvataan myös, kuinka ERNIE-GEN saadaan toimimaan sovelluksissa. Samalla käydään läpi, millä kielillä ERNIE-GEN:ä on saatavilla valmiiksi koulutettuna versiona.

Työn perusteella huomataan, että ERNIE-GEN on helppo saada käyttöön erilaisissa sovelluksissa. Samalla tuotiin ERNIE-GEN:ä ja muita NLP-neuroverkkoja selvemmin Metropolian tietoisuuteen.

ERNIE-GEN on työn perusteella tavallista parempi NLP-neuroverkko, jonka avulla voidaan saada tietyissä tilanteissa parempaa tekstiä kuin mitä maallikko voi kirjoittaa aiheesta.

Avainsanat: neuroverkko, NLP, ERNIE

Abstract

Author:	Jukka Holopainen
Title:	ERNIE-GEN Natural Language Producing Artificial Intelligence
Number of Pages:	28 pages
Date:	29 May 2022
Degree:	Bachelor of Engineering
Degree Programme:	Information and Communication Technology
Professional Major:	Software Engineering
Supervisors:	Auvo Häkkinen, Principal Lecturer

The thesis examines how neural networks work and how ERNIE-GEN differs from other previous NLP neural networks for natural language processing. The study was carried out by reading the related publications and the differences are explained based on the relevant literature.

The paper goes through what neural networks are and how they work. In addition, it explains how to make a simple neural network. Also, how NLP applications work and how ERNIE-GEN differs from conventional NLP applications is explained in more detail, as well as. How the seq2seq framework works and how it is significant as to NLP applications.

ERNIE-GEN is a pre-trained and fine-tuned NLP neural network framework published by Chinese Baidu. ERNIE-GEN achieves better final accuracy in natural language processing compared to its predecessors. The possible weaknesses of artificial intelligence, neural networks and ERNIE-GEN are also reviewed in the paper.

The study was done primarily by examining publications that explain how ERNIE-GEN differs from its other peer-to-peer NLP applications. The paper also describes how to make ER-NIE-GEN work in applications.

The study indicates that ERNIE-GEN is easy to deploy in a variety of applications, i.e.

ERNIE-GEN is a better-than-usual NLP neural network that can provide better than average text in certain situations.

Keywords: Neuralnetwork, NLP, ERNIE

Sisällys

Lyhenteet

GLUE *General Language Understanding Evaluation.* Suorituskykytesti NLP-neuroverkoille. SuperGLUE on siitä seuraava tehty malli.

LSTM *Long Short Term Memory.* Neuroverkkoja, jotka kykenevät oppimaan sanojen riippuvuuden tekstissä. Mikä on yksi NLP-sovelluksien ennustusongelmana. Tätä vaaditaan monimutkaisissa NLP-sovelluksissa kuten koneen käännöksessä.

NLP *Natural Language Processing*. Kuvaa tekoälyjä, joiden tarkoitus on tuottaa sovelluksia, joissa käytetään kirjoitettua tekstiä.

1 Johdanto

ERNIE-GEN on tekoälyohjelma, joka tuottaa englannin- ja kiinankielistä tekstiä neuroverkon koulutusvaiheessa saamansa materiaalin mukaisesti. Jos neuroverkolle annetaan viestiketjuja, niin ERNIE-GEN oppii itse tuottamaa viestiketjuja. Työssä kuvataan ERNIE-GEN-rajapintaa sekä sitä, kuinka se toimii ja miten siitä voi tehdä sovelluksen.

Insinöörityö kuvaa ensimmäiseksi, kuinka neuroverkkoon perustuva tekoäly toimii, ja kertoo, kuinka sellainen tehdään ja mitkä ovat sen heikkoudet. Tekoälyt eivät kuulu Metropolian tieto- ja viestintätekniikan insinöörin pakollisiin opintojaksoihin. Siksi käydään läpi, miten ne toimivat. ERNIE-GEN on valmiiksi koulutettu erikoinen neuroverkkotekoäly.

Myöhemmin työ kertoo ENIE-GEN:stä tarkemmin ja kuvaa, kuinka ERNIE-GEN-rajapinnasta tehdään sovellus ja miten sitä käytetään. Samalla kuvataan, mitä työvälineitä sen käyttöön on annettu ja sitä, kuinka rajapinnalla voi tehdä erilaisia sovelluksia. Kerrotaan myös, kuinka paljon se vie koneesta tehoja sovelluksia varten.

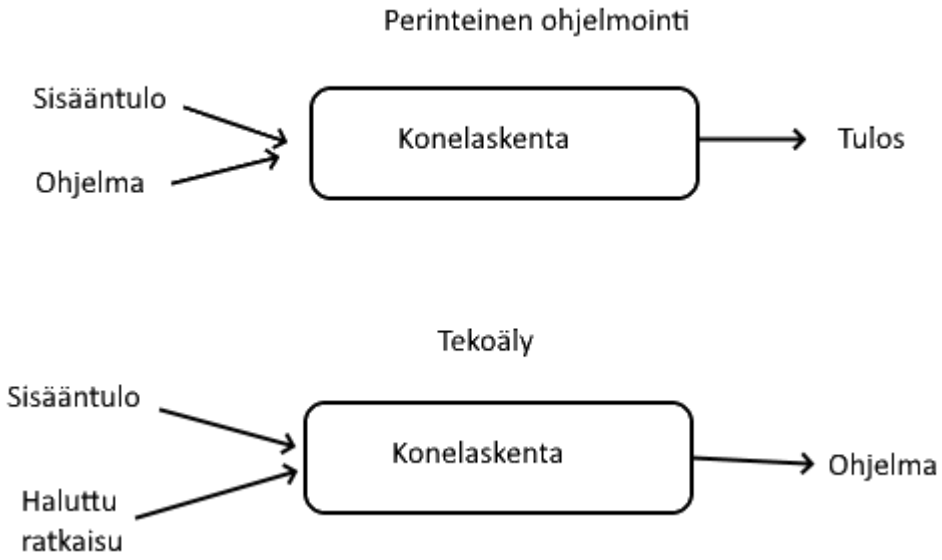
Insinöörityön tarkoitus on kertoa, mikä on ERNIE-GEN ja kuinka se on parantanut NLP-tekoälyjä. Työssä myös kerrotaan, miten ERNIE-GEN voidaan ottaa käyttöön ja kuinka sillä voi tehdä töitä. Insinöörityön alussa kerrotaan, kuinka tekoälyt toimivat, mikä auttaa ymmärtämään, miten ERNIE-GEN toimii.

Työssä käydään myös läpi, kuinka luonnollisen kielen tuottavat tekoälyt tehdään. Tämä tehdään sen takia, jotta lukijat ymmärtävät helpommin, miten ERNIE-GEN toimii eri lailla. Tätä käydään läpi toisessa luvussa, jonka lopussa käydään läpi, miten kieltä tuottavat tekoälyt tehdään ja miten ne ovat tehneet.

Toisen luvun lopussa käydään läpi seq2seq-malli, koska ERNIE-GEN on rakennettu sen pohjalta. Tämän pitäisi auttaa ymmärtämään, miten ERNIE-GEN toimii.

2 Neuroverkkoon perustuva tekoäly

Tekoälyt ovat tietokoneohjelmia, joiden tarkoitus on saada haluttu lopputulos datasta. Varhaiset tekoälyt olivat yleisesti ottaen suuria valintapuita. Tämä tehtiin sen takia, koska niitä on ihmisen kaikista helpointa tehdä. Oppivat tekoälyt tekevät oman valintapuunsa sisääntulodatan ja annettujen vastauksien välillä. Tämä on suurin eroavaisuus tekoälyohjelman ja tavallisen ohjelman mukaan. Nimittäin tavalliselle ohjelmalle tehdään runko, minkä mukaan ohjelma tehdään. Kun taas tekoälyohjelma tekee oman ohjelman ratkaistakseen ongelman, jonka sen tekijä on sille antanut. Kuva 1 auttaa hiekan paremmin ymmärtämään, miten perinteinen ohjelmointi ja tekoäly eroavat toisistaan.



Kuva 1. Periaate miten tekoäly eroaa perinteisestä ohjelmoinnista.

Perinteisessä ohjelmointityylissä tehdään koneelle ohjelma. Tietokone ottaa ohjelman ja sisääntulon sekä tekee laskelmia, josta saadaan tulos. Tekoälyssä annetaan koneen itse rakentaa ohjelma halutun ratkaisun ja sisääntulon avulla.

Tämän takia sanotaankin, ettei kukaan tiedä, miten erilaiset algoritmit toimivat, koska ne on tehty käyttäen tekoälyn periaatteita. Jotta ihminen voi tehdä ohjel-

man, pitää henkilön osata kyseinen asia hyvin ja tehdä sen perusteella ohjelmoidessaan päätelmiä. Kun taas tekoäly voi käydä läpi isoja määriä dataa nopeammin kuin ihminen, ja datan perusteella se voi tehdä päätelmiä tulevaisuudesta halutulla tavalla. Yksi suosituimmista tavoista rakentaa tekoälyä on tehdä siitä neuroverkko.

Neuroverkkoon perustuvat tekoälyt matkivat ihmisen aivojen rakennetta. Mutta tekoäly ei yritä kopioida samaa rakennetta, koska tällä hetkellä aivojen toimintaa on liian monimutkaista matkia ohjelmakoodissa. Sen takia matkitaan niiden toimintaa hyvin selkeällä tavalla. Aivot koostuvat hermosoluista, jotka ottavat viestejä viereisistä soluista ja lähettävät signaaleita eteenpäin. Näitä hermosoluja kutsutaan englanniksi termillä neural, josta tulee neuroverkon nimi. Neuroverkko koostuu neuroneista ja niiden välisistä yhteyksistä. Koodillisesti neuroverkon matkiminen tapahtuu siten, että neuroverkolle annetaan jokin datan arvo bitteinä ja neuroverkko painottaa sen arvoja jokaisen neuronin kohdalla. Neuroverkko tuottaa sille syötetyn alkudatan ja sen oikean vastauksien pohjalta neuroverkon, jonka tarkoitus on saada uudesta datasta haluttu vastaus. Esimerkiksi voidaan saada tietoon, onko kyseisessä röntgenkuvassa keuhkokuume.

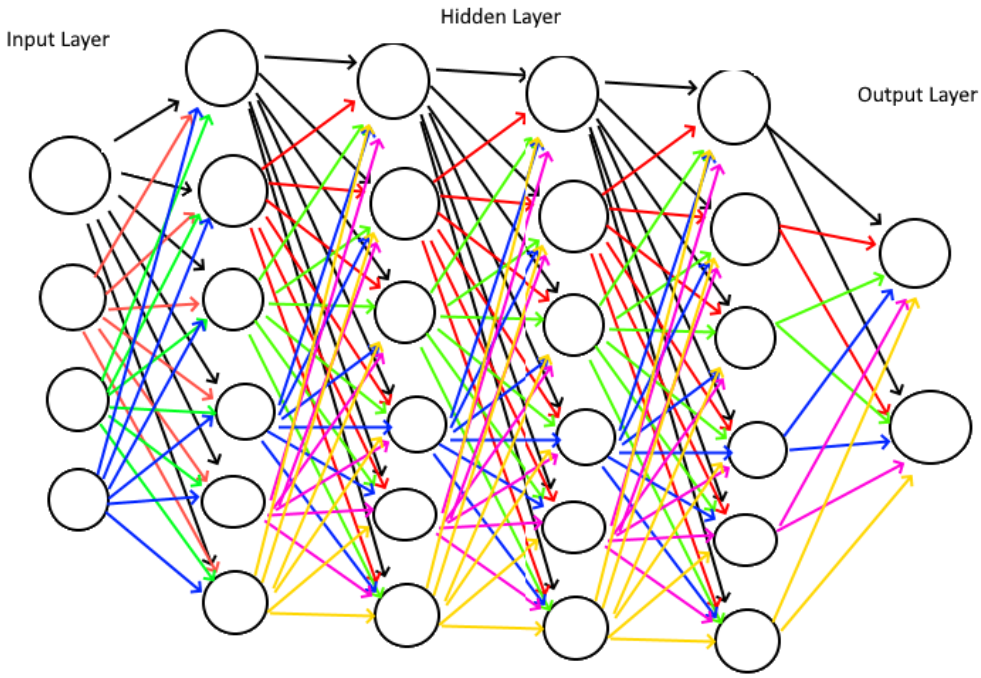
Neuroverkon rakenteen alku ja loppu ovat merkittäviä, ja ne pitää saada oikein, jotta neuroverkko toimii halutulla tavalla. Neuroverkolle tutkittavaksi annetun sisääntulodatan pitää olla koulutukseen käytetyn datarivin mukainen, jotta saadaan jokainen datan merkittävä tekijä vaikuttamaan lopputulokseen. Tämän voidaan tehdä muokkaamalla itse dataa tai rakentamalla neuroverkon hyväksymää sisääntuloa datan mukaiseksi. Neuroverkon loppu pitää tehdä sellaiseksi, että se sisältää niin monta neuronia kuin on mahdollisia tulosvaihtoehtoja. Tämä tehdään sen takia, jotta neuroverkko voi kertoa, mikä on vastaus kyseiseen dataan.

Neuroverkko muuttaa sisääntulon muokkaamalla sen arvoa jokaisella neuronilla. Neuroneiden sisäiset painot yhteyksien välillä muuttavat sisääntulon arvoa toiseksi. Lopuksi, kun sisääntulodata on mennyt neuroverkon viimeisestä neuro-

nista läpi, saadaan neuroverkosta ulostulo. Neuroverkko koostuu useista neuronien riveistä, joista ensimmäistä riviä kutsutaan sisääntulokerrokseksi (engl. input layer). Seuraavat rivit muodostavat piilotettukerroksen (engl. hidden layer) ja viimeinen rivi on ulostulokerros (engl. output layer).

Sisääntulokerros riippuu syötettävästä datasta ja sen osista, sillä sen perusteella se muodostuu. Piilotetun kerroksen teko on paljon vapaampaa. Tosin yleensä se on muutaman neuronia isompi per rivi kuin sisääntulokerros ja piilotetussa kerroksessa on yleensä useita rivejä peräkkäin. Tässä kerroksessa dataa muokataan, jotta siitä saadaan vastaus. Ulostulokerros riippuu kokonaan siitä, mitä neuroverkolla halutaan saada aikaan kyseisestä datasta. Jos kyseessä on vaihtoehtoinen lopputulos, ulostulokerros on lopputuloksen määrän kokoinen. Jos taas halutaan saada yksi arvo, niin yksi neuroni ulostulokerroksessa riittää. Sisääntulo- ja ulostulokerros koostuvat vain yhdestä rivistä, kun taas piilotetulla kerroksella on yleensä useampia rivejä.

Perusrakenteeltaan neuroverkot koostuvat neuroneista. Neuronit jaetaan riveihin ja neuroneilla on yhteys edellisen ja seuraavan rivin neuronin kanssa. Matemaattisesti neuroverkko toimii siten, että kyseisen rivin kaikilla neuroneilla on yhteys seuraavan rivin jokaiseen neuroniin. Samalla jokaisella yhteydellä on erilaiset painot. Tämän jälkeen tulos summataan, josta tulee uuden rivin yhden neuronin arvo. Tämä toistetaan kaikille rivin neuroneille, jolloin saadaan aikaiseksi uusi tulosrivi. Kuvassa 2 on esitetty yksinkertaisen ja lyhyen neuroverkon rakenne.



Kuva 2. Neuroverkon yhteydet ja rakenne.

Neuroverkot ovat hyödyllisiä tekoälyä tehdessä, koska ne matkivat luonnollista rakennetta, jonka takia tekoäly toimii paremmin. Mallin avulla saadaan työkalu, jonka avulla saadaan tehtyä paljon erilaisiin tarkoitukseen tekoälyjä.

Tässä luvussa kerrotaan, miten yksinkertainen neuroverkko tehdään, mikä on niiden rakenne ja miten tavallinen luonnollisen kielen tuottavia neuroverkkojen sovelluksia on. ERNIE-GEN:stä kerrotaan vasta seuraavassa luvussa.

2.1 Neuroverkkoon perustuvien tekoälyjen jaottelua

Neuroverkkoja jaotellaan erilaisiin arkkityyppeihin. Tämä tehdään sen takia, jotta tavallisemmat ongelmat olisivat helpompia ratkaista ilman, että pitää keksiä pyörää uudestaan. Neuroverkkoja jaotellaan pääsääntöisesti niiden rakenteen mukaan. Neuroverkon rakenne yleensä on huomattavasti erilainen silloin, kun neuroverkkoa käytetään vastaamaan erilaiseen kysymykseen.

Seuraavaksi käydään läpi neuroverkkojen tyyppejä ja kuvataan, kuinka ne eroavat toisistaan. Lisäksi kerrotaan myös, millaisissa sovelluksissa näitä neuroverkkoja käytetään.

Feed-forward-neuroverkko on yksinkertaisin keinotekoinen neuroverkko. Nimittäin data kulkee vain yhteen suuntaan. Nämä neuroverkot ovat käytettäessä nopeita, mutta niiden kouluttaminen kestää huomattavasti pidempään verrattuna muihin neuroverkoihin ja sen käyttöön. Lähestulkoon kaikki kuvan- ja puhetunnistussovellukset käyttävät näitä neuroverkkoja. [1.]

Radial basic function -neuroverkko on neuroverkko, jota käytetään lajittelemaan datan jäseniä niiden matkasta keskipisteessä. Tätä voi käyttää silloinkin, kun ei ole dataa koulutusta varten, mutta halutaan silti jakaa dataa eri ryhmiin. Luvuissa 2.2 ja 2.3 käydään tarkemmin läpi, kuinka neuroverkkoa koulutetaan. Erilaiset luokittelusovellukset käyttävät näitä.

Kohonen self-organizing -neuroverkko tekee sattumanvaraisista sisääntulodatasta vektoreita kuvaamaan discrate-kartan, joka koostuu neuroneista. Näitä käytetään esimerkiksi, kun yritetään selvittää datasta selviä yhtenäisyyksiä esimerkiksi lääketieteessä.

Recurrent-neuroverkko tallentaa piilotetun tason tuottaman tuloksen, jota käytetään tulevissa ennustuksissa. Ulostulo tulee olemaan osa tulevaa sisääntuloa. Esimerkiksi konelukeminen eli tietokoneen tekstin ääneen lukeminen käyttää tätä.

Convolution-neuroverkoissa tulo tulee palasissa, kun ne tulevat filtteristä läpi. Filtteri on kuvan muokkaamassa käytetty nimi, jonka tarkoitus on muuttaa koko kuvan halutulla tavalla, esimerkiksi vaihtamalla värit päinvastaiseksi. Tämä auttaa verkostoa muistamaan kuvan osissa. Tätä käytetään esimerkiksi kuvankäsittelyssä.

Viimeisenä on modular-neuroverkko. Siinä käytetään useita neuroverkkoja työskentelemään yhdessä, jotta saadaan ulostulo. Tämä on uusi kokeiluversio, joka on vielä tutkimuksessa.

Kannattaa muistaa, miten näitä erilaisia neuroverkkoja käytetään. Jos tulee olemaan tekemisissä näiden kanssa, neuroverkon toimintatapa auttaa ymmärtämään, miten sovellukset toimivat. Jokaisella neuroverkolla on vielä omat heikkoutensa ja niistä ei mikään voi tehdä kaikkea.

ERNIE-GEN käyttää enimmäkseen Recurrent-neuroverkkoa. ERNIE-GEN on tarkempi kuin edeltäjänsä, sillä piilotetussa tasosta ei poisteta niin paljon tietoa, kun ERNIE-GEN:ia koulutetaan. Tämä hidastaa jonkin verran koulutuksen tehtävien tekoa, mutta mahdollistaa tarkemman neuroverkon luomisen.

Seuraavaksi käydään läpi, miten yleisimmät neuroverkkoon perustuvat tekoälyt tehdään ja mitä ongelmia ne voisi ratkaista. Sitten katsotaan tarkemmin NLG-tekoälyä (natural language generation), sillä ERNIE-GEN on sitä luokkaa oleva tekoäly. NLG-tekoälyt toteutetaan välillä käyttäen erilaisia neuroverkkopohjia, sillä se antaa paremman tuloksen tiettyjen tehtävien teossa.

Tekoälyjä jaotellaan eri tyypeiksi, jotta voidaan valita parhaiten tehtävään sopiva neuroverkko ja ratkaista ongelmia tehokkaasti. Eri neuroverkkotyyppejä tarvitaan, koska ne käyttävät dataa eri muodossa. Tietyt datatyypit joudutaan esivalmistelemaan neuroverkolle eri tavoilla, jotta neuroverkko ymmärtää sitä. Joillakin datatyypeillä on myös huomattavan paljon helpompi tehdä lisää testidataa muokkaamalla jo olemassa olevaa dataa. Esimerkiksi kuvia voidaan helposti muokata, jotta se on neuroverkolle uusi ja tuntematon kuva.

Neuroverkkoa tehdessä voidaan myös jaotella hieman neuroverkoja eri tyypeiksi sillä perusteella, miten halutaan saada lopputulos. Esimerkiksi jos kyseessä on luokitteluongelma, niin neuroverkon loppuneuronien määrä on vaihtoehtojen määrä. Mutta jos ongelmana on skaalattava arvo kuten hinta, niin loppuun laitetaan vain yksi neuroni, jonka arvo on kyseinen skaalattava arvo.

2.2 Feed forward -neuroverkon toiminta

Kuten edellisessä luvussa kerrottiin, feed forward -neuroverkko on rakenteeltaan yksinkertaisin neuroverkko. Sen avulla on helpointa ymmärtää, kuinka neuroverkot yleisestikin toimivat. Vaikka tämä onkin yksinkertaisin neuroverkko, se ei tarkoita, että näillä ei ole erilaisia toteutusmalleja sen perusteella, mikä on neuroverkon tehtävänä. Mutta rakenteellisesti nämä neuroverkot ovat hyvin samankaltaisia keskenään. Eroavaisuudet tulevat pääasiallisesti vasta rakenteen lopussa. Neuroverkon rakenteen alkuosa on muutenkin täysin riippuvainen sen datan muodosta, jota halutaan käyttää neuroverkossa.

Kun koodillisesti aletaan tekemään neuroverkkoa, tehdään ensin neuroverkolle runko, jota ei voi muuttaa kokoamisen jälkeen. Runko rakennetaan sillä periaatteella, että alku näyttää koulutusdatan rivin mukaiselta. Tästä tulee neuroverkon sisääntulotaso. Neuroverkon loppu mukailee vastauksen muotoa, esimerkiksi jos halutaan lopputulokseksi oikein tai väärin, tulee loppuun laittaa kaksi neuronia. Mutta jos halutaan esimerkiksi juokseva arvio, yksi neuroni riittää.

Sisääntulo- ja ulostulokerroksen välissä oleva neuroverkko on paljon vapaampi muokata. Tosin perinteisesti piilotettukerrokselle rakennetaan neuroneiden määrä siten, että ne ovat enemmän kuin mikä on datan tuloneuronien määrä. Tämä on sen takia, että datan ei tarvitse mennä pienempään muotoon, kun se etenee neuroverkossa. Tämän takia piilotettukerroksessa yleensä rivin neuronien määrä pysyy samana. Neuronien määrä ei ole tarve vähentää, sillä datan monipuolisuus yksinkertaistuu. Samasta syystä ei ole tarvetta lisätä neuronien määrää myöhemmin. Joten lyhykäisyydessä neuroverkon keskiosa kannattaa pitää samankokoisena.

Neuroverkon piilotettukerroksessa tehdään yleensä ainakin parikymmenen rivin mittaiseksi. Mitään sääntöä, kuinka pitkä piilotettukerros pitäisi olla, ei ole, ja yleensä kun neuroverkkoa tehdään, testataan useammalla piilotettukerroksen pituuden arvoilla. Tämä tehdään sen takia, jotta voidaan valita paras mahdollinen neuroverkko kyseiseen tehtävään.

Neuroverkon koulutus tehdään yhdessä iteraatiossa. Nämä jaetaan epookkeihin (engl. epoch), jotka ovat osa kokonaista datajoukkoa. Tarkoituksena on, että epokin aikana käytetään datasetin arvoa vain kerran. Epookit koostuvat dataeristä ja niiden koko määrittää, kuinka monta dataerää käydään jokaisessa epookissa. Yleensä käydään muutama epookki, kun neuroverkkoa koulutetaan. Sen avulla neuroverkon ennustaminen tulee tarkemmaksi. Mutta liian monta epookkia ei kannatta tehdä, koska neuroverkosta voi tulla liian ylisovitettu. Ylisovitetulla tarkoitetaan neuroverkkoa, joka on sovitettu liian paljon koulutusdatan mukaisesti. Liiallinen sovitus heikentää lopullista neuroverkon tarkkuutta. Myös koulutukseen kuluva aika ja tietokoneen käyttämät tehot olisivat täysin turhaa kyseisessä tilanteessa. [2.]

Jokaiselle datasetin arvolle tehdään neuroverkossa kaksi operaatiota. Ensimmäisessä data kulkee neuroverkon läpi, jonka jälkeen lasketaan datan oikeasta tuloksesta häviöarvo. Tätä arvoa käytetään, kun lasketaan uudelleen neuroverkon neuroneiden väliset painot. Painot lasketaan lopusta alkuun alla olevan kaavan mukaan. [3.]

$$Z = \text{Bias} + W_1X_1 + W_2X_2 + ... + W_nX_n$$

Siinä Z on seuraavan neuronin arvo, W on edellisen rivin neuronin paino ja X on sen arvo. Bias on W_0 arvo, joka on ominaista kyseiselle neuronille. Kyseinen arvo voi hyvin olla 0. Jokaisella neuronilla on omat W- ja X-arvot.

Koska neuroneiden välisten painojen määrä, jopa pienillä neuroverkoilla, nousee hyvin helposti miljoonien kombinaation luokkaan, niin painoja säädellään eri funktioilla. Tähän tarvitaan virheen laskemista lopputuloksen ja oikean arvon välillä. Neuronien välinen tuleva paino lasketaan kaltevuuden avulla (Gradient descent). Tämän avulla lasketaan pari seuraavaa mahdollista arvoa, josta valitaan parempi arvo, jolla virhe pienenee. Neuroverkossa käytetään yleensä stokastista kaltevuuden laskufunktiota. Siinä on hieman sattumanvaraisuutta, joka mahdollistaa paremman neuroverkon teon. Neuroverkot käyttävät näitä funktiota täysin automaattisesti läpi niiden koulutuksen aikana. [4.]

Sattumalla on hivenen osaa neuroverkon teossa. Nimittäin hetkellisesti voidaan päästä samaan lopputulokseen muutamilla eri tavoilla. Tämän takia, kun neuroverkoja tehdään, niitä tehdään kerralla useampi ja valitaan validointidatan avulla paras neuroverkko kyseiseen työhön. Tosin vaikka ne vaihtelevat, niin saman arkkitehtuurin neuroverkot ovat ominaisuudelta huomattavasti samankaltaisia toisin kuin arkkitehtuurilta erilaiset neuroverkot. Kun neuroverkoja rakennetaan, testataan yleensä muutamaa arkkitehtuurillisesti eroavaa neuroverkkoa, ja valitaan neuroverkko, jonka tarkkuus on paras.

Neuroverkon lopullista tarkkuutta lasketaan validointidatan avulla. Validointidataa yleensä tehdään käyttämällä 10 % - 20 % testidataa. Tämä validointidatan eristäminen tehdään ensimmäisenä. Validointidataa ei saa käyttää testidatana, sillä neuroverkosta voi tulla liian optimoitu, ja se voi kulkea datan sisäisen kohinan mukaan. Tämän takia kannattaa antaa neuroverkoille koulutusvaiheessa ja vaihtelevaa dataa.

2.3 Neuroverkon muodostaminen

Tässä luvussa käydään tarkemmin läpi, miten neuroverkkoa rakennetaan. Vaikka neuroverkoja on erilaisia, niillä on tiettyjä samankaltaisuuksia niiden rakenteessa. Ensimmäiseksi pitää valita, mitä ongelmaa halutaan ratkaista ja missä muodossa data on. Sen jälkeen valitaan, mikä edellä esitellyistä neuroverkoista parhaiten sopii ratkomaan kyseistä ongelmaa.

Tämän jälkeen neuroverkolle rakennetaan runko, jota ei muokata enää myöhemmin. Neuroverkoille rakennetaan kaikki neuronit ja niiden väliset yhteydet. Tämän jälkeen esivalmistellaan kaikki koulutusdata siihen muotoon, miten se halutaan esitettävän neuroverkolle. Yleisesti neuroverkon alku tehdään datan rakenteen mukaiseksi, jotta dataa ei tarvitse muokata neuroverkon käyttämiseen. Tavallisesti kun tehdään testidataa, siitä otetaan noin 10 % talteen validointia varten. Neuroverkkoa lopullista tarkkuutta arvioidaan validointidatalla. Tällä validoidaan lopullista annettua tarkkuuta. Loput datasta käytetään koulutusdatana.

Kun neuroverkkoa rakennetaan, eniten aikaa vievää on saada koulutusdata haluttuun muotoon, ja korjata mahdolliset puutokset ja virheet datasta. Data on harvoin siinä muodossa, jota voisi suoraan käyttää neuroverkossa.

Tietyillä datoilla voidaan hankia enemmän dataa muokkaamalla olemassa olevaa dataa. Yleisimmin muokataan kuvadataa. Muokkaamisella on tarkoituksena saada suurempaa määrää dataa koulutustarpeisiin. Kuvadataa taas voidaan helposti muokata siten, että tuleva kuva on koneelle täysin tunnistamaton toisesta kuvasta.

2.4 Neuroverkon vahvuudet ja heikkoudet

Tässä luvussa käydään läpi neuroverkon vahvuuksia ja heikkouksia. Vaikka neuroverkot voivat ratkaista monia ongelmia, niiden käyttämisessä pitää ottaa huomioon myös niiden heikkouksia.

Yksi suurimmista vahvuuksia neuroverkoissa on niiden kyky ratkaista ongelmia, jotka eivät ole lineaarisia. Esimerkiksi valon nopeuden määrittäminen on lineaarinen ongelma. Se on yksinkertaisempi ja nopeampi ratkaista kyseisen ongelman Newtonin toista lakia käyttäen, kuin tehdä sitä varten oma neuroverkko ratkaisemaan ongelma. Tämäkin mahdollistaa niiden käytön, kun ei tiedetä missä muodossa kysymys on. [4;5.]

Seuraava neuroverkkojen vahvuus on mahdollisuus rakentaa vastauksia ennalta tuntemattomien funktioiden teossa. Nimittäin jotkut funktiot käyttäytyvät hyvin erikoisesti verrattuna siihen, miten esimerkiksi lineaariset funktiot käyttäytyvät. Neuroverkot voivat niiden toimintansa avulla rakentaa toimivan kuvauksen niille tehdyn kyseiseen ongelman ratkaisuun. Tosin näitä tehdessä neuroverkosta voi helposti tulla epätarkka, joten parasta on tehdä useita neuroverkkoja, joita kokeillaan, jotta saataisiin paras lopputulos.

Neuroverkkoja käyttävillä tekoälyillä on tiettyjä rajoituksia ja heikkouksia, mitkä kannattaa pitää mielessä. Esimerkiksi jos tekoäly saa liian pienen ja yksipuolisen koulutusdatasetin, tekoälystä voi tulla aika hyödytön realistisessa käytössä. Tekoäly voi analysoida dataa liian tarkkaan ja ottaa datassa olevan kohinan ilmiönä oikeasta maailmasta. Sama voi tapahtua, jos data on yksipuolista eikä ole riittävän kattavaa.

Tekoälysovellukset, joita tehdään, ovat luonteeltaan hyvin erikoistuneita. Se tarkoittaa, että kannattaa olla varovainen, mitä johtopäätöksiä tehdään tuloksista. Esimerkiksi kuvista tehdyt analyysit päätyvät usein niin, että kyseinen asia, kuten banaani, on selvästi erilainen kuin oikean elämän banaani. Lyhyesti tekoälyt ovat erikoistuneita ongelman ratkaisijoita.

Koska tekoäly on erittäin erikoistunut, pitää olla tarkka, että sitä käytetään juuri sille tarkoitettuihin asioihin. Tällaista apuvälinettä on myös helppo käyttää väärin. Esimerkiksi jos se rakennetaan tuottamaan vääränlaista ulostuloa, niin saadusta lopputuloksesta ei ole realistisessa maailmassa mitään hyötyä.

Jotkut neuroverkon tyypit voivat ratkaista ongelmia, mitä toiset neuroverkot eivät kykene ratkaisemaan. Tämän vuoksi pitää olla hyvin tarkka tai kokeilla erilaista neuroverkkoa. Muuten neuroverkkotekoäly ei ratkaise ongelmaa, jota se rakennettiin ratkaisemaan.

Koska ratkaisut voivat olla erittäin erikoistuneita, ne toimivat ainoastaan asioissa, joihin ne on tehty. Niitä ei myöskään kannata kouluttaa uudestaan vaan uuteen tarkoitukseen tulee tehdä uusi tekoäly. Pitää myös muistaa käyttää arkijärkeä tulosten tulkinnassa, kun tekoäly esittää jotain aivan outoa. Esimerkiksi ne voivat olettaa jonkun taudin olevan lievempi, vaikka se on vakavampi tauti. Tämä tehdään sen takia, koska vakavammassa taudissa lähdetään aikaisemmin sairaalahoitoon. Sen takia kuolleisuus voi tippua vakavasta muodosta enemmän, jonka takia tekoäly, joka katsoo ainoastaan tilastoa, päätyy lopputulokseen, että vakava versio on lievempi.

Neuroverkkojen yksi ongelma on se, että ne käyttävät koulutusvaiheessa paljon laskentatehoa. Kun tulokseksi saatua neuroverkkoa käytetään, ne ovat nopeita, eivätkä ne enää tarvitse paljon laskentatehoa. Neuroverkot eivät myöskään skaalaannu kovinkaan hyvin, kun koulutusdatan ja tai niiden neuronien määrä kasvaa. Näissä tapauksissa neuroverkkojen kouluttaminen kestää yleensä useita päiviä.

3 Luonnollisen kielen tuottavat tekoälyt

Tässä luvussa käydään läpi, mitä luonnollisia kieltä tuottavia tekoälyjä on, ja tarkemmin, mikä seq2seq-työkalu on. Luonnollisen kieltä tuottavat eli NLP-tekoälyt ovat sovelluksia, jotka työskentelevät tekstien kanssa.

Esimerkiksi konekääntäjät ja tekstiluku ovat NLP-sovelluksia. NLP-sovelluksia ovat yksinkertaisesti sovellukset, joissa käytetään tekstiä tai käytetään puhetta, kuten esimerkiksi tekstipuhesovellusta. Ennen vuotta 2015 suurin osa NLP-sovelluksista tehtiin statistiikan avulla. Mutta sen jälkeen on alettu käyttää neuroverkkoja ja muita tekoälyjä käyttäviä sovelluksia ratkomaan NLP-tehtäviä. [6.]

Tällaisia tekoälyjä ovat muun muassa tekstiluku ja puheen tunnistaminen. Myös keskustelun etenemisen analysoinnissa käytetään NLP-tekoälyjä. Vaativimmat tekoälyt koneen prosessoinnin kannalta ovat kääntäjät, kirjan tekijät ja konekääntäjät. Vaikka konekääntäjiä ja muuten kirjoitusta tuottavia tekoälyjä on nykyisin tehty, ei pidä vielä pelätä, että kone vielä kykenee esimerkiksi käännöshommissa selviämään erityisen hyvin. Kone ei ymmärrä vielä kaikkia pieniä hienouksia teksteissä, vaan se pystyy esimerkiksi vain kääntämään sanasta sanaan. Esimerkiksi tavanomaiset lauseet tietyssä ympäristössä kuulostavat erilaisilta.

Tavallisesti tekstiä analysoitaessa NLP-tekoäly vaihtaa tekstin sanat joihinkin arvoihin. Esimerkiksi sana vaihdetaan numeeriseksi arvoksi 1. Tämä mahdollistaa sen, että kone voi käydä helpommin tekstejä läpi. Tämän avulla voidaan myös saada selville, mitä sanoja käytetään tiettyä tarkoitusta varten.

Tähän mennessä tekoälyt eivät ole onnistuneet tuottamaan tai tulkitsemaan tekstejä erityisen hyvin. NLP-tekoälyt katsovat yleisesti vain tiettyjä sanoja, koska se on helpompi ja nopeampi tehdä kuin sanojen merkityksen analysointi. Välillä eri chatbotit haluavat, että tietyt tiedot pitää antaa tietyssä muodossa.

NLP on saanut suosiota 2010-luvulla, kun teknologia sai uusia kehityksiä silloin. Huomattiin, että uudet, tehokkaammat laitteet voivat käydä läpi esimerkiksi erilaisia määriä lääketieteellisiä tekstejä. Tekoälyn avulla saadaan paremmin tietoon, miten tiettyjä sairauksia voidaan hoitaa. Nämä löydökset ovat mahdollisia suuren datamäärän avulla, johon ilman tekoälyä pitäisi upottaa hyvin suuri määrä työtunteja.

NLP-tekoälyjä on monenlaisiin sovelluksiin. Myös niiden kehitys on ollut nopeaa. Muutama vuosi sitten ei ollut vielä mahdollista tehdä esimerkiksi tekoälyä, joka tuottaa uutta kirjoitettua tekstiä, joka on halutun tyylin mukaista.

Se mitä ERNIE-GEN tarjoaa, on tällä hetkellä vaativinta, mitä NLP-tekoäly voi tuottaa. Nimittäin se on tarkempi kuin veroisensa tekoälyt, joka voi auttaa esimerkiksi saamaan konekäännökset paljon tarkemmiksi.

3.1 Seq2Seq-malli

Seq2seq-malli on encoder-decoder kehys Tensorflow’iin, jota voi käyttää esimerkiksi käännöstöihin ja tekstin tutkimiseen. Se rakennettiin yleistyökaluksi, jota voi helposti käyttää eri tekoälytilanteisiin koulutuksen aikana. [7.]

Seq2seq-malleilla (sequence to sequence) tarkoitetaan malleja, jotka ovat erikoistuneet Recurrent-neuroverkkoihin. Näitä käytetään tavallisesti ratkomaan sellaisia ongelmia kuin konekääntäminen, kysymyksiin vastaaminen, keskustelubottien tekeminen tai tekstin tiivistäminen. [8.]

Seq2seq-malleja käytetään esimerkiksi Google-kääntäjässä. Seq2seq-malli on tarkempi kuin Googlen aiemmin käyttämät mallit. Myös Googlen puhetunnistus käyttää kyseistä mallia.

Mallin rakenne tavallisissa tapauksissa on encoder-decoder-arkitehtuuri. Rakenne koostuu kahdesta osasta encoderista ja decoderista. Encoder muuntaa tekstin koneen ymmärtämään muotoon. Decoder muuntaa lopuksi vastauksen käyttäjän haluamaan muotoon. Encoder muokkaa sisääntuloa tekoälysovelluksen ymmärtämään muotoon, ja decoder muokkaa tulosta ulostulon tarvitsemaan muotoon. [9.]

Kumpikin ovat LSTM-malleja (Long short term memory). Encoder lukee tulon ja tiivistää sen joko sisäisen tilan vektoriksi (internal state vector) tai tilanteen vektoriksi (contex vector). Mallissa ei oteta huomioon tulevaa tulosta, vaan se säästää sisäisen tilan vektorina. Vektorin tavoitteena on sisältää tieto kaikista lähdön elementeistä, minkä avulla se auttaa decoderia tekemään tarkempia ennustuksia. [10.]

Piilotettu taso h_t lasketaan käyttämällä kaavaa

$$h_t = f\left(W^{(hh)} h_{t-1} + W^{(hx)} x_t\right)$$

jossa W on paino, h on edellinen tila ja x on tulo ajan hetkellä t.

Decoder on LSTM, jonka alku on tehty decoderin LSTM. Encoderin viimeisen neuronin tulos annetaan decoderin ensimmäiseen neuroniin. Näitä arvoja käyttäen decoder alkaa tuottamaan ulostuloa ja näitä käytetään myös tulevien ulostulojen määritelmässä.

Joukko näitä decoderin LSTM-yksiköitä laitetaan tekemään ennustuksia tietylle ajalle. Jokainen uudelleen käytetty yksikkö hyväksyy piilotetun tilan edelliseltä yksiköltä ja tuottaa ja lähettää sen ja myös oman piilotetun tilan.

Dekoder laskee piilotetun tilan käyttäen edellistä tilaa ja kertomalla sen sille tarkoitetulla muuttujalla. Se laskee ulostulon softmax-funktiolla, joka tuottaa todennäköisyysvektorin. NLP-neuroverkko käyttää tullutta vektoria auttamaan tekemään lopullista ulostuloa. [11.]

Dekoderin ongelmia ovat esimerkiksi arkkitehtuurillisesti niiden pieni muisti. Tämä johtuu siitä, kuinka LSTM rakenteellisesti loppuu, koska viimeisessä piilotetussa tilassa LSTM:ssä laitetaan koko teksti tulkittavaksi. Tämä teksti on yleisesti muutaman sadan yksikön pituinen. Tämän takia suurien tekstien kanssa tarvitaan suurempia neuroverkkoja. Liian suuret neuroverkot taas ovat yleisesti hennompia, eli neuroverkolla on heikompi yhteys sanojen kanssa ja niiden kouluttaminen kestää pidempään.

Tämän takia Seq2seq-neuroverkosta tulee helposti monimutkaisia. Se on muissakin neuroverkoissakin ongelma, koska niitä on vaikeampi kouluttaa ja kouluttaminen kestää kauemmin verrattuna yksinkertaisempiin neuroverkoihin. Yleisesti ongelmana on se, että syvien neuroverkkojen kouluttaminen vaatii huomattavan paljon aikaa ja tehoja. Monimutkaisempi neuroverkko ei myöskään tarkoita sitä, että neuroverkosta olisi enemmän hyötyä, vaan yleisesti ne ovat vain hiukan tarkempia. Vaikka LSTM yrittää vähentää tätä, ei se poista tätä rakenteellista ongelmaa.

Seq2seq-malleja käytetään hyvin paljon nykyisessä NLP-tekoälyjen teossa ja ERNIE-GEN on tehty tämän periaatteen mukaan.

3.2 PaddlePaddle

PaddlePaddle on Kiinan Googlea vastaavan Baidun avoimen lähdekoodin koneoppimisen kehys. Kuten Google antaa käyttää vapaasti, TensorFlow-rajapintaa erilaisissa sovelluksissa, niin Baidu antaa käyttää PaddlePaddlea. [11;12.]

PaddlePaddle on ollut vapaasti käytettävissä sitten vuodesta 2016. PaddlePaddlella on paljon toimintoja, jotka kattavat syväoppimiskehyksen, tavalliset mallikirjastot, loppukäyttäjän välineet, työkalut ja komponentit ja myös palvelualustoja. PaddlePaddle on otettu käyttöön laajasti tuotanto- ja palvelualoille. Kehyksellä on yli 2,3 miljoonaa sovelluskehittäjää. Tämä on auttanut lisäämään tekoälyn käyttöä maailmalla. [9.]

Käytännössä PaddlePaddle on vain apu neuroverkkojen tekemiseen ja koneoppimiseen, joka on suosittu Kiinassa, mutta on vielä länsimaissa tuntematon.

4 ERNIE-GEN

ERNIE-GEN on paranneltu esikoulutettu ja hienosäädetty kehys luonnolliseen kielen tekemiseen. Tällä hetkellä ERNIE-GEN:llä on kielivaihtoehtoina kiina ja englanti. ERNIE-GEN on tehty PaddlePaddlen päälle, ja se käyttää PaddlePaddlen ominaisuuksia. [13.]

ERNIE-GEN on tehty tuottamaan luonnollista kieltä samalla tavalla kuin ihminen tuottaa. ERNIE-GEN on toteuttanut sen paremmin kuin muut vastaavat tekoälyt. [14.]

Helpoin tapa käyttää ERNIE-GEN:iä on käyttää valmiita malleja, joita tekijät tarjoavat heidän github-sivullaan. Lisäksi ERNIE-GEN tarvitsee toimiakseen vain Python-ympäristön ja pip-pakkaustenhallintasovelluksen. Käyttöönotto-ohjeet ovat englanniksi ja kiinaksi. Tosin sovellukset mitä ERNIE-GEN vaatii toimiakseen ja niiden asennusohjeet ovat kiinaksi. Onneksi komentokehotteet on kirjoitettu englanniksi, joten asennuksen etenemistä voi seurata, vaikka ei osaisi kieltä. [15.]

ERNIE-GEN on rakennettu eri tavalla kuin tavallinen NLP-tekoäly. Tämän rakenteellinen ero mahdollistaa ERNIE-GEN:n oppivan dataa keskivertoa NLP-tekoälyä nopeammin ja paremmin. Se ilmenee paremmin tuotettuna tekstinä. ERNIE-GEN on tullut tarkemmaksi kuin perinteiset seq2seq-mallilla tehdyt tekoälyt, sillä siinä on käytetty myös vanhempia, erittäin tehokkaiksi osoittautuneita menetelmiä. [16;17.]

ERNIE-GEN julkaistiin toukokuussa 2020. Se on hyvin esikoulutettu malli tehtävien generoimiseen. Täyttösanojen (engl. Infilling generation) ja kohinan (engl. noise) havaitseva menetelmä ovat arkkitehtuurillisesti uusia NLP-tekoälyissä.

ERNIE-GEN on myös mallinnettu toimimaan monisäikeisyyden huomioivan arkkitehtuurin mukaan.

Parempiakin ERNIE-malleja on julkaistu. Viimeisin ERNIE-M-malli on julkaistu joulukuussa 2021. Myös aikaisemmat mallit ovat hyvin arvostettuja.

4.1 Eroavaisuuksia muista kehyksistä

ERNIE-GEN on parannettu Seq2seq-koulutuskehys. Kaksi suurinta eroavaisuutta ovat lauseen sisällön rakentaminen ja datan sisäisten virheiden tuottamisen. [16.]

Lauseen täytön tekemisessä suurin erikoisuus on, että kun puhuttavaa lausetta muodostetaan ja tutkitaan, pidetään hetken rauhaa. Tämä tehdään sen takia, että paino viimeiseen sanaan putoaisi, mikä on ongelma edellisissä kehyksissä. Sanojen erilaiset kontekstit tarkistetaan kesken neuroverkon koulutuksen, kun taas aikaisemmissa katsottiin vasta lopussa. [14.]

Datan sisäisten virheiden tuottaminen on tekoälyissä tärkeää, jotta tekoäly huomaa, miten erilaisia virheitä sen tulee huomata tulevissa datassa. ERNIE-GEN hoitaa tämän niin, että se vaihtaa sattumavaraisesti sanoja toisiin sanakirjan sanoihin. Vaikka tapa on hyvin yksinkertainen, sen on huomattu olevan erittäin tehokas.

Myös ERNIE-GEN:in tekstin tuottaminen on hyvin johdonmukaista. Sillä ERNIE-GEN hoitaa span-by-span-tuottamisenmenetelmän. ERNIE-GEN käsittelee tulevaa lausetta sana sanalta ja määrittää jokaista sanaa kohden ennustuksen seuraavasta sanasta. ERNIE-GEN, samalla kun se arvailee tulevia sanoja, pitää arvaillut sanat tietyissä aiheissa kontekstista riippuen. Kun taas perinteiset tekoälyt arvaavat sana kerrallaan, miten edetä.

Pitkissä teksteissä on huomattu myös, että ERNIE-GEN:n tuottama teksti pysyy paremmin yhtenäisenä verrattuna UNILM- ja MASS-malleihin. UNILM- ja MASS-mallit ovat aikaisempia esikoulutettuja NLP-neuroverkkomalleja, jotka

keskittyvät enemmän kouluttamaan yhtäaikaa encoderia ja decoderia. ERNIE-GEN on erittäin tehokas abstraktin tekstin kuvaamiseen, kysymyksien tekemiseen, dialogin tuottamiseen ja kysymyksiin vastaamiseen.

ERNIE-GEN hoitaa hyvin koulutuksen. Koulutuksen aikana, jokaisen koulutuksen tehtävän jälkeen, ERNIE-GEN muokkaa sen parametreja hoitamaan paremmin tehtäväänsä. Tämän avulla ei menetetä dataa, kun tehtävät on suoritettu.

ERNIE-GEN säätää sen parametreja kolmessa osassa sanan, rakenteen ja merkityksen huomioivaan esikoulutustehtävään. Sanoja huomioivan tehtävän tarkoitus on saada tekoäly arvaamaan esimerkiksi, mikä on virkkeen alkuperäinen sanajärjestys. Toinen tehtävä on arvioida, mitkä sanat pitää aloittaa isolla kirjaimella, mikä voi helpottaa tekstin ymmärtämistä. Viimeinen tehtävä on tunnistaa, kuinka sanat ovat eri järjestyksessä lauseissa.

Rakenteen huomioivaan esikoulutustehtävään kuuluvia toimenpiteitä ovat esimerkiksi virkkeen sanajärjestyksen muokkaaminen. Toinen osatehtävä on selvittää, miten lähellä virkkeet ovat keskenään. Siihen on annettu kolme arvoa. Ovatko ne vierekkäin samassa dokumentissa, ovatko ne samassa dokumentissa vai ovatko virkkeet kokonaan eri dokumenteista peräisin?

Periaatteena on, että ERNIE-GEN ei unohda niin paljon koulutustehtävien välillä. Tämä parantaa ERNIE-GEN:n tarkkuutta sille rakennettuun tehtävään. Tämän takia ERNIE-GEN saa tuotettua parempaa tekstiä haluttuun tehtävään, koska se pienentää tietokoneen vaadittavaa tehoa ja antaa paremman lopputuloksen.

Kuvassa 3 verrataan ERNIE-GEN:ä muutamiin toisiin luonnollista kieltä tuottaviin tekoälyihin. Niitä ovat esimerkiksi Bert, Pegasus Seq2Seq ja Platon. Näitä malleja on testattu eri tilanteissakin.

Kuva 3 kertoo, kuinka ERNIE-GEN on edennyt ja kuinka se kehittyy, kun sitä koulutetaan. Lisäksi verrataan, kuinka sen base-versio ja large-versio eroavat

keskenään koulutuksessa. Base-version rakennettiin siten, että sen piilotettujen kerrosten määrä on 12, sen piilotettu koko (engl. hidden size) on 768, itseään tutkivia neuroneiden (engl. self-attention heads) määrä on 12 ja kaikkien parametrien määrä on 110 M. Kun taas large-version piilotettujen kerrosten määrä on 24, piilotettu koko on 1024, itseään tutkivia neuroneiden määrä on 16 ja kaikkien parametrien määrä on 340 M.

Task	Epoch		Learning Rate		Noising Rate ρ_f		Dropout Rate		Batch Size	Label Smooth	Beam Size	Evaluation Metric
	BASE	LARGE	BASE	LARGE	BASE	LARGE	BASE	LARGE				
SQuAD QG	10	10	2.5e-5	1.5e-5	0.7	0.7	0.1	0.2	32	0.1	1	BLEU-4, METEOR (MTR), ROUGE-L (RG-L)
CNN/DailyMail	30	20	5e-5	4e-5	0.7	0.7	0.1	0.1	64	0.1	5	ROUGE-F1 scores: ROUGE-1 (RG-1), ROUGE-2 (RG-2), ROUGE-L (RG-L)
Gigaword	10	5	3e-5	3e-5	0.5	0.6	0.1	0.2	128	0.1	5	
Persona-Chat	-	30	-	1e-4	-	0.0	-	0.1	64	0.1	10	BLEU-1, BLEU-1, Distinct-1, Distinct-1
Generative CoQA	-	10	-	1e-5	-	0.5	-	0.1	32	0.1	3	F1-score

Kuva 3. ERNIE-GEN base- ja large-versioiden vertailutaulukko. [16.]

4.2 ERNIE-GEN:n käyttöönotto

ERNIE-GEN-kehyksen voi ladata ilmaiseksi GitHubista. Sieltä voi ladata myös samalla valmiita datasettejä englannin ja kiinan kielille. [14.]

ERNIE-GEN-kehys tarvitsee Python-version 3.6 tai uudemman. Pip pitää olla vähintään versio 20.2.2**.**

Esimerkiksi ERNIE-GEN käyttää PaddlePaddle 1.7:ää tai uudempaa version runkoa. Jotta PaddlePaddle toimii, se kaipaa pythonista version 3.6 tai uudemman.

Koulutus- ja käyttötiedostot pitää olla tsv-tyyppisiä (engl. tab-separated values), eli data-arvot pitää erotella toisistaan tabuloinnilla. Datasetin pitää olla tietyllä tavalla rakennettu. Sarakkeiden tunnusten pitää olla text_a ja label. Helpoimmalla pääsee, kun ensimmäiseksi lataa datasetin. Tästä saa kaksi tiedostoa, josta yksi tiedosto on koulutusta varten ja toinen on testejä varten. Tämän jälkeen data muokataan haluttuun muotoon, samalla tavalla kuin muillekin neuroverkoille. Lisäksi vaihdetaan sarakeotsikot. Tämän jälkeen jaotellaan data kahteen osaan perinteisellä tavalla 80:20-jaottelulla, josta 20 on testidatan määrä.

Tämän jälkeen pitää laittaa data oikeisiin paikkoihin. Tämän jälkeen pitää siirtää train-, dev- ja test- tiedostot datasettiin. Parametrit siirretään parametrikansioon, ja se pitää siirtää ERNIE-kansioon.

Lisäksi hakemiston polut pitää lisätä ympäristömuuttujiin sitten, että datasettiin ja parametrit ovat oikein.

4.3 ERNIE-GEN:n saavutukset

ERNIE-GEN:iä voi käyttää esimerkiksi sellaisten pelien tekemiseen, joissa tietokone itse generoi joitakin huoneiden kuvauksia. Koska se voi tuottaa suuria

määriä kirjoituksia hetkessä, niin sitä voi käyttää myös uusien ideoiden tuottamiseen. Ideat, jotka neuroverkko generoi, on kumminkin oikein kirjoitettu, jonka takia se herättää ideoita helposti.

ERNIE-GEN-tekoäly voi auttaa kirjoittamisen oppimista. Vaikka osaisikin kirjoittaa oikein kieliopillisesti, ei se tarkoita, että osaisi tuoda esiin ajatuksia helposti kirjoitetussa muodossa.

Koska ERNIE-GEN on ilmainen rajapinta, se on hyvä pohja sovelluksien teolle. Sitä voi käyttää monissa erilaisissa sovelluksissa. ERNIE-GEN:in ilmaisuus haastaa myös maksullisten kehitysympäristöjen tekijöitä tuottamaan entistä parempia toteutuksia.

ERNIE-GEN mahdollistaa tarkemman konekäännöksien tekemisen kuin aikaisemmat neuroverkot. Tämä helpottaa ja auttaa ihmisiä, jotka eivät tiedä täysin toista kieltä, mutta haluavat asioida sillä kielellä.

Koska ERNIE-GEN antaa valmiiksi koulutettuja malleja, tämä mahdollistaa neuroverkon käyttämisen ilman, että tarvitsee kouluttaa niitä käyttäen omaa dataa. Tämä nopeuttaa tiettyjen sovelluksien tekoa ja mahdollistaa parhaassa tilanteessa käyttää sitä suoraan joihinkin muihin sovelluksiin.

4.4 Kielivaihtoehdot

Kielivaihtoehtona on tällä hetkellä englannin kieli ja kiinan kieli. Niille on valmiiksi 12 esikoulutettua mallia. [14.]

Malleja voi myös itse kouluttaa eri kielisiksi, kun ERNIE-GEN:lle tarjoaa tarpeeksi laajan datasetin ja harjoittaa sen sillä perusteella. [14.]

Kielille, joissa käytetään paljon prepositioita, saadaan helpommin tehtyä toimiva tekoäly kuin kielille, joissa on vähemmän prepositioita. Tämä johtuu tekoälyn tekotavasta. Tekoälyn mielestä esimerkiksi talo ja talon ovat kaksi eri sanaa. Se,

että tekoäly ymmärtäisi, että talon on vain sanan talo genetiivimuoto, vaatii tekoälyltä paljon enemmän oikeinkirjoituksen suhteen kuin esimerkiksi englannin prepositiot.

ERNIE-GEN:n edeltäjässä, ERNIE 2.0 -mallissa, englannin kielen koulutukseen käytetty data on peräisin Reditistä ja Googlelta. Kiinan kielen koulutusdata on peräisin Baidun uutisista ja sanakirjoista. Kiinan kielen esikoulutetut mallit käyttävät suuremman määrän tiedostoja kuin mitä englannin kielen esikoulutetut mallit käyttävät. Yksi syy voi olla kokonaan kirjoitusmerkkien määrän takia. Toinen on se, että ERNIE-GEN-kehitystyötä on tehty kiinan kielellä.

Tekijät tarjoavat useita valmiiksi tehtyjä runkoja, joita ovat esimerkiksi muutama kiinan eri kielillä ja muutama englannin kielillä. Tämä helpottaa ERNIE-GEN:n käyttöönottoa, koska ei tarvitse kouluttaa ja käyttää siihen aikaa. Kun neuroverkkoa tehdään, siitä pitää aina tarkistaa sen tarkkuus. Neuroverkon muodostus vie aikaa, eikä se välttämättä tuota ensimmäisellä kerralla vielä hyvää tulosta.

4.5 ERNIE-GEN:n rajoituksia

Vaikka ERNIE-GEN pystyy tuottamaan hyvin tekstiä, ERNIE-GEN ei vielä tuota yhtä hyvää tekstiä kuin mitä kirjoittajat tuottavat. Se ei esimerkiksi kykene vielä valmistamaan mitään mestariteoksia kirjallisuudesta.

Lisäksi kannattaa myös tarkistaa, mitä ERNIE-GEN kirjoittaa. Se ei välttämättä kirjoita mitään järjellistä, koska se katsoo tiettyjä sanojen tarkoitusta erilaisena eikä sillä ole tietoa, miten oikea maailma toimii, joten tarvitaan vielä tuotetun tekstin manuaalista tarkistamista ja parantamista.

ERNIE-GEN voi parantaa chatboxien toimintaa merkittävästi, jotta saadaan tuntumaan, että oikea ihminen on vastaamassa. Myös tämä voi auttaa yllättävissä tilanteissa tekoälyn puolesta.

ERNIE-GEN on vain rajapinta, jota rakennetaan. Se ei välttämättä sovellu kaikkiin ongelmiin. Esimerkkinä sovellukseen mihin se ei välttämättä sovellu, on ei-NLP-ongelmat. ERNIE-GEN on erikoistunut tekstin tuottamiseen, jonka takia tekstin analysointi voi olla heikompi ei-NLP-ongelmissa kuin toisilla tai itse rakennetulla tekoälyllä.

4.6 ERNIE-ohjelmia ja niiden eri tarkoituksia

ERNIE-sovelluksia on rakennettu toteuttamaan vastauksia erilaisiin kysymyksiin. Tässä luvussa käydään lyhyesti läpi ERNIE 3.0, ERNIE-ViLG ja ERNIE-GEN:in pohjautuvaan kiinalaisen tekstin yhteenvetoalgoritmiin, jota kutsutaan nimellä ERNIE-GEN-CTS.

ERNIE 3.0 on viimeisin pääversio, mitä Baidu on julkaisut. Se on iso neuroverkko, jossa on 175 miljardia parametria. Tämä malli on Baidun mukaan saavuttanut SuperGLUE-suorituskykytestissä (GLUE tulee sanoista General Language Understanding Evaluation) ihmistä paremmin tuloksen (90,6 % verrattuna ihmisen 89,8 %). Mallia ei ole vielä vapaassa käytössä. [18.]

SuperGLUE-suorituskykytesti on rakennettu GLUE-suorituskykytestin päälle. Näiden tarkoitus on testata NLP-sovelluksia erilaisissa tilanteissa. SuperGLUE antaa GLUE:een verrattuna vaikeampia testejä, jotka vaativat luovaa ratkaisuja. Toinen SuperGLUE tavoitteista on myös saada aikaan uusi julkinen tulostaulukko. Myös jotkin näistä SuperGLUE:n antamista kysymyksistä ovat monitulkinnaisia, mutta sellaisia, jotka ihmiset osaavat ratkaista helposti. Yksinkertaisesti SuperGLUE on uusi tasokoe NLP-neuroverkoille. [19;20.]

ERNIE-ViLG -malli laatii kuvasta tekstiä vaativiin tehtäviin. Näitä tarvitaan esimerkiksi automaattiseen kuvatekstien käytössä. ERNIE-ViLG-malli toteuttaa myös tekstistä-kuva funktion, eli se osaa tuottaa tekstin pohjalta kuvan. Tämä on mahdollista ilman ylimääräistä hienosäätöä. ERNIE-ViLG osaa muokata kuvia, kun annetaan tarkempia ohjeita. Tätä on esimerkiksi testattu käyttämällä kiinalaisia runoja. [21.]

Tarkemmin testeistä testattiin käyttämällä kahta yleistä kiinalaista kuvateksti datasettejä, AIC-ICC ja COCO-CN, ja näitä käyttämällä koulutettiin kaksi ERNIE-ViLG-neuroverkkoa. Tämän jälkeen ne ERNIE-ViLG validointiin käyttämällä BLUE@4, METEOR, ROUGE-L ja CIDERr suorituskykytestejä. Näistä ERNIE-ViLG sai paremmat tuloksen kuin valmiiksi koulutetut mallit.

ERNIE-GEN-CTS on kiinalaistekstin yhteenvetoalgoritmin tuottamiseen tehty sovellus. Tämä eroaa edellisistä malleista pienentämällä altistusvaikutusta, joka oli merkittävä ongelma edellisistä yhteenvetomalleissa. Mutta ERNIE-GEN-CTS pitää vielä tutkia, sillä tulokset saatiin käyttämällä vasta pientä datasettiä käyttäen. [22.]

5 Yhteenveto

Insinöörityössä käytiin läpi, millainen ERNIE-GEN on ja kuinka sitä otetaan käyttöön. Myös käytiin läpi ERNIE-GEN:in vahvuuksia ja heikkouksia. Työssä käytiin läpi, miten neuroverkot toimivat ja kuinka yksinkertainen neuroverkko rakennetaan. Samalla käytiin läpi, kuinka NLP-sovellukset toimivat ja mitä ne ovat.

Työn edetessä huomattiin, että ERNIE-GEN erottui paremmaksi malliksi kuin vertaisensa, varsinkin pitkissä tekoälyn tuottamista teksteissä. ERNIE-GEN:llä on tehty viime aikoina uusija malleja, jotka ovat onnistuneet saamaan hyviä tuloksia erilaisissa testeissä.

Samalla käytiin läpi, kuinka NLP-sovellukset ovat edenneet ja kuinka ne ovat tulleet paljon tarkemmaksi näiden viimeisten vuosien aikana. Toivottavasti insinöörityön avulla saadaan aikaan suurempaa huomiotta Baidun valmistamiin NLP-neuroverkkoihin ja saadaan enemmän parempia NLP-sovelluksia.

Kokonaisuudessa työ kuvasi, kuinka NLP-sovellukset ovat edenneet ERNIE-GEN:in avulla ja miten ollaan pian tilanteessa, missä nämä sovellukset ovat tavallisen ihmisen tasolla.

Lähteet

1 An Ultimate Tutorial to Neural Networks 2022. Verkkoainesto. Simplilearn <https://www.simplilearn.com/tutorials/deep-learning-tutorial/neural-network> Luettu 15.3.2022.

2 Epoch in Neural Networks by baeldung. 2021. Verkkoainesto. Baeldung <https://www.baeldung.com/cs/epoch-neural-networks#:~:text=Epoch%20in%20Neural%20Networks%20An%20epoch%20means%20training,we%20use%20all%20of%20the%20data%20exactly%20once.> Luettu 31.3.2022.

3 Neha Seth, 2021 Estimation of Neurons and Forward Progation in Neural Net. Verkkoainesto. Analytics Vidhya <https://www.analyticsvidhya.com/blog/2021/04/estimation-of-neurons-and-forward-propagation-in-neural-net/> Luettu 15.3.2022.

4 Matthew Mayo, 2017 Neural Network Foundations, Explained: Updating Weights with Gradient Descent & Backpropagation. Verkkoainesto. KDnuggets <https://www.kdnuggets.com/2017/10/neural-network-foundations-explained-gradient-descent.html> Luettu 31.3.2022.

5 Advantages and Disadvantages of Neural Networks. 2020 Verkkoainesto. Bealdung <https://www.baeldung.com/cs/neural-net-advantages-disadvantages> Luettu 31.3.2022.

6 Natural language processing. Verkkoainesto. Wikipedia <https://en.wikipedia.org/wiki/Natural_language_processing> Luettu 6.2.2022.

7 Seq2Seq Model | Understanding Seq2seq Model Architecture. 2020 Verkkoainesto. Analytics Vidhya <https://www.analyticsvidhya.com/blog/2020/08/a-simple-introduction-to-sequence-to-sequence-models/> Luettu 8.4.2022.

8 Francois Chollet 2017, A ten-minute introduction to sequence-to-sequence learning in Keras. Verkkoainesto. The Keras Blog <https://blog.keras.io/a-ten-minute-introduction-to-sequence-to-sequence-learning-in-keras.html> luettu 31.1.2022

9 PaddlePaddle. Verkkoainesto. Baidu <https://github.com/PaddlePaddle/Paddle> luettu 3.7.2022

10 Jason Brownlee 2017, A Gentle Introduction to Long Short-Term Memory Networks by the Experts. (summer 2021) Verkkoainesto. Machine Learning Mastery <https://machinelearningmastery.com/gentle-introduction-long-short-term-memory-networks-experts/> luettu 2.28.2022

11 Tutorial: Neural Machine Translation seq2seq. Verkkoainesto. Google <https://google.github.io/seq2seq/nmt/> luettu 31.1.2022

12 Joseph Nelson, 2021 What is PaddlePaddle?. Verkkoainesto. roboflow <https://blog.roboflow.com/what-is-paddlepaddle/> luettu 24.2.2022

13 Fatema Patrawala 2019, Baidu open sources ERNIE 2.0, a continual pre-training NLP model that outperforms BERT and XLNet on 16 NLP tasks. Verkkoainesto. Packt <https://hub.packtpub.com/baidu-open-sources-ernie-2-0-a-continual-pre-training-nlp-model-that-outperforms-bert-and-xlnet-on-16-nlp-tasks/> luettu 22.3.2022

14 Ram Sagar 2019, BAIDU's ERNIE 2.0 Gets NLP Top Honours, Eclipses BERT & XLNet. Verkkoainesto. Analyticsindiamag<https://analyticsindiamag.com/baidus-ernie-2-0-gets-nlp-top-honours-eclipses-bert-xlnet/> luettu 22.3.2022

15 ERNIE/README.en.md. Verkkoainesto. Baidu <https://github.com/PaddlePaddle/ERNIE/blob/develop/README.en.md#setup> luettu 7.3.2022

16 Ernie 2.0: A Continual Pre-Training Framework for Language Understanding | AISC. video. Verkkoainesto. AISC <https://www.youtube.com/watch?v=8K1IX7VJ5Fc> Luettu 23.2.2022.

17 ERNIE-GEN: An Enhanced Multi-Flow Pre-Training and Fine-Tuning Framework for Natural Language Generation. 2020 Verkkoainesto. Baidu <https://deepai.org/publication/ernie-gen-an-enhanced-multi-flow-pre-training-and-fine-tuning-framework-for-natural-language-generation> Luettu 22.3.2022.

18 ERNIE 3.0: Large-scale Knowledge Enhanced Pre-training for Language Understanding and Generation. 2021 Verkkoainesto. Baidu <https://arxiv.org/pdf/2107.02137.pdf> Luettu 14.4.2022.

19 SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. Verkkoainesto. The New York University Center for Data Sciences <https://w4ngatang.github.io/static/papers/superglue.pdf> Luettu 6.5.2022.

20 Jakub Lewkowicz. 2019 SuperGLUE benchmark challenges natural language processing tasks. Verkkoainesto. SDTimes <https://sdtimes.com/ai/superglue-benchmark-challenges-natural-language-processing-tasks/> Luettu 6.5.2022.

21 ERNIE-ViLG: Unifield Generative Pre-Training for Bidirectional Vision-Language Generation. 2021 Verkkoainesto. Baidu <https://arxiv.org/pdf/2112.15283.pdf> Luettu 14.4.2022.

22 Chinese Text Summarization Generation Algorithm based on ERNIE-GEN with Few-shot Learning. 2021 Verkkoainesto. ECS <https://iopscience.iop.org/article/10.1088/1742-6596/1961/1/012051/pdf> Luettu 14.4.2022.