Tampereen yliopisto

Mikael Mansikka

UUTISDATAN LUOKITTELU

Suomenkieliset uutisotsikot

Diplomityö
Informaatioteknologian ja viestinnän (ITC) tiedekunta
Prof. Jyrki Nummenmaa
Prof. Jaakko Peltonen
10 2020

TIIVISTELMÄ

Tekstinkäsittely on saanut jalansijaa kasvavan datamäärän ja internetin myötä. Tekstinkäsittelyn eräs sovellus on tekstin kategorisointi eli tekstinluokitus. Tekstin luokittelun tarkoitus on jakaa tekstimuodossa oleva data ennalta määrättyihin luokkiin. Tämä tarkoittaa siis sitä, että tekstin luokittelun tarkoitus on siis selvittää funktio, jonka avulla voidaan mallintaa näytteiden kuuluvuus eri kategorioihin.

Tekstin luokittelun sovellukset ovat osa nykyistä arkipäivää. Tekstipohjaisen datan automaattinen luokitus, automaattinen sähköpostisuodatus, potilasasiakirjojen automaattinen luokittelu, suositusjärjestelmät ja esimerkiksi hakukoneet (Google, Bing, jne.) ovat esimerkkejä, joissa tekstin luokittelua käytetään vähentämään resurssikulutusta.

Diplomityön tarkoituksena on tutkia, minkälaisia menetelmiä on luokitella tekstiä sisältävää dataa ja kuinka hyvin ne toimivat. Tärkeimpänä huomiona metodien valitsemiseen on kieli, joka tässä työssä on suomi. Diplomityössä päädytään vertaamaan neljää eri luokittelijaa SVM (Support Vector Machine/Tukivektorikone), Naiivi Bayes, BiLSTM (Bidirectional Long-Short-Term Memory), BERT (Bidirectional Encoder Representations of Transformers) suomenkielisen uutisdatan luokittelussa. Kyseiset metodit on valittu aikaisempien tutkimuksien tuloksien, sekä niiden helppokäyttöisyyden ja toteutuksien saatavuuden takia.

Uutisdata koostuu uutisten otsikoista sekä näiden luokituksesta viiteen eri luokkaan (”Urheilu”, ”Kotimaa”, ”Ulkomaat”, ”Politiikka”, ”Kulttuuri”). Luonnollisten kielten käsittelyn eli NLP (Natural Language Processing) tekniikoiden avulla uutisotsikot muunnetaan numeeriseen muotoon, jota on mahdollista käsitellä koneellisesti. Tätä vaihetta kutsutaan esiprosessoinniksi. Koska esiprosessoitu data tuottaa parempia tuloksia, kuin häiriöitä sisältävä data. Työssä hyödynnetään esiprosessointina datan alkiot eli tässä tapauksessa sanat muunnetaan pieniksi kirjaimiksi, sekä ylimääräiset ja merkityksettömät sanat eli stoppisanat poistetaan. Esiprosessoinnin jälkeen ominaisuuksien valitseminen on vaihe, jonka tarkoitus on auttaa luokittelijoita luokitustehtävässä. Tämän jälkeen TF-IDF-muunnoksella sekä Python-kirjastojen valmisfunktioilla saavutetaan halutut koneellisesti ymmärrettävät muodot.

Luokitteluun käytettävistä metodeista parhaiten suoriutuu BERT. Kyseinen metodi sai parhaimmaksi tarkkuudekseen 94 %, mikä on 2 prosenttiyksikköä korkeampi kuin muut työssä vertailtavat algoritmit ja metodit. Toiseksi parhaimman tarkkuuden saavutti SVM jakaen saman tarkkuuden BiLSTM-neuroverkon kanssa. Työssä alhaisimman tarkkuuden tuotti Naiivi Bayes, jonka ominaisuudet selittävät sen toimivan heikosti epätasaisen jakauman omaavalla datalla.

ABSTRACT

Mikael Mansikka: Finnish news headlines classification
Master of Science Thesis
Tampere University
Master's Degree programme in Signal Processing and Machine Learning
10 2020

The field of natural language processing is a continuously expanding field of machine learning. Text classification is an application of NLP. Text classification has become more important because of its use on data from the world wide web. The purpose of text classification, which is also known as text categorization is to assign textual data into predefined categories. This means that the purpose of text classification is also to generate a function, which models the membership of data in separable categories.

Applications of text classification are part of everyday life. Examples of everyday text classification include e-mail filtering and search engines (Google, Bing, etc.).

The purpose of this Master of Science Thesis is to examine, what kind of methods exists for classifying text-based data on how well they work. The most important factor when choosing methods for text classification is the language of the text. In this thesis the language is chosen to be Finnish. During this thesis, the focus is on SVM (Support Vector Machine), Naive Bayes, BiLSTM (Bidirectional Long-Short-Term Memory) and BERT (Bidirectional Encoder Representations of Transformers). These methods are chosen, because of their success in previous studies and because of all previously mentioned methods have easy to use implementations available.

The data consist of news headlines and corresponding labels. Each news headline is assigned with one ”true” label, which can be either ”Homeland”, ”Abroad”, ”Sports”, ”Culture”, ”Economy” or ”Politics”. Natural Language Processing techniques are used to transform news headlines into numerical format. The first step in text classification is to preprocess the data. In this work data is preprocessed to only have lowercase letters. Part of preprocessing is to remove irrelevant words, which are also known as stopwords. Feature selection method is chosen to be TF-IDF and other Python libraries are used as feature selection tools when needed.

The best text classification method in this Master of Sciences Thesis is BERT, which achieved accuracy of 94 percent. Compared to other text classification methods, BERT achieved 2 percentage points higher accuracy. After BERT, the second most accurate method is SVM with the same score as BiLSTM. Naive Bayes performed the worst out from selected methods.

ALKUSANAT

Tämä diplomityö on toteutettu yhteistyössä Insta DefSec Oy:n kanssa. Työssä käytettävä data ja sen soveltuvuus käyttötapaukseen on suunniteltu yhdessä Insta DefSec Oy:n kanssa. Työn tarkoitus on toimia jatkumona harjoittelijana olemiselle, sekä jatkaa harjoittelun aikana käytettyjen ja tutkittujen aihealueiden tutkimista syvemmin.

Innoitus työn aiheeseen on saatu Insta DefSec Oy:ltä, tästä syystä haluaisinkin kiittää Insta DefSec Oy:tä sekä erityisesti esimiestäni Jarkko Hartikaista. Suuri kiitos työn edistymisestä ja valmistumisesta kuuluu juuri esimiehelleni Jarkko Hartikaiselle, jonka tuella työ on saatu kasaan, ja jonka luvalla sain vapaat kädet menetelmien valitsemiseen. Kiitokset kuuluvat myös ohjaajilleni Prof. Jyrki Nummenmaalle ja Prof. Jaakko Peltoselle.

Haluaisin kiittää myös Suomen Kielipankkia, joka on mahdollistanut työssä käytettävän datan hyödyntämisen.

Tampereella 2.10.2020

Mikael Mansikka

SISÄLLYSLUETTELO

KUVALUETTELO

LYHENTEET JA MERKINNÄT

CNN	engl. Convolutional Neural Network, konvoluutioneuroverkko
BERT	engl. Bidirectional Encoder Representations from Transformers, luokittelualgoritmi
BiLSTM	engl. Bidirectional Long Short-Term Memory, luokittelualgoritmi
GPU	engl. Graphical Processing Unit, grafiikkasuoritin eli näytönohjain
LSTM	engl. Long Short-Term Memory, luokittelualgoritmi
NB	engl. Naive Bayes, Naiivi Bayes -luokittelualgoritmi
NLP	engl. Natural Language Processing, luonnollisten kielten käsittely
RNN	engl. Recurrent Neural Network, neuroverkkorakenne
SVM	engl. Support Vector Machine, luokittelualgoritmi
VRAM	engl. Video Random Access Memory, grafiikkasuorittimen sisäinen muisti

1. JOHDANTO

Tekstin luokitusta eli tekstin kategorisointia on tutkittu useaan otteeseen viimeisten vuosikymmenien aikana. Automaattisella tekstinluokituksella on tarvetta useammassa yhteiskunnan osa-alueessa, joissa datan määrä on jatkuvassa kasvussa. Asiakasarviointien, suositusjärjestelmien, sähköpostisuodatuksien ja dokumenttien järjestämisen alueella jatkuvat virrat dataa vaativat jo jonkin tasoista automaattista luokitusta ja järjestelyä, jotta kaiken sen datamäärän hallitseminen on mahdollista.

Tekstinluokituksella voidaan myös parantaa tekstipohjaisen datan laatua. Tekstinluokitukseen käytettävien metodien kyky luokitella esimerkiksi vihapuhetta, huijausviestejä tai sosiaalisen median julkaisuja on tukittu paljon.

Automaattinen tekstinluokitus auttaa yrityksiä järjestämään datavirtaa ja hallitsemaan käytettävissä olevaa tekstipohjaista dataa. Automaattinen tekstinluokitus vähentää ihmisen käsin tehtävää työtä ja mahdollistaa ihmisresurssien siirtämisen muuhun tarkoitukseen, kuin mekaaniseen datan jäsentelyyn. Ihmisen käsittelyssä datavirran luokittaminen on hidasta, joka aiheuttaa pahimmillaan ajallisten ihmisresurssien menetystä.

Aiemmissa tutkimuksissa todetut lineaariset menetelmät sekä neuroverkkoratkaisut ovat osoittautuneet datan luokitukseen liittyvissä tehtävissä hyvin suoriutuviksi. Tyypilliset valinnat tekstin luokitusta varten ovat SVM ja erilaiset neuroverkkoratkaisut, kuten CNN tai RNN. Lyhyiden tekstien, kuten uutisten, sosiaalisen median kirjoituksien tai tuotearvosteluiden luokitus on osa tekstinluokituksen laajaa sovelluskirjoa. Datan, varsinkin tekstipohjaisen datan kasvava määrä johtaa tilanteeseen, jossa kaikkea saatavana olevaa datavirtaa on mahdotonta luokitella ihmisvoimin. Tämän takia automaattista, ilman ihmisvoimaa toimivia menetelmiä kehitetään ja niiden tuloksia parannetaan tutkijoiden kesken.

Tämän diplomityön tarkoitus on tutkia automaattisia menetelmiä luokitella suomenkielisiä lyhyitä tekstejä. Työssä lyhyitä tekstejä kuvaavat uutisotsikot, jotka on kerätty vuosilta 2012-2018. Uutisotsikot ovat kirjoitusasultaan vaihtelevia ja yksittäinen uutisotsikko voidaan usein tulkita kuuluvan useampaan kuin yhteen luokkaan. Uutisten automaattisen luokituksen merkitys kasvaa jatkuvasti saatavissa olevan tarjonnan kanssa. Erilaiset internet-palvelut tarjoavat monia mahdollisuuksia analysoida uutisvirtaa ja seurata analyysin tuloksia mm. älypuhelimien kautta. Uutisvirran analysoimiseen käytettäviä työkaluja on paljon, joten työn tarkoituksena on kartoittaa ja tutkia erilaisten työkalujen ja metodien

toimintaa. Diplomityön tutkimuksessa pyritään vastaamaan kahteen seuraavaan kysymykseen:

1. Mitä menetelmiä on luokitella suomenkielistä uutisdataa automaattisesti?
2. Kuinka hyvin suomenkielistä dataa voidaan luokitella ja mitkä suomenkielisen uutisdatan luokitukseen käytettävistä metodeista toimivat parhaiten?

Kysymys 2 on kysymyksen 1 alikysymys, jonka tarkoitus on toimia tukena työssä käytettävien menetelmien valinnassa.

Diplomityön rakenne on seuraavanlainen. Luku kaksi koostuu kirjallisuuskatsauksesta eli aiempien tutkimuksien läpikäynnistä. Luvun kaksi aliluvuissa esitellään luokittelumenetelmiä ja niiden toimintaa teoreettisella tasolla, ottamatta kantaa työssä käytettäviin metodeihin. Luvussa kolme esitetään tässä työssä käytettävät metodit ja tarkkuuden mittaamiseen käytettävät mittarit. Luvun kolme tarkoitus on esitellä työhön valitut menetelmät, menetelmien tarjoavat kirjastot sekä parametrit, joita menetelmiä on käytetty. Neljännessä luvussa käydään läpi tulokset ja pohditaan tuloksien aiheuttajia. Viidennessä luvussa on yhteenveto sekä tulevaisuuden tutkimuksen kohteet. Viimeisenä esitetään työssä käytetyt lähteet.

2. TEKSTIN LUOKITTELUN METODIT / KIRJALLISUUSKATSAUS

Luonnollisten kielten käsittely on ollut tutkimuksen kohteena, jotta erilaisten kielten ominaisuudet voitaisiin toistaa tietokoneella käsiteltävässä muodossa. Kielelliset ominaisuudet voidaan muuttaa koneellisesti käsiteltäväksi luonnillisesti esiintyvästä kielestä. Luonnillisesti esiintyvä kieli tarkoittaa ihmisten välistä kommunikaatiota, joka voi olla sekä tekstiä että puhetta. Luonnollisten kielten käsittelyn toisena tarkoituksena on toimia mallina ihmisten käyttämälle kielelle. Voidaankin todeta, että NLP eli luonnollisten kielten käsittelyn tavoite on taata mahdollisuus ilmaista luonnollista kieltä mahdollisimman ihmismäisesti.

Luonnollisten kielten käsittelyssä tyypillisin ongelma on kielten monimuotoisuus. Luonnollisten kielten käsittelyssä eräs tärkeämmistä piirteistä on tunnistaa käsittelyssä oleva kieli. Kielten rakenteessa, merkistössä ja säännöstössä on eroavaisuuksia, jotka tekevät luonnollisten kielten käsittelystä monimutkaista ja tarvitsevat kielen tunnistamisen toimiakseen.

Tekstin luokitteleminen on yksi osa lukuisista luonnollisten kielten käsittelyn sovelluksista. Muita luonnollisten kielten käsittelyä käyttäviä sovelluksia ovat tekstin tiivistäminen, puheentunnistus, tekstin erilaiset muunnokset ja käyttöliittymät. Tekstin luokituksessa tarkoituksena on käyttää hyväksi NLP-menetelmiä ja kielen eri ominaisuuksia, jotta teksti voidaan jakaa ennalta määrättyihin tai tuntemattomiin luokkiin. Tekstin luokittelun teoriaa ja sen sovelluksia käsitellään työn seuraavassa osiossa.

2.1 Tekstin luokittelun teoria

Tekstin luokittelun tarkoitus on jakaa alkiot (esimerkiksi uutiset, sähköpostit, arvostelut tai tekniset raportit) ennalta määrättyihin tai ennalta määräämättömiin luokkiin. (Zhang, 2008) Luokilla tässä tapauksessa voidaan tarkoittaa esimerkiksi aihepiiriä tai yleisimmin esiintyviä sanoja. Kokonaisuudessaan tekstin luokittelu koostuu datasta $D = \{X_1, X_2, X_3, \dots, X_n\}$, missä X_i on tekstialkio kuten uutinen, arvostelu tai dokumentti. Jokainen alkio koostuu lauseista l sekä sanoista s. Datan tekstialkiot on jaettu luokkiin $K = \{1,2,3, \dots, k\}$. Tämän pohjalta erilaisten tekstin luokittelun metodien on tarkoitus muodostaa funktio, joka parhaiten erottelee käsittelyssä olevan datan luokat toisistaan.

Tekstin luokittelun merkitys nousee erityisen tärkeäksi, kun käytettävissä oleva datan määrä on suuri. Erityisesti täysin automaattinen tekstinluokitus parantaa ja nopeuttaa suuren datamäärän luokittelua tilanteissa, joissa valmiiksi annotoitua dataa tai käsin tehtäviä annotaatiota ei ole saatavana. Tässä yhteydessä annotoitu data tarkoittaa dataa, jossa halutut ulostulot ovat jo merkitty valmiiksi. Esimerkki valmiiksi annotoidusta datasta sisältää mahdolliset luokat (kuten "roskaposti" ja "ei roskaposti") sekä molempiin luokkiin kuuluvat tekstisisällöt.

Tekstin luokittelu voidaan jakaa automaattiseen ja manuaaliseen prosessiin. Manuaalinen tekstin luokittelu tässä tapauksessa tarkoittaa sitä, että ihminen manuaalisesti jakaa tekstit luokkiin. Tämän kaltainen tekstin luokittelu on kuitenkin hidasta, resursseja kuluttavaa, sekä kasvattaa luokittelusta vastaavan kognitiivista taakkaa ja kognitiivisen taakan takia virheitä. Automaattinen tekstin luokittelu on tekstin luokittamista koneellisin keinoin, vähentäen resurssien kulumista. Automaattisen tekstin luokittelun tärkeys korostuu, kun kyseessä on suuret määrät dataa. Suuren datamäärän luokittelu vaatii resursseja, kuten aikaa, mikäli datamäärä luokitellaan manuaalisesti, mutta automaattinen, koneellinen luokittelu mahdollistaa datan luokittamisen nopeasti ja kustannustehokkaasti.

Tekstin luokittelun toteuttamiseen tarvitaan menetelmiä, joilla luokkien väliset erot voidaan mallintaa. Tekstin luokittelun metodit voidaan jakaa kahteen eri ryhmään. Ohjaamattomat menetelmät (“*unsupervised methods*”) ovat menetelmiä, joissa luokittelualgoritmit muodostavat itse luokat ja niiden eroavaisuudet ilman ennalta määrättyä dataa. Tällaiset algoritmit eivät aina vaadi erikseen määritettävää opetukseen- ja testaukseen näytettävää dataa. Voidaan todeta ohjaamattomien menetelmien tarkoituksen olevan tekstin luokittelun mahdollistaminen ilman määritettyjä luokkia. Ohjaamattomat menetelmät tyypillisesti luovat ratkaisun ongelmaan, johon ei ole ennalta määrättyä toivottua lopputulosta, kuten aiheiden ennustaminen.

Toinen ryhmä tekstin luokittelualgoritmeja ovat ohjatut (”*supervised methods*”) menetelmät. Ohjattujen menetelmien ero ohjaamattomiin on valmiiksi luokitellussa datassa. Ohjatut menetelmät vaativat opetusmateriaalin, jonka rakenne koostuu luokittelemiseen käytettävästä sisääntulodatasta, sekä tavoite luokista. Tavoiteluokilla tarkoitetaan ennalta määrättyjä kategorioita/luokkia, joihin datan alkiot on jaettu. Tarkoituksena on käyttää valmiiksi luokiteltua aineistoa, esimerkiksi ihmisten tekstikategorioihin luokittelemia tekstejä kuten käyttäjäarvioita, mallin opetusmateriaalina, jotta saadaan muodostettua kaavan 1 kaltainen tilanne.

$$y = f(x), \tag{1}$$

missä y on haluttu ulostulon arvo ja f funktio, jolla sisääntuloa x vastaava ulostulo mallinnetaan. Ohjatun menetelmän opetusmateriaalin avulla malli oppii valvotusti halutut luokat eli muodostaa funktion f, jonka tarkoitus on kuvata luokkien välisiä eroavaisuuksia. Ohjatut menetelmät ovat myös enemmän resursseja kuluttavia, sillä ne vaativat valmiiksi annotoitua dataa.

Tekstin luokittelun yleisin ongelma on luokittelussa käytettävän sanaston suuri määrä suhteessa itse luokituksessa käytettävään sanamäärään. Tämä voi tarkoittaa esimerkiksi sitä, että luokiteltavat alkiot ovat lyhyitä verrattuna koko käytettävissä olevaan sanaston määrään. Edellä mainittua ongelmaa kutsutaan suuren ja harvan dimension ongelmaksi. Ongelman ratkaisemiseksi on toteutettu erilaisia luokittelumenetelmiä, joiden toimintaa esitetään tarkemmin luvussa 2.5.

Tekstin luokittelu jaetaan usein eri vaiheisiin, jotka ovat esitelty seuraavissa osioissa. Ensimmäinen vaihe on tekstin esiprosessointi. Esiprosessoinnin aikana tekstin merkistö tunnistetaan, ylimääräiset merkit poistetaan, sekä teksti jaetaan alkioihin. Alkiot tässä tapauksessa voivat olla sanoja, lauseita tai muita haluttuja määreitä, jotka valitaan ongelmakohtaisesti. Esiprosessoinnin mahdollistamiseksi, tulee data muuntaa haluttuun muotoon. Haluttu muoto voi olla esimerkiksi JSON-tiedosto tai tekstitiedosto. Haluttuun muotoon muuntaminen voidaan jättää pois, jos aineisto on valmiiksi muodossa, joka luokitteluohjelmistossa tarvitaan.

Toinen tekstin luokittelun vaihe käsittelee tekstin esittämistä laskennallisessa muodossa, joka tarkoittaa numeerista, esimerkiksi vektorimuotoista esitystapaa. (V Korde, 2012) Vektorimuotoisen esitystavan ideana on muuntaa tekstin alkiot koneellisesti ymmärrettävään muotoon. Vektorimuotoisen esitystavan avulla datan alkioiden vertailu sekä haluttujen dimensioiden valitseminen on helpompaa. Laskennallisen muodon tekeminen voidaan yhdistää myös kolmannen vaiheen (tekstin piirteiden valitsemisen) kanssa.

Kolmas vaihe on tekstin piirteiden valitseminen (*'feature selection'*). Piirteiden valitsemisen tarkoitus on auttaa luokittelualgoritmien toimintaa ja joissain tapauksissa jopa parantaa tulosten tarkkuutta. Piirteiden valitsemisella pyritään vähentämään tekstissä olevien ylimääräisten ja luokituksen kannalta merkityksettömien alkioiden määrää. (Mähönen Mika, 2013) Työssä käytettäviä piirteiden valitsemisen tekniikoita käsitellään tarkemmin luvussa 3.3.

Neljäntenä vaiheena tekstin luokittelu matemaattisten mallien avulla. Neljäs vaihe tarkoittaa datan luokittelemista valitulla algoritmilla/metodilla. Vaihe sisältää metodin valitsemisen ja sen käyttämisen opetusdatalle. Neljäs vaihe voidaan jakaa kahteen osaan: **1.** opetusosa ja **2.** testausosa. Opetusosan tarkoitus on syöttää käytettävä opetusdata metodille, jonka perusteella metodi, joka voi olla esimerkiksi matemaattinen malli muodostaa päätelmät luokista ja niitä erottavista piirteistä. Opetusdata koostuu kahdesta osasta (oikeat luokat ja sisältö), kuten luvussa on aiemmin mainittu. Opetusosa johtaa luokitteluun käytettävän metodin kykyyn tunnistaa halutut luokat myös tuntemattomalla alkiolla eli alkiolla, jota metodi ei ole aiemmin nähnyt. Testausosan vaihe on validoida menetelmän oppima funktio käyttäen dataa, joista menetelmälle ei kerrota haluttuja luokkia.

Viimeisenä vaiheena ovat tulokset sekä tulosten arviointi. Tämän vaiheen tarkoitus on varmistaa sekä tarkistaa luokittelijan kyky luokitella data. Tuloksia voidaan myös arvioida muullakin kuin pelkällä tarkkuudella ja siksi tähän vaiheeseen kuuluvat pohdinnat ovat tärkeä osa itse tulosten sanallinen arviointi. Tulosten arvioinnilla on toinenkin tehtävä. Tulosten arvioinnin ja erilaisten tarkkuuden mittarien avulla voidaan vertailla metodien välisiä yhteyksiä sekä kykyjä. Vertailu mahdollistaa parhaimman mahdollisimman metodin valitsemisen useampien metodien joukosta. Tarkoituksena on tutkia vastaako metodi esitettyihin tutkimuskysymyksiin. Tämän takia tarkkuuden mittareiden käyttäminen on pakollista, jotta lopulliset päätelmät, kuten vastaavatko käytetyt metodit haluttuja tutkimuskysymyksiä voidaan tehdä.

Tekstin luokittelu sisältää viisi erilaista vaihetta, joilla kaikilla on omat tehtävänsä tekstin luokittelussa. Kuitenkin voidaan sanoa, että tekstin luokittelua varten tarvittavia vaiheita on itse asiassa kuusi. Ennen edellä mainittuja vaiheita on myös olemassa vaihe, datan kerääminen ja sen tutkiminen. Tämän vaiheen avulla saadaan tieto siitä, minkälaisesta datasta on kyse ja minkälaiset omat ongelmat data tuo tekstinluokituksen eri vaiheisiin. Datan keräämistä, sekä muita tekstin luokittelun vaiheita esitellään tarkemmin seuraavassa luvussa.

2.2 Metodit

Kappaleiden 2.3-2.5.6 tarkoitus on esitellä datan keräämistä, esiprosessointia, sekä luokittelualgoritmeja ja niiden toimintaa. Kappaleessa käydään myös lyhyt katsaus työssä käytettävään dataan ja tekstin luokittelun soveltamiseen uutisdatalla. Tarkoiteuksena on antaa käsitys luokittelualgoritmeista, aiemmista tutkimuksista sekä luokittelualgoritmien toiminnasta ennen varsinaista algoritmien valintaa.

2.3 Datan kerääminen

Datan keräämiseksi on esitetty tapoina mm. webscraping, joka tarkoittaa tiedon automaattista hakua internetistä ohjelmistollisesti, kuten Wikipediasta sekä erilaisista tietokannoista. Datan kerääminen tekstimuodossa voi sisältää esimerkiksi useita tiedostomuotoja, kuten ”.txt”, ”.pdf” ja ”.docx”. Datan keräämisessä tarkoitus on koota mahdollisimman laaja ja sovelluskohteeseen sopiva kokonaisuus. Datan keräämisen tärkeys korostuu tarvittavan datan määrästä ja luonteesta. Tarkkuutta vaativat sovellutukset lisäävät sopivan datan merkitystä ja ongelmakohtaisen vastaavan datan käyttämistä opetusvaiheessa.

Datan kerääminen on työn vaihe, jossa merkitykselliseksi nousee valmiit internetistä ladattavat datakokonaisuudet, sekä kaikki mahdollinen käyttökohteeseen kuuluva valmiiksi annotoitu data. Ilman valmiita datakokonaisuuksia on mahdollista suorittaa tekstin luokittelua, mutta se vaatii suuren resurssikulutuksen datan etsimistä ja annotoimista varten.

Datan rakenne tekstin luokittelun kannalta voi olla useanlainen. Data voi sisältää pitkiä tekstejä, kuten kokonaisia tieteellisiä artikkeleita ja kirjoja tai data voi sisältää vain lyhyitä tekstejä kuten sosiaalisen median julkaisuja tai uutisotsikoita. Aiemmin mainittu datan pituus tulee ottaa huomioon jo keräämisvaiheessa, jotta kerätty data vastaa käsittelyssä olevaa ongelmaa.

Uutisotsikoiden pituus merkkeinä on lyhyt, jonka takia voidaan todeta uutisotsikoiden luokituksen kuuluvan lyhyiden tekstien luokitukseen. Huomattavaa on myös lyhyiden tekstien vaikutus itse luokitukseen, sillä lyhyet alkiot (tekstit) sisältävät hyvin vähän informaatiota ja ovat siksi haastavimpia luokitella. Uutisotsikoiden teksti on sanastoltaan runsasta ja eri vuosien uutisotsikot voivat olla teemoiltaan hyvinkin erilaiset. Trendit, tapahtumat ja ilmiöt luovat omat haasteensa jokaiselle vuodelle. Uutisotsikot ovat erittäin laaja-alaisia ja saattavat sisältää sanoja useammasta eri luokasta, mikä lisää uutisotsikoista koostuvan tekstin luokittelun vaikeutta ja datan dimensioiden hallinnan vaikeutta.

2.4 Esiprosessointi ja piirteiden valinta

Data on esiprosessoitu tulosten yhtenäistämiseksi ja parantamiseksi, tarkoituksena myös taata mahdollisimman yksinkertainen ja toistettavissa oleva koeasetelma. Esiprosessoinnin tehtävänä on poistaa ylimääräiset, luokituksen kannalta epäolennaiset osat datasta, jonka voidaan myös huomata vaikuttavan luokitustulokseen sekä luokkien päällekkäisyyteen. Esiprosessoinnin aikana käsiteltävissä oleva data ja sen piirteet tulevat tutuiksi, jonka takia tekstin luokittelu vaatii osaamista myös datarakenteista. Esiprosessoinnin voidaan sanoa kuuluvan erääksi tärkeimmistä tekstin luokittelun vaiheista, sillä sen avulla vähennetään datassa olevaa kohinaa, joka parantaa metodien tarkkuutta.

Esiprosessoinnin tyypillisiä vaiheita ovat merkistön siistiminen, stoppisanojen poistaminen, avaimiin jako, jne. Näitä menetelmiä esitellään tarkemmin seuraavissa kappaleissa.

Tyypillinen esiprosessoinnin ensimmäinen vaihe on muodostaa saaduista tiedostoista tiedostomuoto, jota on ihmisen helppo lukea. Esiprosessoinnissa ihmisluettava tiedostomuoto on tärkeä työn alkuvaiheessa virheiden ja muiden epäjatkuvuuksien tarkasteluiden sallimiseksi. Ihmisluettavuuden lisäksi tiedoston parsiminen edesauttaa tekstin prosessoimista, poistamalla ylimääräiset suomen kieleen kuulumattomat merkit ja symbolit. (Mähönen Mika, 2013) Tällaisia merkkejä ja symboleja esiintyy esimerkiksi HTML- ja XML-tiedostoissa, joissa kyseisen tiedostomuodon standardin merkistö eroaa suomenkielisestä ja luonnollisen kielen merkistöstä. Esiprosessoinnissa on juuri tyypillistä muuttaa käsiteltävä data haluttuun tiedostomuotoon tai formaattiin, josta edempänä esitellyt metodit voidaan toteuttaa koneellisesti.

Esiprosessoinnin tärkeä tehtävä on yhtenäistää teksti ja sen sisältämät alkiot (sanat), kuten aiemmin mainittu. Esiprosessoinnissa merkistölle tehdään muutoksia käsittelyssä olevan ongelman piirteiden perusteella. Sanojen yhtenäistäminen esimerkiksi muuttamalla kirjaimet pelkästään pieniksi kirjaimiksi on yksi keino. Kirjainten muuntaminen pienillä kirjoitettuihin kirjaimiin tuo esiin pohdinnan aiheita. Jotkin sanat ja lauseet voivat menettää merkitystään, kun ne muunnetaan pieniksi kirjaimiksi. Tästä syystä tärkeää on muistaa säännönmukaisuus koko esiprosessoinnin aikana. Pelkästään pieniksi kirjaimiksi muuttaminen ei usein ole riittävää tekstin luokittelun kannalta ja siksi esiprosessointiin voidaan sisällyttää ylimääräisten välimerkkien, kuten esimerkiksi pisteiden, pilkkujen, kysymysmerkkien ja huutomerkkien poistaminen.

Mahdollisimman parhaimman tuloksen saamiseksi data on pilkottu avaimiin (*”tokens”*) sekä datasta on poistettu stoppisanat (”stopwords”). Avaimiin jakaminen tarkoittaa tekstin jakamista valittuihin osakokonaisuuksiin, jotka voivat olla esimerkiksi sanoja tai lauseita. (Mähönen Mika, 2013) Jakaminen avaimiin tarkoittaa myös sitä, että niitä voidaan

käsitellä yksitellen, joka helpottaa esimerkiksi seuraavaksi esitettyä vaihetta eli stoppisanojen poistamista.

Stoppisanat ovat sanoja, joilla on vähän tai ei lainkaan merkitystä ymmärtämisen kannalta. Tällaisia sanoja ovat suomen kielessä esimerkiksi pronominit sekä konjunktiot, kuten ”ja”, ”mutta”, ”se” ja ”hän”. Stoppisanoihin voidaan luokitella kuuluvan myös muut ylimääräiset tai liian usein esiintyvät sanat. Liian usein esiintyvillä sanoilla tarkoitetaan tässä tapauksessa sanoja, jotka esiintyvät dokumenteissa usein, mutta jotka eivät indikaatiota dokumentin luokasta. Stoppisanojen poistamiseksi on luotava tai käytettävä listaa, johon datassa esiintyviä sanoja verrataan.

Keinoja yhtenäistää sanat ovat stemmaus ja lemmatisointi. Lemmatisointi tarkoittaa sanan perusmuotoistamista. Suomen kielessä tämä tarkoittaa sitä, että esimerkiksi sanat lintu, linnut ja lintujen taipuvat samaan perusmuotoon ”lintu”. Lemmatisoinnissa tarkoituksena ei ole siis poistaa vain sanan vartalon päätteitä vaan hakea sanan perusmuoto. Lemmatisoinnilla voidaan myös välttää suomen kielessä erilaisten sanapäätteiden vaikutusta itse luokitustulokseen.

Lemmatisoinnin lisäksi on olemassa sanojen stemmaus eli päätteiden poistaminen, joka esiintyy useassa aiemmassa tutkimuksessa käytettävänä metodina. Stemmauksen ideana on poistaa sanan pääte, joka yhtenäistää sanoja ja vähentää sanamuotojen merkitystä luokitukseen. Stemmauksessa tyypillisin ja yksinkertaisin versio koostuu listasta, jonka mukaan stemmausalgoritmi poistaa kyseisessä listassa olevat päätteet sanojen vartaloista. (Vijayarani, 2015)

Tekstin piirteiden valintaan kuuluu tekstin muuttaminen luokittelualgoritmeille sopivaan muotoon. Luokittelualgoritmeista esimerkiksi SVM ja Naiivi Bayes käyttävät molemmat TF-IDF-muunnosta. Kaavan 2 mukaisesti TF-IDF-muunnoksen tarkoitus on muuntaa alkiot tekstissä (sanat) vektoreiksi.

$$TFIDF = \log(1 + f_{ij}) * (\log(\frac{td}{td_{wi}})), \qquad (2)$$

missä f_{ij} on sanan i esiintymistaajuus dokumentissa j, td on kaikkien dokumenttien määrä ja td_{wi} on sanan i sisältävien dokumenttien määrä. (U. Pujianto, 2019) TF-IDF-muunnoksen takana on muodostaa malli, jossa dokumentit (esimerkiksi lauseet) esitetään sanarykelminä. TF-IDF-muunnoksen etuna muihin vastaaviin dokumenttien esittämismuotoihin on se, että se painottaa sanoja kahdella eri tapaa, kertoen sanan esiintymistaajuuden (TF) käänteisellä esiintymistaajuudella (IDF). Ensimmäisessä tavassa (TF) sanojen esiintyvyys tietyssä dokumentissa lasketaan, tätä kutsutaan termitaajuudeksi. Toisessa tavassa (IDF) lasketaan sanan esiintyvyys kaikissa dokumenteissa, joka

tarkoittaa käänteistä termitaajuutta. (Donghwa Kim, 2019) TF-IDF-muunnoksen mukaan sana/termi on tärkeä, mikäli sen esiintymistaajuus on suuri tietyssä, yhdessä dokumentissa ja pieni kaikissa muissa dokumenteissa.

2.5 Luokittelualgoritmit

Luokittelualgoritmeja, joita hyödynnetään tekstin luokitteluun, on monia. Tähän asti tutkimukset ovat keskittyneet lineaarisiin luokittelijoihin. Kuten aiemmin on todettu, ohjatut luokittelumenetelmät vaativat opetus- ja testivaiheen. Opetusvaiheen tarkoitus on muodostaa yhdessä luokittelualgoritmin kanssa malli, jolla eri luokat mallinnetaan. Testivaiheen tarkoitus on varmentaa opetusvaiheen tuloksia.

Seuraavissa alaluvuissa 2.5.1-2.5.6 suoritetaan kirjallisuuskatsaus kuuteen erilaiseen luokitteluun sopivaan algoritmiin.

2.5.1 SVM

SVM eli Support Vector Machine (Tukivektorikone) luokittelee kaksi näytejoukkoa mahdollisimman tarkasti tukivektoreiden avulla. Tukivektorikone eli SVM mallintaa datan avaruuteen, jossa ongelman käsitteleminen onnistuu jakamalla luokat omille alueilleen. Ensimmäinen versio SVM-luokittelualgoritmista kehitettiin vuonna 1995 Vapnik ja Cortes toimesta. SVM-luokittelualgoritmin muodostamien tukivektoreiden hypertasolla erotetaan kaksi luokkaa toisistaan. Oletetaan, että binäärisen ongelman tapauksessa data koostuu N määrästä vektoreista $x_i (i = 1, 2, 3, \dots, N)$, jotka ovat luokitellut luokkiin $y_i \in \{-1, +1\}$. Näiden kahden luokan ja niiden välisen hypertason avulla etsitään paras malli uuden datan luokittamista varten. Hypertaso toimii luokituksessa siten, että se etsii suurimman etäisyyden, lähempänä toisiaan olevien eri luokkien alkioiden välillä, joka ratkaisee kaavan 3 ongelma, missä w on painovektori.

$$\gamma(w, \varepsilon) = \frac{1}{2}\|w\|^2 + c\sum_{i=1}^{N} \varepsilon_i, \quad (3)$$

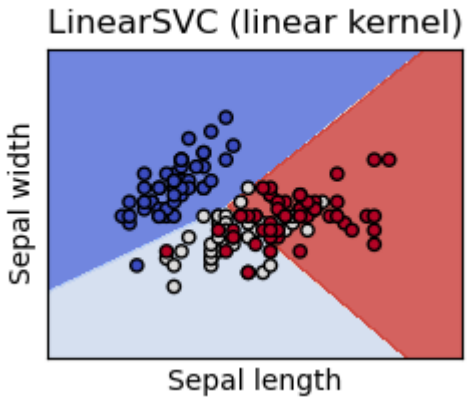
missä c on regularisointiin käytettävä termi ja ε_i on ns. slack variable -muuttuja, joka kuvaa vaatimusta näytteen i etäisyydestä luokkarajasta. Tämän takia SVM tarkkuus riippuu tukivektoreiden muodostamisesta ja se toimii parhaiten kahden luokan tapauksessa. SVM kyky muodostaa hypertasot korkeissa ulottuvuuksissa mahdollistaa sen oppimisen riippumatta datan ominaisuuksien dimensioista. Tämän hypoteesin avulla voidaan todeta SVM kyvyn oppia olevan riippumaton ominaisuuksien määrästä ja perustuvan vain SVM kykyyn erottaa data toisistaan tukivektoreiden muodostamalla hypertasolla. SVM kykyä luokitella erilaisissa tehtävissä voidaan parantaa käyttämällä erilaisia kernelfunk-

tioita. SVM-luokittelualgoritmia voidaan käyttää sekä lineaarisella kernelillä että epälineaarisella kernelillä. Lineaarinen kernel kuitenkin muiden tutkimusten mukaan tuottaa yhtä hyviä, ellei jopa parempia tuloksia, kuitenkin ollessaan riippuvainen käsiteltävästä ongelmasta. Kuvassa 1 on esitetty yksi mahdollisuus SVM toiminnasta lineaarisella kernelillä. Tarkoituksena kuvalla on toimia esimerkkinä, minkälaiset luokkien väliset tasot lineaarinen SVM voi saada aikaan kolmen luokan tapauksessa.

SVM-luokittelualgoritmin tarkkuutta ja soveltuvuutta luokittelutehtäviin on tutkittu paljon. (Zhang, 2008) (Goudjil, 2018) (Chen;Zhong;Xie;& Cai, 2014) Erilaisia käytännöllisiä ratkaisuja on myös esitetty SVM-luokittelualgoritmin käyttämiseen. (A. B. Prasetijo, 2017)

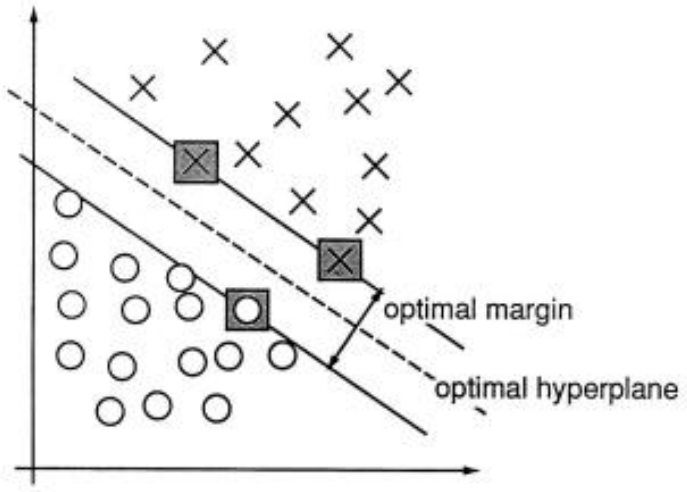
SVM-luokittelualgoritmi on tekstidatan kanssa käytetty useasti tekstin luokitteluun. Erilaiset NLP sovellukset ovat olleet SVM-luokittelualgoritmin kanssa tutkimuksen alla. Nämä tulokset ovat osoittaneet SVM-luokittelualgoritmin suoriutuvan tekstinkäsittelyyn liittyvistä tehtävistä hyvin, ellei jopa erinomaisesti. Tarkkuus SVM-luokittelualgoritmille on tutkimuksissa osoittautunut olevan monilla testiaineistoilla yli 90 %, kuten esimerkiksi Niusha Shafiabady ym. saavat tutkimuksessaan Reuters-uutisdatalla testitarkkuudeksi 95,1 %, sekä Wen Zhang tutkimus, jossa lineaarinen SVM-luokittelualgoritmi saavuttaa 91,68 % tarkkuuden. (Niusha Shafiabady, 2016) (Zhang, 2008) Reuters-uutisdata sisältää 11228 kansainvälistä uutista luokiteltuna yli 46 kategoriaan.

Manek, Shenoy, Mohan ja Venugopal tutkivat työssään SVM sekä Naiivin Bayesin kykyä suoriutua elokuvien arvosteluiden luokittelussa yhdessä Gini-indeksin kanssa. Elokuvien, kuten muidenkin arvosteluiden luokittelu liittyy mielipiteiden louhintaan. Tulokset osoittavat lineaarisen SVM pärjäävän parhaiten suhteessa muihin työssä verrattaviin luokittelijoihin. (Asha S Manek, 2017) (Ahmed, 2018) SVM-luokittelualgoritmin käyttöä on tutkittu myös valvomattomana. Shafiabady, et.al. osoittavat työssään SVM-luokittelualgoritmin suoriutuvan luokittelutehtävästä myös ohjaamattomalla menetelmällä, jossa datana on käytetty koneellisesti annotoitua tekstiä. (Niusha Shafiabady, 2016) Joachims huomaa tutkimuksessaan SVM-luokittelualgoritmin suoriutuvan paremmin kuin yksikään tavanomaisista menetelmistä, kuten kNN, jopa parametrien valinnasta riippumatta. (Joachims, 1998)



Kuva 1 *sklearn.svm.linearSVC SVM-luokittelualgoritmin toiminta (saatavana: sklearn)*

Toinen SVM-luokittelualgoritmin etu on sen kyky muodostaa useampi yhdistelmä eri luokkien erottelevasta hypertasosta. Usea yhdistelmä tarkoittaa sitä, että SVM-luokittelualgoritmi voi opetusvaiheessa muodostaa enemmän kuin yhden erilaisen hypertason, joiden kykyä toimia tehtävässä voidaan verrata keskenään. Useamman yhdistelmän muodostaminen tekee SVM-luokittelualgoritmista hyvin kestävän korkeita ulottuvuuksia vastaan. Kuten aiemmin on todettu, tekstidata on harvaa ja suuriulotteista, sekä se sisältää usein vain vähän merkityksettömiä alkioita, joten SVM-luokittelualgoritmi sopii hyvin tekstin luokitteluun. Kuvassa 2 esitetään esimerkki, miten SVM-luokittelualgoritmi muodostaa hypertason tukivektoreiden avulla kahden luokan välille. Kuva 2 esittää tilanteen, jossa harmaalla neliöllä merkatut symbolit kuvastavat SVM-luokittelualgoritmin muodostamia tukivektoreita, joiden tarkoitus on määrittää suurin erotus kahden käsittelyssä olevan luokan välillä (ympyrät ja x-merkit). (Cortes C., 1995)



Kuva 2 *SVM-menetelmän optimaalisen hypertason esimerkki* ***(Cortes C., 1995)***

SVM-luokittelualgoritmin soveltuvuutta tekstin luokitteluun korostaa myös se, että tekstin luokittelun alkiot jakautuvat usein lineaarisesti. Tämän takia lineaarinen SVM-luokittelualgoritmi sekä lineaarinen SVM-luokittelualgoritmin kernel pystyvät muodostamaan vektorit luokkien välille.

2.5.2 Naiivi Bayes

Naiivi Bayes on todennäköisyyteen perustuva luokittelija. Naiivin Bayes -luokittelualgoritmin ominaisuus on olettaa jokaisen luokan piirteen olevan toisistaan riippumattomia. Naiivin bayes -luokittelualgoritmin nopeus perustuu oletukseen, jossa jokaisen luokan todennäköisyys ja jokaisen sanan todennäköisyys kuulua tiettyyn luokkaan voidaan ennustaa erikseen. Tarkoituksena Naiivilla Bayes -luokittelualgoritmillä on siis laskea todennäköisyys, jolla käsiteltävä entiteetti kuuluu tiettyyn luokkaan ja Naiivin Bayes -luokittelualgoritmin kyky luokitella teksti kuuluvaksi tiettyyn luokkaan perustuu siihen, esiintyykö sana tekstissä vai ei. (V Korde, 2012) Naiivin bayes -luokittelualgoritmin kaava, joka pohjautuu Bayes-teoriaan, esitetään kaavassa 4.

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}, \quad (4)$$

missä $X = (x_1, x_2, x_3, \dots, x_n)$. Naiivin Bayes -luokittelualgoritmin tapauksessa voidaan tehdä riippumattomuusoletus, jolla todennäköisyyden estimointi saadaan helpommaksi, jossa $P(x_i|y, x_1, \dots, x_{i-1}, x_{i+1}, \dots x_n) = P(x_i|y)$ kaikilla i arvoilla. Riippumattomuusoletuksen avulla Bayes-teoria yksinkertaistuu muotoon kaavan 5 mukaan,

$$P(y|X) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(X)}, \quad (5)$$

jossa $P(X)$ on vakio syöte, jonka avulla ennustetaan y jolla kaava 5 on suurin. Naiivin oletuksen takia Naiivi Bayes -luokittelualgoritmi osoittautuu tehokkaaksi luokittelijaksi, vaikka oletuksella sanojen välisiä suhteita tai muiden sanojen todennäköisyyksiä ei oteta huomioon. Sanojen väliset suhteet jätetään huomioitta, jolloin kaava tulee tällöin muotoon,

$$P(y|X) \propto P(y) \prod_{i=1}^{n} P(x_i|y), \quad (6)$$

Naiivi Bayes -luokittelualgoritmin kykyä luokitella tekstidokumentteja on tutkittu useasti. (B. Tang, 2016) (Frank E., 2006) (Rennie, 2001) Kuitenkin voidaan todeta sen suoriutuvan parhaiten lineaarisista tehtävistä sekä tehtävissä, jotka sisältävät vain vähän opetusdataa. Naiivi Bayes -luokittelualgoritmi suoriutuu parhaiten, jos käsittelyssä olevaan ongelmaan käytettävän datan ominaisuuksien dimensiot on pienennetty ongelmaan sopivalla tavalla. (Tiago A, 2011) Dimensioiden pienentäminen voidaan esimerkiksi tehdä käyttämällä luvussa 2.4 esitettyjä ominaisuuksien valintaan perustuvia algoritmeja.

Hussain toteaa, Naiivista Bayes -luokittelualgoritmista on olemassa kaksi eri suuntausta, Bernoulli- sekä Multinomial Naiivi Bayes -luokittelualgoritmi. Bernoulli-malli Naiivi Bayes -luokittelualgoritmin suuntaus toimii parhaiten tapauksessa, jossa luokitteluongelma on binäärinen. (Hussain, 2019) Multinomial-mallia on käytetty tutkimuksissa usein tekstipohjaisen datan luokitukseen. (Bayu Yudha Pratama, 2015) Tämän takia työssä käytettävä Multinomial-mallin Naiivi Bayes -luokittelualgoritmi on tutkimuksissa osoittautunut suoriutuvan paremmin laajalla, sekä useampaan luokkaan jakautuvalla datalla. Multinomial-mallin Naiivi Bayes -luokittelualgoritmin ero tavalliseen Naiiviin Bayes -luokittelualgoritmiin on oletus siitä, että tekstien pituus on riippumaton tekstin luokasta. Pituuden ollessa riippumaton luokasta voidaan olettaa myös, että sana on riippumaton sen paikasta ja kontekstista tekstissä. (Vidhya.K.A, 2010)

Naiivia bayes -luokittelualgoritmia käytetään (B. Tang, 2016) tutkimuksessa tekstin kategorisointiin, jossa he ehdottavat uusia menetelmiä tekstin kategorisoimiseen ennalta määrättyihin luokkiin. Samaan kategoriaan kuuluu roskapostisuodatus käyttäen Naiivia Bayes -luokittelualgoritmia. (Tiago A, 2011) Tarkkuudeksi edellä mainituissa tutkimuksissa Naiivi Bayes -luokittelualgoritmi on saanut noin 95 %. U. Pujianto, M. F. Hidayat and H. A. Rosyid tutkimuksessaan saavat Naiiville Bayes -luokittelualgoritmille tarkkuudeksi vaihtelevia arvoja, suurimman tarkkuuden ollessa noin 84 %. (U. Pujianto, 2019) Naiivia Bayes -luokittelualgoritmia on käytetty muuhunkin kuin tekstimuodossa olevan datan luokitteluun. Karandikar, McLeary, Turner ja Schmitz käyttävät Naiivia Bayes -luokittelualgoritmia monitoroimaan työkalujen kulumistasoja. Tutkimuksessa Naiivin Nayes

-luokittelualgoritmin käyttöä perustellaan sen resurssikulutuksen sekä monikäyttöisyyden avulla. (Karandikar;McLeay;Turner;& Schmitz, 2015)

2.5.3 Bi-directional LSTM (Kaksisuuntainen LSTM)

Long Short-Term Memory, lyhyesti LSTM on uusiutuvien neuroverkkojen (RNN) joukkoon kuuluva neuroverkkorakenne. Neuroverkot, kuten LSTM voidaan osoittaa kuuluvan lineaaristen luokittelijoiden joukkoon, kuten suurin osa tekstinluokittamiseen käytettävistä ja soveltuvista algoritmeista.

LSTM ero tavalliseen uusiutuvien neuroverkkojen rakenteeseen on LSTM-neuroverkossa oleva piilotettujen vektoreiden tilalla oleva muistilaatikko. Muistilaatikko varastoi muistisolun, jonka tehtävänä on nimensä mukaisesti varastoida informaatiota. Informaation pidempi varastoiminen mahdollistaa neuroverkolle kontekstin oppimisen.

Kuvan 3 mukaan f on unohdusportti (forget gate). Tämän tehtävä on määrittää, mikä informaatio unohdetaan muistisolun lopussa eli mikä informaatio on epäolennaista kyseistä ongelmaa tarkasteltaessa. (Harjule, 2020) Vastaavasti i ja o ovat sisään-(input-gate) ja ulostuloportit (output-gate).

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i), \quad (7)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f), \quad (8)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o), \quad (9)$$

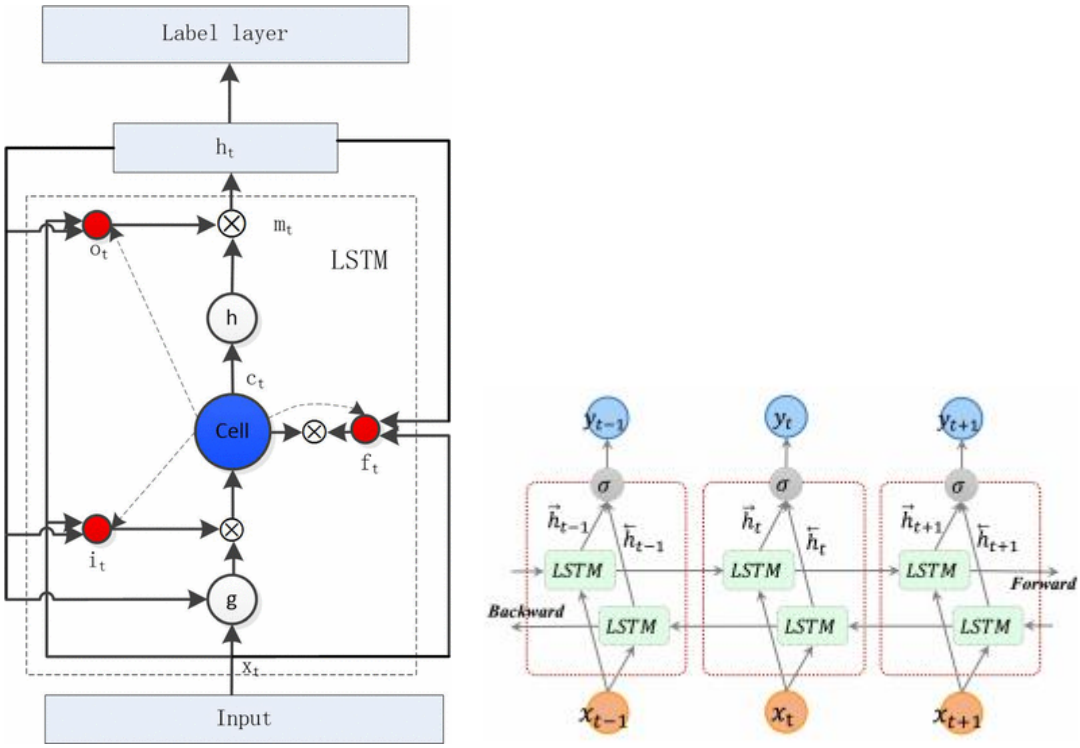
$$ç_t = tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c), \quad (10)$$

$$c_t = f_t \otimes c_{t-1} i_t \otimes ç_t, \quad (11)$$

$$h_t = o_t \otimes tanh(c_t), \quad (12)$$

missä i_t, f_t, o_t ja c_t ovat sisääntuloportin, unohdusportin, ulostuloportin sekä solutilan vektori hetkellä t. Tässä tapauksessa b tarkoittaa vääristymävektoria, sekä W piilotetun kerroksen painoja. Edellä esitetyt kaavat 7, 8, 9, 10, 11, 12 kuvastavat sisääntuloportin, unohdusportin, ulostuloportin sekä solutilan vektoreiden funktioita.

Työssä käytettävä versio LSTM-neuroverkkorakenteesta, kaksisuuntainen LSTM-neuroverkko (BiLSTM), joka on esitetty kuvassa 3. BiLSTM eroaa tavallisesta LSTM-neuroverkon rakenteesta kaksisuuntaisuudellaan. Tämä tarkoittaa sitä, että BiLSTM-neuroverkon läpi kulkeva data luetaan sekä positiivisessa, että negatiivisessa suunnassa (alusta-loppuun ja lopusta-alkuun). Kaksisuuntaisuus on tekniikka, joka mahdollistaa datassa esiintyvän kontekstin ymmärtämisen, sillä neuroverkko pystyy näkemään kaksisuuntaisuudella sekä aiemman, että seuraavan sanan sanajonosta.



Kuva 3 *LSTM-neroverkon rakenne. Vas. LSTM rakenne* ***(Zhiyong Cui, 2020)*** *oik. BiLSTM rakenne* ***(Yan Yan, 2017)***

Aiemmissa tutkimuksissa, kuten Zhang Y., Liu Q. ja Song L. toteavat tutkimuksissaan LSTM saa tarkkuudeksi 80,72 % sekä BiLSTM 81,73 %. Zhang ym. tuloksista voidaan päätellä, että BiLSTM käyttäminen tavallisen LSTM-neuroverkon sijasta hieman parantaa luokituksen tarkkuutta. (Zhang Y., 2018)

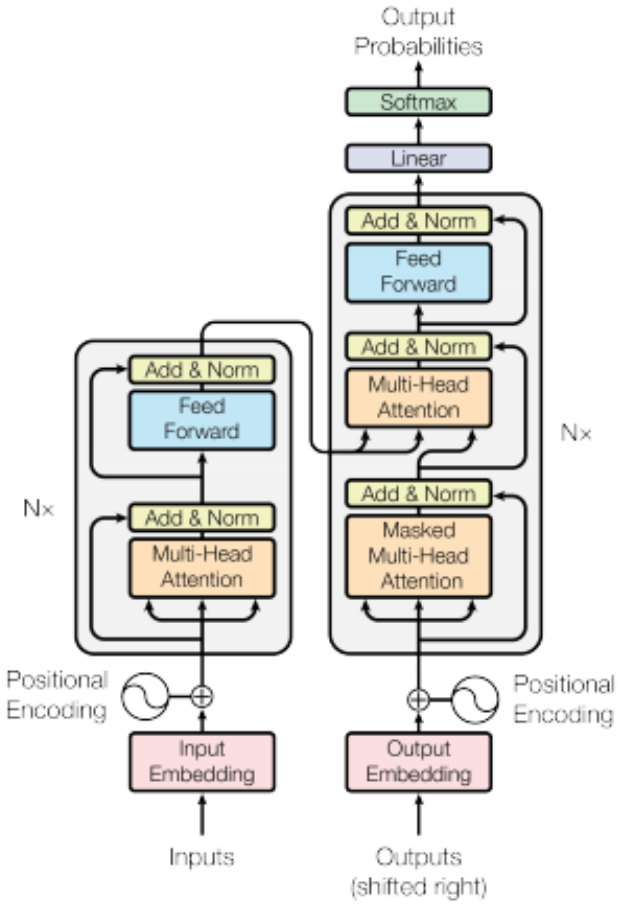
2.5.4 BERT

BERT tulee sanoista Bidirectional Encoder Representations from Transformers. “BERT on Googlen kehittämä malli, jonka tarkoitus on mahdollistaa huippuluokan tulosten saaminen vain yhden ylimääräisen ulostulokerroksen avulla.” (Toutanova, 2018) BERT-malleja voidaan opettaa joko käyttämällä valmentavaa opetusta (*pre-training*) tai täsmentävää opetusta (*fine-tuning*).

BERT-mallin toiminta perustuu MLM eli masked language model –tekniikkaan. MLM-tekniikan ideana on mahdollistaa malli, joka pystyy ymmärtämään sanan sen ympärillä olevan kontekstin avulla. MLM-tekniikka käytää maskia, joka peittaa opetusvaiheessa satunnaisten sanojen id:n. MLM-tekniikalla tämä id selvitetään oppimalla konteksti maskeeratun sanan ympäriltä sekä vasemmalta (aiempi sana) että oikealta (seuraava sana). MLM-tekniikassa huomio kiinnittyy sanojen oppimiseen sekä vasemmalta-oikealle, että oikealta vasemmalle. Kaksisuuntainen oppiminen voi aiheuttaa sen, että malli näkee oikean haetun sanan, sillä sana voidaan etsiä molemmasta suunnasta. Tämän takia MLM-

tekniikan etu on aiemmin mainittu satunnaisten sanojen maskeeraus eli peittäminen, jonka avulla malli ei näe etsittävää sanaa kaksisuuntaisella haulla.

BERT opetuksessa (*pre-training*) on mallinnettu sanojen välisiä suhteita yksinkertaisella seuraavan lauseen ennustamisen -tehtävällä. Seuraavan lauseen ennustamisen tarkoitus on tarjota lause A, jonka jälkeen tulee lause B ja BERT-mallin tarkoitus on oppia mahdolliset vaihtoehdot seuraavilla lauseille.



Kuva 4 *Muuntajien (Transformers) rakenne, enkooderin ollessa kuvassa vasemmalla ja dekooderin oikealla* ***(Polosukhin, 2017)***

Tekniikaltaan BERT-malli pohjautuu transformers-luokittelijoihin eli muuntajiin. Muuntajien perusperiaatteena on yhdistää sanat ja niiden merkitys suhteessa muihin sanoihin, eikä prosessoida sanoja yksi kerrallaan, kuten tavallisten lineaaristen luokittelijoiden periaatteeseen kuuluu. Transformereiden rakenteessa keskeisessä asemassa on päällekkäin asetetut itsensä huomioivat enkooderi- sekä dekooderikerrokset. Rakenteessa

esiintyvät enkooderit sisältävät itsensä huomioivan mekanismin ja koodin purkajat (dekooderit) taas normaalin huomiomekanismin. Kuvasta 4 voidaan nähdä muuntajien rakenne, jossa yhdistyvät täysin yhdistetty enkooderi sekä dekooderi.

BERT-mallin enkooderit koostuvat kahdesta alakerroksesta, jotka ovat huomiokerros ja eteenpäin syöttävä neuroverkkokerros. Huomiokerroksen ideana on mahdollistaa muiden sanojen katsomisen tietyn sanan enkoodauksen kohdalla. Enkoodereita ja dekoodereita esiintyy BERT-mallissa yhtä paljon, joka on tavallisesti 6 kappaletta molempia. BERT-mallin toiminta perustuu enkoodereiden ja dekoodereiden yhteistyöhön. Sisääntuloon käytettävä data syötetään ensimmäisen enkooderin läpi. Tämän enkooderin jälkeen data siirretään seuraavalle enkooderille, kunnes saavutetaan viimeinen enkooderi. Enkoodereilta data siirretään jokaiseen dekooderiin samaan aikaan, joissa lopputulos eli ulostulon arvo lasketaan.

Huomiomekanismeina enkoodereissa ja dekoodereissa käytetään huomiomatriisia. Huomiomatriisi lasketaan kaavan 13 mukaan.

$$huomiomatriisi = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad (13)$$

missä Q on sisään menevä kyselymatriisi, K on avaimet sisältävä matriisi, sekä V on avaimia vastaavien arvojen matriisi. Skaalaukseen käytettävä termi on $\frac{1}{\sqrt{d_k}}$, jonka tarkoitus on korjata yksittäisten suurten alkioiden merkittävyyttä. Skaalauksen termi d_k kuvastaa avainten dimensiota. Enkooderi ja dekooderi käyttävät kuitenkin molemmat multi-head attentionia, joka saadaan huomiomatriisin kaavasta kaavan 14 mukaan.

$$multihead = Concat(head_1, \dots, head_h)W^O, \text{missä } head_i = huomiomatriisi(QW_i^Q, KW_i^K, VW_i^V), \quad (14)$$

missä $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$, $W_i^O \in \mathbb{R}^{hd_v \times d_{model}}$ ja huomiomatriisi on esitetty kaavassa 13. Kaavassa 14 d_{model} tarkoittaa mallin sisääntulon ja ulostulon dimensioita. Multi-head attention eli huomiomekanismin tarkoitus dekooderissa on mahdollistaa datan ymmärtäminen jokaisessa dekooderissa. Samaan tapaan enkooderissa olevan huomiomekanismin tarkoitus on mahdollistaa datan saaminen aiemmasta enkooderista.

BERT-mallin avulla saadaan toteutettua nykyaikaisia tuloksia ilman suuria muutoksia BERT-mallin rakenteessa. BERT-mallin tuloksia tekstin luokittelussa on tutkittu esimerkiksi käyttämällä IMDB-dataa, joka koostuu 50000 elokuva-arvostelusta. Tällaisen arvosteluihin perustuvan luokittelijan tarkkuudeksi saadaan 93,87 % González-Carvajal S., ja Garrido-Merchán E. C. mukaan. (González-Carvajal S., 2020)

BERT malleja on tyypillisesti kaksi kappaletta BERT-large sekä BERT-base. BERT-large -malli on parametreiltään suurempi (340M parametriä), kun taas BERT-base on pienempi malli (110M parametriä). BERT-base-mallin käyttäminen vaatii ainakin 16 GB muistia sisältävän GPU:n. BERT-malli luotiin, jotta on mahdollista luokitella erilaisia datarakenteita, suorittaa NLP tehtäviä ja sovelluksia pelkästään täsmentävää opetusta käyttämällä (fine-tuning) muutamassa tunnissa yhdellä graafisella suorittimella (GPU:lla).

2.5.5 Kapselit

Kapselit ovat neuroverkkojen rakenne, joita voidaan käyttää hyväksi myös tekstinluokituksessa. Kapselit neuroverkossa mahdollistavat visuaalisten kokonaisuuksien tunnistamisen ja niiden muuntamisen vektoreiksi. (Kim;Jang;Park;& Choi, 2020) Kapselit neuroverkkona kokonaisuuksia yhdistämällä ylä- ja ala tason kapseleita dynaamisella reitityksellä. (Kim;Jang;Park;& Choi, 2020) Dynaamisen reitityksen tarkoituksena on kapselineuroverkossa oppia, mikä alempi kapseli liitetään ylempänä neuroverkossa olevaan kapseliin vertaamalla kapselien välisiä vektoreita keskenään. Piirteet vektoreista ja kokonaisuuksista ovat dynaamisessa reitityksessä yhdessä sovittuja eli sekä ylemmänettä alemman kapselin sopimia. Yhteisen sopimuksen takia luokituksen tarkkuus on parempi kuin tavallisissa konvoluutio neuroverkoissa.

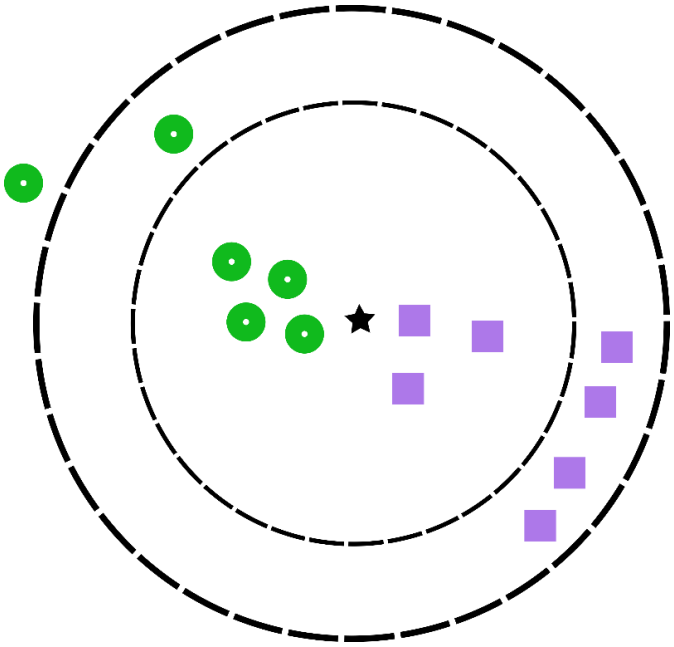
Kapselineuroverkot kehitettiin parantamaan konvoluutio neuroverkkojen heikkoja kohtia, joihin kuuluu objektien suuntaus ja avaruudelliset yhteydet. Objektien suuntaus ja avaruudelliset yhteydet tarkoittavat yleisten piirteiden esimerkiksi kasvojen piirteiden (silmät, nenä, suu) parempaa tunnistamista, vaikka ne esiintyisivät kasvoille epätyypillisissä paikoissa kasvoilla. Kapselit siis muuttavat avaruudessa sijaitsevan objektin vektoreita, mikäli sen paikka muuttuu. (Goldani, 2020)

Dynaamista reititystä ja sen toimintaa tekstin luokitteluun ovat tutkineen mm. Zhao, ym., Sabour, Frost ja Hinton. (Sara Sabour, 2017) (Wei Zhao, 2018) Kapselineuroverkkojen toimintaa tekstipohjaisella datalla ovat tutkineet myös Zhao Wei, Peng Haiyun, Eger Steffen, Cambria Erik ja Yang Min. Zhao, ym. tutkivat työssään kapselineuroverkon soveltuvuutta erilaisiin NLP sovelluksiin. Tarkkuudeksi Zhao, ym. saa parhaimmillaan 97,05 % tarkkuuden. Yksi Zhao, ym. mainitsemista sovelluksista on tekstin luokittelu useaan luokkaan. Usean luokan samanaikainen luokittelu eroaa tavallisesta luokittelusta siten, että yksi entiteetti datassa voi saada useamman kuin yhden mahdollisen luokituksen. Zhao, ym. tutkivat myös kysymys-vastaus-parien tutkimista kapselineuroverkoilla. Molemmat sovellukset osoittautuvat suoriutuvan työssä hyvin, ellei jopa erinomaisesti. (Zhao W. P., 2019)

Kapselineuroverkkoja on käytetty tekstin luokittelemisessa tunnistamaan muiden luokittelijoiden tapaan tunteita. Srivastava S., Khurana P. ja Tewari V. esittävät työssään kapselineuroverkkojen rakenteen, jonka tarkoitus on suodattaa vihapuhetta ja muuta sopimatonta, tekstimuodossa esiintyvää kieltä automaattisesti. (Srivastava S., 2018) Toisessa tutkimuksessa, jonka tekivät Wei Zhao and Jianbo Ye and Min Yang and Zeyang Lei and Suofei Zhang and Zhou Zhao kaselineuroverkko, jota käytetään mielipiteiden luokitukseen saavuttaa 93,8 % tarkkuuden. (Wei Zhao, 2018)

2.5.6 KNN

KNN-luokittelualgoritmi eli k lähintä naapuria menetelmä luokittelee datan kokonaisuudet etsimällä *k* lähintä alkioita käsittelyssä olevalle alkiolle kaikkien dokumenttien joukosta. Tämä perustuu hypoteesiin, jossa oletetaan dokumentin olevan muita samaan luokkaan kuuluvia dokumentteja lähellä vektoriavaruudessa. KNN-luokittelualgoritmin toiminta voidaan kuvata seuraavassa järjestyksessä: 1) KNN-luokittelualgoritmi laskee etäisyydet jokaisen datan pisteen ja tarkastelussa olevan pisteen välillä. Pisteistä valitaan k:n arvon mukaan. K:n arvo on KNN-luokittelualgoritmissa etäisyydeltään lähimpien alkioiden lukumäärä, joihin haluttua datapistettä verrataan (positiivinen kokonaisluku, joka on yleensä pariton, jotta tasapiste tilanteet ovat mahdottomia). KNN-luokittelualgoritmille tyypillistä on olla riippuvainen etäisyysfunktion sekä k-arvon valinnasta, tämän takia on tyypillistä valita k:n arvo välillä $[1, \dots, n]$, missä n on yhtä suuri kuin vektoriavaruuden kaikkien alkioiden lukumäärä.



Kuva 5 *Esimerkki KNN-luokittelualgoritmin k:n valinnasta*

Kuvasta 5 voidaan huomata k:n arvon valitsemisen vaikutus itse luokitustulokseen. Kuvassa on kaksi eri luokkaa: vihreät ympyrät ja violetit neliöt. K:n arvon ollessa 7, keskellä oleva musta tähti luokittuu luokkaan vihreä neliö. Toisessa tilanteessa, jossa k:n arvo on 12, keskellä oleva musta tähti luokittuu luokkaan violetti neliö. Tästä esimerkistä voimme havaita k:n arvon valinnan merkityksen itse luokitukseen.

KNN-luokittelualgoritmin voidaan luokitella kuuluvaksi valvottuihin menetelmiin, minkä takia KNN-luokittelualgoritmi vaatii toimiakseen opetusdataa, jonka perusteella etäisyys käsittelyssä olevaan alkioon voidaan tehdä. Toki voidaan todeta opetusvaiheen vain olevan vertailua vektoriavaruudessa olevien pisteiden etäisyyksien välillä. KNN-luokittelualgoritmi perustuu etäisyyksiä laskeviin funktioihin, joita on esimerkiksi euklidinen etäisyys, joka lasketaan kaavan 15 mukaan.

$$d(X,Y) = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}, \qquad (15)$$

missä X ja Y ovat vektoreita, x_i ja y_i vektorin ominaisuuksia ja k on vektoreiden ominaisuuksien lukumäärä. Muita mahdollisia etäisyysfunktioita ovat Hamming-, sekä Manhattan-etäisyys. Manhattan-etäisyys on esitetty kaavassa 16.

$$\sum_{i=1}^{k}|x_i - y_i|, \qquad (16)$$

missä x_i ja y_i vektorin ominaisuuksia ja k on vektoreiden ominaisuuksien lukumäärä. KNN-luokittelualgoritmia käytetään tekstinluokituksessa sen yksinkertaisuuden ja nopeuden takia. KNN-luokittelualgoritmin ollessa yksinkertainen, tutkimuksissa saadut tarkkuudet ovat alhaisempia kuin edellä mainituilla metodeilla. Pratama B. Y. ja Sarno

R. tulevat tutkimuksessaan lopputulokseen, jossa englanninkielisellä tekstipohjaisella datalla Naiivi Bayes -luokittelualgoritmin tarkkuus on 60,63 %, SVM-luokittelualgoritmin tarkkuus on 59,62 % ja KNN-luokittelualgoritmin tarkkuus on 59,8 %. (Pratama B. Y., 2015) KNN-luokittelualgoritmin on etäisyyksiin perustuva luokittelija, joten data, jossa dimensiot ovat suuret, voidaan olettaa KNN-luokittelualgoritmin suoriutuvan huonommin kuin muut edellä mainitut metodit. Uutisotsikoiden ollessa dimensioiltaan suuria, voidaan täten tehdä oletus, että KNN-luokittelualgoritmin suorituskyky on alhaisempi kuin muiden työssä esiteltyjen metodien.

2.6 Tekstin luokittelu ja uutisdata

Uutisdataan liittyvää tekstin luokittelua on tutkittu niin sisältö että otsikkotasolla. Barberá, Pablo, ym. kuvailevat käytännöllisiä ratkaisuja uutisartikkeleiden automaattiseen luokituksen. (Barberá, 2019) M. I. Rana, S. Khalid ja M. U. Akbar mainitsevat työssään uutisotsikoiden tutkimisen menetelmien olevan esitetty usean tutkijan toimesta. Rana, Khalid ja Akbar esittävät, että lyhyiden uutisotsikoiden etuna pitkiin uutisteksteihin on laskennallinen tehokkuus, sekä vähemmän esiin työntyvät väärät luokitukset, jotka johtuvat uutisotsikoiden pituudesta. Työssään he tutkivat luokitusta KNN, Naiivin Bayes, SVM, ANN sekä päätöspuiden avulla. (M. I. Rana, 2014)

Uutisdatalle tekstin luokittelua on tutkittu eri kielissä. Suosittuja kieliä ovat englanti, arabia ja kiina, jotka tarjoavat erilaisia haasteita kielen merkistön takia. (Einea;Elnagar;& Al Debsi, 2019) Arabian ja Kiinan kielen haasteina ovat esimerkiksi englannista poikkeavat merkistöt ja symbolikirjaimet. Englannin kielen haasteena on uudet tekstin luokittelun sovellukset, kuten keskustelufoorumien ja muiden kieliopillisesti erilaisten tekstirakenteiden käyttäminen.

Uutisdataa on käytetty tekstidatan luokittelussa myös tunteiden tunnistamiseen. Bhowmick, Plaban Kr, ym. tutkivat työssään, miten tunteiden esiintymistä voidaan luokitella uutisista. Tarkoituksena heillä Bhowmick ja Plaban Kr, ym. on tutkia, miten uutistekstit voidaan automaattisesti jakaa eri tunteita esiintyviin luokkiin, kuten ”suru” ja ”ilo”. Uutisdatan käyttäminen tapahtuu tässä työssä luokittamalla tunteet lukijan perspektiivistä eli minkälaisia tunteita erilaiset uutiset aiheuttavat lukijassa. (Bhowmick, 2009)

Toisesta suunnasta toteutetut tutkimukset uutisdatalle kuvailevat roskapostien ja väärennettyjen uutisten tunnistamista. Samaan kategoriaan voidaan luokitella kuuluvan myös huijauksien (*”hoax”*) tunnistaminen uutisdatan avulla. Prasetijo, Agung B., ym. tutkimuksen ideana käytetään uutisdataa ja tunnistetaan niiden sisältämät huijaukset käyttämällä uutissivustolla esiintyviä uutisia. (A. B. Prasetijo, 2017)

Uutisdatan käyttöä personoidussa uutisjakelussa on tutkittu Billsus D. ja Pazzani M. J. toimesta. He käyttävät uutisdataa ja tekstin luokittelua hyödyksi suositusjärjestelmässä, jossa yksittäisen käyttäjän mieltymyksien perusteella luodaan räätälöity päivittäinen uutiskatsaus. (Billsus, 1999) Vuonna 2001 Chan, C. H., Sun, A., & LIM, E. P. tutkivat automaattista internet-uutisten personoitua luokitusta. Luokittelumetodina työssä on valittu SVM-luokittelualgoritmi, sekä palveluun rekisteröityminen. Palveluun rekisteröityminen ei sinänsä ole metodi luokitella uutisdataa, mutta henkilökohtaisten mielenkiinnon kohteiden avulla uutisten suosituksia voidaan suodataa tarkemmin kuin pelkän SVM-luokittelualgoritmin avulla. (CHAN Chee-Hong, 2001) On huomattava, että uutisaiheisen tekstin luokitusta on tutkittu myös apuvälineenä sosiaalisen median tutkimuksessa. (Jotikabukkana, 2015) Samoin uutisdataa on hyödynnetty osana tutkimusta harvinaisen kielen tutkimuksessa. (Mangal, 2014)

Wermter Stefan ja Hung Chihli tutkivat työssään ”Reuters” uutisdatan luokitusta itse organisoituvilla menetelmillä. (Wermter, 2002) Työssään Wermter ja Hung käyttävät Reuters dataa vuosilta 1996 ja 1997, yhteensä 806791 uutisartikkelilla. Heidän tarkoitus työssään on käyttää itse organisoituvia menetelmiä sekä WordNet-toteutusta tekstin luokitukseen. (Wermter, 2002)

Nabamita Deb, Vishesh Jha, Alok K Panjiyar ja Roshan Kr Gupta tutkivat työssään erilaisten luokittelumetodien toimintaa uutisotsikoiden luokitukseen. Nabamita Deb ym. tulos kertoo sen, että Naiivi Bayes -luokittelualgoritmi toimii parhaiten heidän käsittelyssänsä olevan datan kanssa. Huomattavaa on, että SVM ja neuroverkon tarkkuus on sama työn aikana. Huomattava ero tässä työssä on datan määrässä, jossa 2225 riviä eli 2225 erilaista uutisotsikkoa luokituksineen käytetään opetukseen. Kyseinen rivimäärä on tavallista pienempi, mikäli verrataan uusimpiin tutkimuksiin aiheesta.

Tekstin kategorisointia eli tekstin luokitusta on tutkittu uutisten luokittelussa eri tasoilla, aina uutissisällöstä uutisotsikoihin. Seuraavassa luvussa esitellään tarkemmin tässä työssä käytettävä uutisdata ja ratkaisu, jolla suomenkielistä uutisdataa pyritään luokittamaan automaattisesti.

3. RATKAISU JA TOTEUTUS

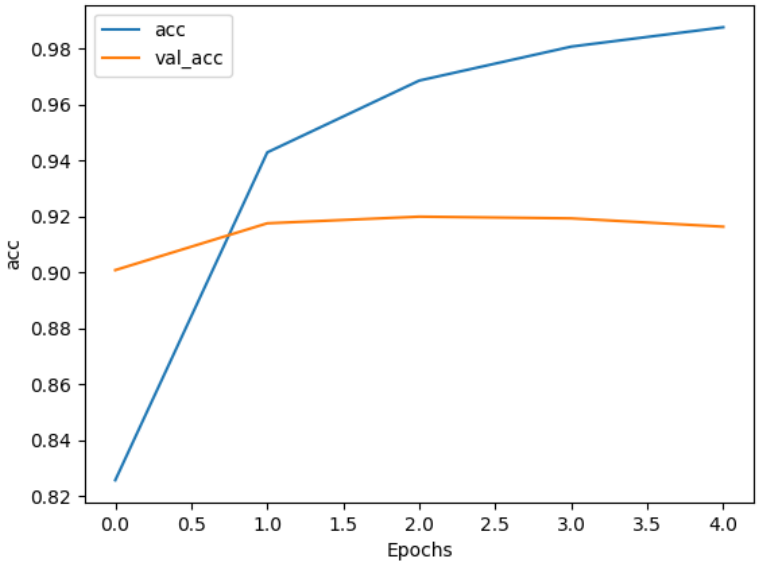
Työn tarkoituksena on tutkia erilaisten luokittelumenetelmien kykyä lyhyiden tekstien automaattisessa luokittamisessa. Työssä tutkitaan erityisesti Naiivin Bayes-, SVM-, Bi-LSTM- sekä BERT-luokittelumetodien toimintaa ja niiden kykyä luokitella suomenkielistä uutisdataa.

Kaikki testit suoritetaan Python ohjelmointikielellä, Windows 10 -käyttöjärjestelmässä. Datasettinä työssä käytetään Suomen Kielipankin oikeuksien alla olevaa STT uutisdataa. Data ja valitut luokittelumetodit esitellään tarkemmin seuraavissa aliluvuissa.

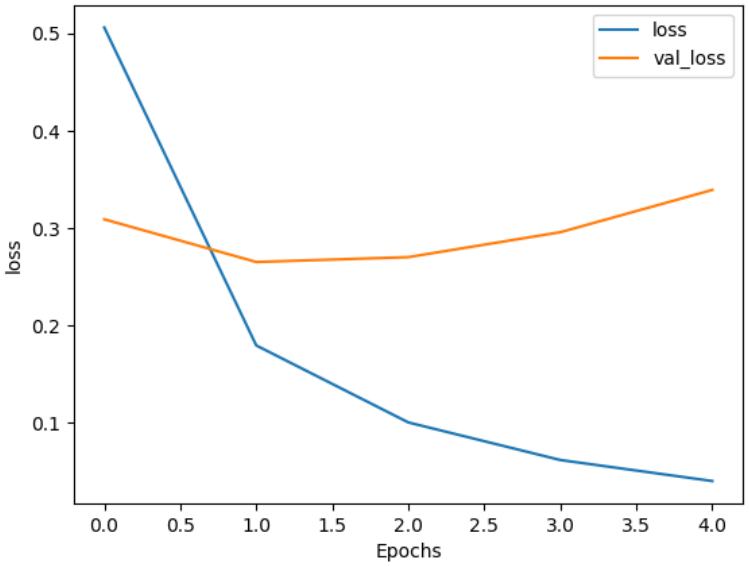
3.1 Esitutkimus

Datana työn alussa käytettiin YLE:n uutisaiheista korpusta. YLE:n korpus sisälsi kuitenkin luokituksen kannalta liian monimutkaisia ja tarkkoja uutisaiheita, joka oli osa valintaa, jossa kyseinen data jätettiin pois käytöstä. Data sisälsi myös uutisia, joiden kategoriaa ei ollut annettu sekä uutisia, joiden eivät sopineet määritelmään *”lyhyt teksti”*. YLEN:n data aiheutti myös muita haasteita, kuten merkistön poistamisen vaikeuksia, minkä takia se päätettiin jättää pois työstä ja tilalle etsittiin toinen data käytettäväksi.

Bi-LSTM-neuroverkon rakenteen ja sen parametrien vaikutusta tutkitaan luokitustulokseen. Tutkimuksessa parametrit valitaan käytettävissä olevien resurssien ja satunnaisvalinnan avulla. Parametrit, joita tutkitaan ovat: sanaston koko, näytteenoton koko sekä sulautuksen dimensio. Parametrien arvoja kokeillaan muokkaamalla eri yhdistelmiä parametreistä. Yhdistelmistä tuotetaan kuvaajat sekä tarkkuudesta, että hävikistä. Näitä kuvia vertaamalla valitaan optimaalisimmat arvot Bi-LSTM-neuroverkon parametrejä varten. Esimerkit kuvista löytyvät kuvasta 6 ja kuvasta 7. Kuvissa epochit eli kierrokset ovat vain 4 asti, aikaisen pysäytysparametrien takia. Aikainen pysähdysparametri selitetään tarkemmin luvussa 3.3.2.



Kuva 6 *Bi-LSTM accuracy-tuloksien kuvaaja*



Kuva 7 *Bi-LSTM validation loss -kuvaaja*

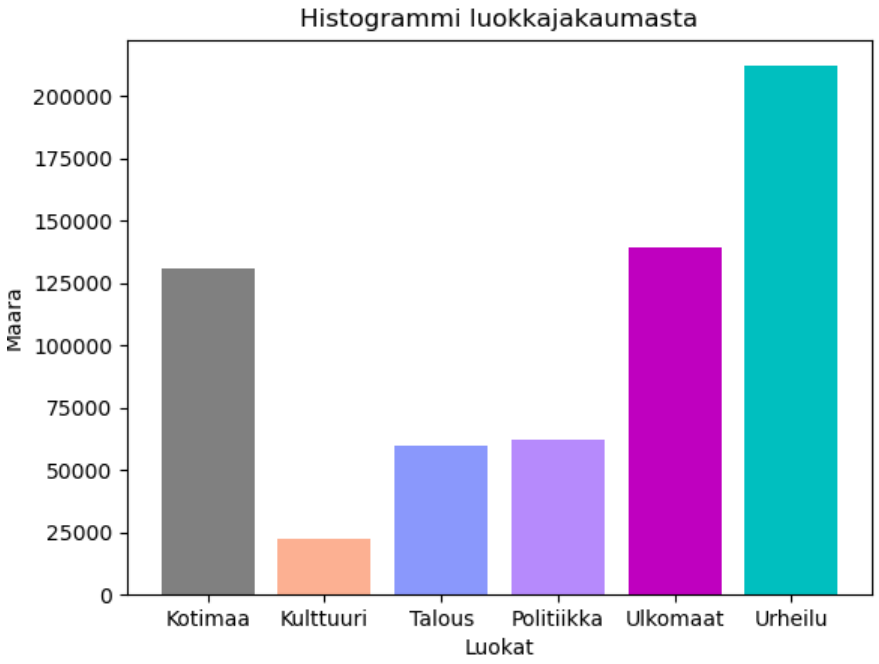
Optimaalisimmiksi arvoiksi valitaan sanaston kooksi 200000, näytteenottotaajuuden ollessa 32 ja sulatuksen dimension ollessa myös 32.

3.2 Datan käsittely

Tässä työssä datan kerääminen on suoritettu Kielipankin avustuksella. Kielipankin luvan alla vuosien 1992-2018 STT uutisten otsikoihin ja niiden luokituksin on saatu käyttöoikeudet. (Finnish News Agency Archive 1992–2018) Työssä kuitenkin käytetään vain

vuosien 2012–2018 STT uutisia, kyseisten vuosien samanlaisten merkintätapojen takia. Vuodet 1992–2011 on jätetty pois niiden erilaisten merkintätapojen sekä uutisotsikoiden vaihtelevien lause- ja virkemäärien takia.

Data edustaa ominaisuuksiltaan epätasaista jakaumaa. Tässä tapauksessa epätasainen jakauma tarkoittaa urheilu-uutisten korkeampaa lukumäärää sekä kulttuuriuutisten pientä lukumäärä suhteessa muihin luokkiin. Kuvasta 8 voidaankin todeta urheilu-uutisia olevan suhteessa enemmän kuin mitään yksittäistä muuta luokkaa. Tämän takia luokittelualgoritmien voidaan olettaa tunnistavan paremmin urheilu-uutisia, kuin muita uutisluokkia. Toinen huomio kiinnittyy kulttuuriuutisiin. Kulttuuriuutisen pieni määrä tarkoittaa sitä, että työssä on tärkeää huomioida myös kulttuuriin kuuluvien alkioiden tarkkuus. Luokkiin Talous, Politiikka, Kotimaa ja Ulkomaat kuuluvia uutisia on vaihteleva määrä, joka osaltaan vahvistaa epätasaisen jakauman vaikutusta luokitustulokseen.



Kuva 8 *Datan jakauma määrinä*

Kulttuuriuutisten määrä vaikuttaa esimerkiksi joidenkin luokittelualgoritmien luokitustarkkuuteen kyseisestä luokasta. Naiivi Bayes –luokittelualgoritmi on yksi luokittelija, jonka toimintaan vaikuttaa epätasainen datan jakauma ja erityisesti pienen datamäärän luokat tuottavat virheellisiä luokituksia Naiivi Bayes -luokittelualgoritmin tapauksessa.

Data työssä koostuu XML-tiedostoista, jotka sisältävät uutisotsikon, uutisluokan ja muuta metatietoa. Työn tarkoitus ei ole tutkia metatietoja, vaan keskittyä uutisotsikoihin ja niiden luokkiin, joten XML-tiedostot on purettu tekstitiedostoiksi, joissa XML-standardin merkistö sekä ylimääräiset metatiedot on poistettu. Tuloksien yhtenäistämiseksi kaikki

sanat datassa on muunnettu sisältämään vain pieniä kirjaimia sekä jokainen uutisotsikko ja luokka pari on tallennettu omalle rivilleen.

XML-tiedostosta data luetaan tekstitiedostoon kuvan 9 mukaisesti, jossa yhden rivin muoto on seuraavanlainen *#Luokannimi uutisotsikko*. Uutisotsikon ja luokannimen välissä on tabulaattori myöhemmän prosessoinnin helpottamiseksi. Työn kannalta on valittu jättää lemmatisoinnin tarkkuus testaamatta muilla kuin SVM- ja Naiivi Bayes -luokittelijoilla.

```
#Kotimaa:	rakentajia ohjataan ympäristöä säästäviin energiamuotoihin
#Kotimaa:	kriisin kokeminen voi tuoda paremman elämän
#Kotimaa:	nöyryytys on miehille vaikeaa, menetykset naisille
#Kotimaa:	nettideittailu on arkipäiväistynyt ja tullut jäädäkseen
#Kotimaa:	vammainen äiti vierastaa sankarin viittaa
#Kotimaa:	unelmien vaatekaappi ei synny vahingossa
#Kotimaa:	nettideittailijat edustavat kahta koulukuntaa
#Kotimaa:	näitä seuranhakusivustoja suomalaiset käyttävät
#Kotimaa:	maksulliset nettideittipalvelut kasvattaneet suosiotaan
#Kotimaa:	maksulliset nettideittipalvelut kasvattaneet suosiotaan
#Kotimaa:	joka kymmenes korttipetoksesta epäilty romanialainen
#Kotimaa:	romanialaisepäiltyjen osuus omaisuusrikoksissa laskussa
#Kotimaa:	joka kymmenes korttipetoksesta epäilty romanialainen
#Kotimaa:	mikä e-luku?
#Kotimaa:	talouden epävarmuus ei enää verota suunnittelukilpailuja
#Kotimaa:	järjestä kirppis ystävien kanssa
#Kotimaa:	pukeutumisvalmentaja: moni shoppaa suruun
#Kotimaa:	valokuvakilpailujen säännöissä riittää tarkennettavaa
#Kotimaa:	päämiehellä ja edunvalvojalla voi olla eri näkemys rahankäytöstä
#Kotimaa:	muista nämä, jos osallistut
#Kotimaa:	valokuvakilpailujen säännöissä usein epäselvyyksiä
#Kotimaa:	valokuvakilpailujen säännöissä usein epäselvyyksiä
#Kotimaa:	edunvalvoja toimii päämiehensä parhaaksi
```

Kuva 9 *Tekstitiedoston rakenne datasta*

Työssä käytetty data on jaettu suhteessa 70 % opetukseen ja 30 % testaukseen, käyttämällä sklearn-kirjaston toteutusta opetus- ja testidatan jakamiseen. Jako 70 % ja 30 % on perusteltu antamalla testimateriaalille noin kolmasosa saatavana olevasta materiaalista. Opetukseen ja testaukseen käytettävä materiaali on sekoitettu, joka tarkoittaa sitä, että kaikkien vuosien dataa voi esiintyä sekä opetus- että testidatan jaoissa.

Työssä käytettävän data edustaa epätasaista jakaumaa, mutta luokkien tasoittamiseksi ei ole käytetty resursseja. Voidaan kuitenkin todeta datan tasoittamisen olevan yksi tulevaisuuden jatkotutkimuskohde.

Jokaiselle algoritmille on käytetty sen vaatimaa metodia datan esiprosessoinnista. SVM ja Naiivi Bayes kanssa käytetään TF-IDF-muunnosta, mutta BiLSTM ja BERT käyttävät

omaa muunnostaan datalle. TF-IDF-muunnos tässä tapauksessa muodostaa uutisotsikoista vektoreita, joissa jokainen sana on muunnettu vastaamaan tiettyä vektoria. Muut käytettävät muunnokset esitetään tarkemmin seuraavissa luvuissa.

3.3 Algoritmit

Seuraavissa alaluvuissa esitetään työssä käytettävät luokittelumetodit, luokittelumenetelmiä käyttävät kirjastot sekä luokittelumetodien parametrit. Kaikki algoritmit ovat valittu helppokäyttöisyyden ja saatavuuden mukaan.

3.3.1 SVM

Luvussa 2.5.1 todetaan SVM-luokittelualgoritmin onnistuvan hyvin luokitteluun liittyvissä tehtävissä. SVM-luokittelualgoritmin, varsinkin lineaarisen SVM-luokittelualgoritmin suorituskyky on todettu hyväksi kirjallisuuskatsauksen pohjalta, mikä on peruste käyttää sitä myös tämän työn toteutuksessa.

SVM toteutuksena käytetään *sklearn*-kirjaston toteutusta *svm.LinearSVC*. SVM-luokittelualgoritmin kanssa käytetään TF-IDF-muunnosta ja kokeillaan sanojen lemmatisoinnin sekä duplikaattien poistamisen vaikutusta itse luokituksen tarkkuuteen. Duplikaattien poistamiseen käytettävä tekniikka on pitää kirjaa jo nähdyistä uutisotsikoista. Mikäli sama otsikko esiintyy toistamiseen, poistetaan uudelleen esiintynyt kerta pois otsikkolistasta. TF-IDF muunnoksen tekemiseen käytetään *sklearn.feature_extraction.text TfidfVectorizer* -toteutusta. TfidfVectorizer suorittaa sekä tokenisoinnin, että TF-IDF muunnoksen samalla funktiolla.

Svm.LinearSVC-parametrit on jätetty oletuksilla, jotka ovat dokumentaation mukaan: $c = 1.0$, $penalty = L2$. Oletusarvoja on käytetty siitä syystä, että ne tarjoavat helpon ja nopean tavan suorittaa luokitteluja, sekä yhtenäistävät toteutusta.

TF-IDF toteutuksen parametrit ovat kaikki vakio arvoilla, mikä tarkoittaa sitä, että ennen tokenisointia TF-IDF toteutus muuntaa sanat pieniksi kirjaimiksi, sekä *min_df* ja *max_df* ovat arvoilla 1.0 ja 1. Kaikki muut parametrit ovat arvolla *"None"* tai *"False"* eli niitä ei oteta huomioon.

3.3.2 Bidirectional LSTM

Työssä käytettävä BiLSTM-neuroverkko on toteutettu valitsemalla kirjallisuuskatsauksessa esiintyvät parhaiten suoriutuvat ja tutkimuskysymystä vastaavat toteutukset. Kirjallisuuskatsauksen myötä valittu BiLSTM-neuroverkko on saavuttanut aiemmissa tutkimuksissa huippuluokan tuloksia ja tämän takia se on valittu osaksi käytettäviä luokittelumetodi.

Työssä käytettävän BiLSTM-neuroverkon rakenne on seuraavanlainen. Ensimmäinen kerros on embedding-kerros. Embedding- eli upotuskerroksen tarkoitus on muuttaa tekstin muuttujat (tässä tapauksessa sanat) määrätyn pituisiksi vektoreiksi. Kerroksen tavoite on mahdollistaa koneopittava muoto datalle. Neuroverkon toinen kerro on Bi-directional layer eli kaksisuuntainen kerros. Kaksisuuntainen kerros mahdollistaa datan kaksisuuntaisen syötön LSTM-neuroverkkorakenteelle, jonka toimintaa on esitetty tarkemmin aiemmassa luvussa. Kolmas ja neljäs kerros ovat keras-kirjaston dense toteutuksia. Kolmas kerros on aktivointikerros, jossa aktivointifunktiona käytetään ‘relu’. Relu aktivointifunktiona tarkoittaa sitä, että sisääntulo syötetään suoraan eteenpäin, mikäli se on mahdollista. Tämä tarkoittaa myös sitä, että sisääntulo muuttuu arvoon 0, mikäli sisääntulo ei ole suurempi kuin 0. Neljännen kerroksen aktivointi on ‘softmax’, jonka tarkoitus on tiivistää neuroverkon ulostulo haluttuun määrään luokkia.

BiLSTM-neuroverkolle datan muuntamiseen ja piirteiden valitsemiseen ei ole käytetty TF-IDF-muunnosta. Bi-LSTM kuten LSTM kuuluvat takaisinkytkettävien neuroverkkojen ryhmään. Takaisinkytkettävyyden johdosta Bi-LSTM-neuroverkon idea on kaksisuuntainen takaisinkytkettävyys, jonka ansioista data syötetään neuroverkolle molemmista suunnista (input-output sekä output-input). Toteutuksen kannalta takaisinkytkettävyys ei aiheuta muutoksia datan rakenteeseen. Työssä Bi-LSTM neuroverkko on toteutettu käyttämällä Keras Python – kirjastoa. Tämän kirjaston ansiosta itse neuroverkon toteuttaminen on nopeaa ja yksinkertaista runsaan dokumentaation sekä käytettävyyden helppouden ansiosta.

BiLSTM vaatii opetus- ja testitdatan muuntamiseen keras-kirjaston pad_sequences funktiota, jonka tarkoitus on muuntaa data vektorimuotoon. Parametreinä funktiolle on annettu maksimipituudeksi 200, joka tarkoittaa vektorien pituutta. Täyte sekvenssinä varten parametri padding arvo on ”post”, joka tarkoitaa, että sekvenssien täyttäminen suoritetaan vasta sekvenssin jälkeen. Samaan tapaan ”truncating”-parametrin arvo on ”post”, joka tarkoittaa sekvenssin loppupäähän tehtävää katkaisua.

BiLSTM luokittelijalle voi asettaa toteutuksessa ehdon, jonka täyttyessä opetus lopetetaan ennen, kuin kierrosten määrä on loppunut. Työssä käytettävä ehto on validation loss eli tarkistus tarkkuuden seuranta.

3.3.3 Naiivi Bayes

Kolmantena luokittelijana työssä on Naiivi Bayes –luokittelija. Naive Bayes luokittelija on generatiivinen luokittelija, joka tarkoittaa, että kukin luokka pystyy mallintamaan havaintojen todennäköisyyden. Naive Bayes –luokittelija on toteutettu sklearn (scikit-learn) -kirjastolla. Sklearn-kirjaston toetutuksista valitaan Multinomia Naiivi Bayes -toteutus,

joka osoittautui kirjallisuuskatsauksessa sopivaksi luokitusongelmaan, jossa luokitteluongelma ei ole binäärinen ja datan jakauma on epätasainen.

Multinomial Naiivi Bayes parametrit ovat valittu olemaan vakiot. Vakio parametrejä tässä tapauksessa ovat *alpha*, *fit_prior*, sekä *class_prior*. Alpha tarkoittaa laplace/lidstone tasoitusta, jonka vakio arvo on 1. *Fit_prior* tarkoittaa sitä, oppiiko Naiivi Bayes -luokittelualgoritmi varhaisempia todennäköisyyksiä ja sen vakio arvo on ”True”. Viimeinen parametri on class_prior, joka tarkoittaa varhaisten todennäköisyyksien opettamista jokaiselle luokalle. Class_prior parametrin vakio arvo on ”None”. Naiivi Bayes tapauksessa uutisotsikot on muunnettu TF-IDF muunnoksella sopivaan muotoon, kuten luvussa 3.2 ja seuraavassa kappaleessa esitetään.

TF-IDF muunnoksen tekemiseen käytetään sklearn.feature_extraction.text TfidfVectorizer -toteutusta. TfidfVectorizer suorittaa sekä tokenisoinnin, että TF-IDF muunnoksen samalla funktiolla. TF-IDF toteutuksen parametrit ovat kaikki vakio arvoilla, mikä tarkoittaa sitä, että ennen tokenisointia TF-IDF toteutus muuntaa sanat pieniksi kirjaimiksi, sekä *min_df* ja *max_df* ovat arvoilla 1.0 ja 1. Kaikki muut parametrit ovat arvolla ”None” tai ”False” eli niitä ei oteta huomioon.

3.3.4 Ktrain BERT

Neljäntenä luokittelijana työssä käytetään Ktrain Python 3 -kirjaston tarjoamaa rajapintaa BERT-malleille. (Maiya, 2020) BERT perustuu luokittelijana muuntajiin (transformers). BERT-malli on valittu sen tarkkuuden ja uutuusarvon takia.

Ktrain-kirjasto, jonka rajapinnan kautta BERT-algoritmi on otettu käyttöön, tarjoaa päällysteen monen muun kirjaston, kuten Tensorflow ympärille. Ktrain tarkoitus on tuottaa helposti lähestyttävä ratkaisu viimeisimpien koneoppimisalgoritmien käyttämiseen. Ktrain sisältää toteutukset moneen eri ongelmaan, kuten kuvantunnistus, NER prosessointi ja tekstin luokittelu. Ktrain BERT toteutusta varten data jaetaan opetus- ja testi materiaaleihin. Opetusmateriaalit on jaettu todellisiin uutisluokkiin sekä uutisotsikoihin, joista jokaiselle uutisotsikolle tulee olla valmiiksi annettu uutisluokka. BERT-mallia varten data käsitellään Ktrain-kirjaston tarjoamalla rajapinnalla muuntajien esiprosessointia varten.

Ktrain-kirjaston avulla BERT-algoritmiä ajetaan 4-4-2 epochin sarjoissa. Tärkeää on muistaa, että BERT on suorituskykyä vaativa algoritmi ja siksi ajaminen tapahtui vain maksimissaan 4 epochia kerralla. Työssä ajetaa 4 epochin sekä 2 epochin sarjat, jonka avulla saadaan tuloksia myös mallien jatko-opettamisesta, että suorituskyvystä reaaliajassa. Samaan kategoriaan kuuluu batch-size-parametrin valinta. Parametri valitaan

kokoon 16, sillä se on suorituskyvyltään ja resursseiltaan paras saatavana olevaan laitteistoon nähden.

Ktrain-kirjaston tarjoaa rajapinnan myös HuggingFace-kirjaston valmiille BERT-malleille. Työssä malleina on käytetty *multilingual-bert-base-uncased* -mallia, sekä Turun Yliopiston tuottamaa FINBERT-mallia. (Pyysalo, 2019) FINBERT-malli on opetettu valmiiksi suomenkielisellä YLE ja STT uutisilla vuodesta 1992 vuoteen 2018 saakka, sekä keskustelufoorumi Suomi24 keskusteluilla vuodesta 2001 vuoteen 2017 saakka. Dataa FINBERT-malliin on hankittu myös internet hauilla. FINBERT käyttää opetusvaiheessaan samaa dataa, kuin tässä työssä on käytetty, joten sitä käytetään vain vertailukohtana *multilingual-bert-base-uncased* -mallille, eikä FINBERT-mallin tuloksia voida sen takia käyttää suoraan muiden algoritmien ja metodien väliseen vertailuun.

Multilingual-bert-base-uncased -malli koostuu 100 kielestä ja niistä kerätyistä Wikipedia-artikkeleista. Kielet on valittu Wikipedian top-100-listasta, johon on lisätty 100 suurinta kieltä Wikipediassa. Suurin kieli tarkoittaa kieltä, jolla on kirjoitettu laajimmin Wikipedia-artikkeleita. Tämän takia edellä mainitun BERT-mallin voidaan olettaa toimivan monikielisissä luokitustehtävissä, jotka sisältävät vaihtelevaa sanastoa.

3.4 Tarkkuuden mittarit

Luokittelijan tarkkuuden mittaamiseen käytetään sklearn-kirjaston toteutusta *accuracy_score* ja *classification_report*. Accuracy_score laskee tarkkuuden kaavan 17 mukaan.

$$accuracy = \frac{Oiken\ luokiteltujen\ alkoiden\ lukumäärä}{Kaikkien\ alkioiden\ lukumäärä}, \tag{17}$$

tämä metodi on yleisesti käytetty mittaamaan luokittelun tarkkuutta. Työssä on käytetty tarkkuuden mittareina myös sklearn-kirjaston *classification_report* -toteutusta, joka tarjoaa numeeriset tarkkuuden mittarit myös F1-pisteytykselle, precisionille ja recallille. Precision tai suomeksi täsmällisyys/tarkkuusaste lasketaan kaavan 18 mukaan,

$$precision = \frac{tp}{tp+fp}, \tag{18}$$

missä tp (*true positive*) tarkoittaa oikein luokiteltujen alkioiden lukumäärää ja fp (*false positive*) tarkoittaa väärin oikeiksi luokiteltujen alkioiden lukumäärää. Recall voidaan suomentaa tarkoittamaan saantia eli osumien määrää. Saanti lasketaan kaavan 19 mukaan,

$$recall = \frac{tp}{tp+fn}, \tag{19}$$

missä fn (*false negative*) tarkoittaa puuttuvia oikeita alkioita ja tp tarkoittaa oikeiden luokiteltujen alkioiden lukumäärää. Samaan tapaan F1-pisteytys voidaan laske kaavan 20 mukaan,

$$F1 = 2 * \frac{precision * recall}{precision + recall} = \frac{tp}{tp + \frac{1}{2}(fp + fn)}, \quad (20)$$

missä tp tarkoittaa oikeiden luokiteltujen alkioiden lukumäärää, fp tarkoittaa väärin oikeiksi luokiteltujen alkioiden lukumäärää sekä fn tarkoittaa puuttuvia oikeita alkioita. Näiden tarkkuuden mittareiden avulla selvitetään luokittelumetodien suorituskyky.

4. TULOKSET

Taulukosta 1 nähdään Naiivin Bayesin sekä SVM suoriutuminen suomenkilisten uutisotsikoiden luokituksesta. Rivi ”*TF-IDF”* tarkoitta sitä, että stoppisanat on poistettu TF-IDF muunnoksen tekevän funktion avulla. Seuraava rivi ” stoppisanat sisällä*”* tarkoittaa sitä, että stoppisanat ovat vielä tekstissä eli niitä ei ole poistettu uutisotsikoista. Kolmas rivi tarkoittaa sitä, että stoppisanat on poistettu, mutta vertaamalla jokaista tekstivirran sanaa stoppisanalistaan ja poistamalla stoppisanat tekstimuotoisista alkioista. Neljäs ja viimeinen rivi tarkoittaa sitä, että stoppisanoja ei ole poistettu, mutta useasti esiintyvät uutisotsikot on poistettu. Useasti esiintyvien uutisotsikoiden poistaminen tarkoittaa sitä, että tietty uutisotsikko voi esiintyä vain kerran koko opetusmateriaalissa, joka aiheuttaa alkioiden määrän vähenemisen datassa.

Parhaan tuloksen taulukon 2 mukaan sai Ktrain-kirjaston BERT toteutus. BERT toteutuksessa kuitenkaan tarkkuus ei noussut yli 94 %, mikä tarkoittaa, että SVM on vain 2 prosenttiyksikköä huonompi kuin BERT kyseisessä tehtävässä. Samanlainen päätelmä voidaan tehdä myös verratessa BERT-mallia ja BiLST-neuroverkkoa, joiden ero on keskenään vain noin 3 prosenttiyksikköä. BERT-mallin tarkkuutta voidaan nostaa käyttämällä FINBERT-pohjamallia, jolloin tarkkuus saadaan nostettua 2 prosenttiyksikköä alkuperäisestä *’multilingual-bert-base-uncased’* mallista. Naiivin Bayes -luokittelijan tarkkuus ei nouse samalle tasolle muiden luokittelumetodien kanssa. Havainto selitetään tarkemmin seuraavissa kappaleissa.

SVM-luokittelualgoritmin confusion-matriisia tarkastelemalla, kuvaa 10. Confusion-matriisi on normalisoitu, joka näyttää jokaista luokkaa koskevan tarkkuuden normalisoituna. Matriisista voidaan huomata isoimman datamäärän sisältävän luokan oppivan luokituksen parhaiten. Toisaalta luokat, joissa on vähemmän dataa luokittuvat huonommin. Tämä voidaan selittää sillä, että epätasaisesti jakautuvan datan luokittaminen on lineaarisilla luokittelijoilla vaikeampaa kuin tasaisen jakauman omaavalla datalla.

Tutkittaessa väärin menneitä alkioita, datasetistä ja aiheesta selvisi muutamia asioita. Datasta löytyi otsikoita, jotka esiintyivät useampaan kertaan. Samalla selvisi myös uutisluokitusten olevan erilaisia. Yksi työn tutkimuksen kohde onkin selvittää, mitä vaikutusta on useasti esiintyvien uutisotsikoiden poistamisella tarkkuuteen ja itse luokitusprosessiin.

Taulukko 1 Naiivi Bayes ja SVM tarkkuudet eri metodeilla

	Naive Bayes	**SVM**
TF-IDF	86.65	92.31
stoppisanat sisällä	86.52	92.34
stoppisanat tokeneista	86.65	92.31
Duplikaatit	81.83	-

Useasti esiintyvien otsikoiden poistamisen jälkeen kaikkien luokittelijoiden tarkkuus laski muutamalla prosenttiyksiköllä. Tämä tarkoittaa sitä, että useasti esiintyvät otsikot auttavat luokitusta, eivätkä huononna tulosta. Voidaankin todeta useasti esiintyvien otsikoiden lisäävän opetusdatan määrää ja tämän takia vaikuttavan luokitustuloksen tarkkuuteen.

Tämän seurauksena työssä on tutkittu väärin menneitä alkioita ja SVM tapauksessa niistä löytyi yhteneviä piirteitä.

Taulukko 2 BERT ja BiLSTM tarkkuus, precision, recall ja F1-pisteytys

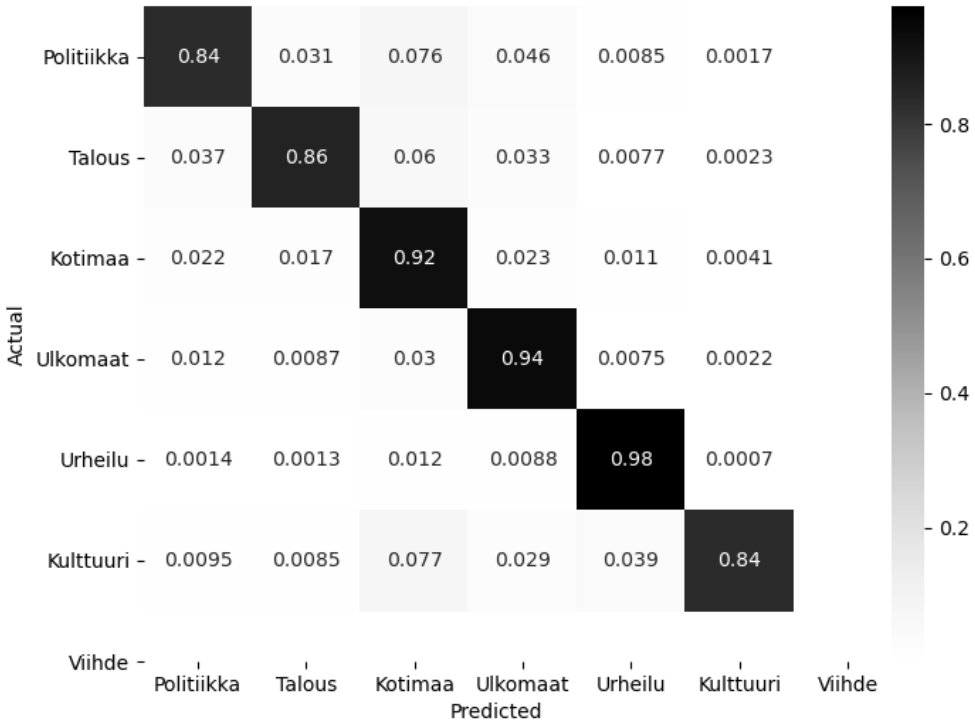
	BERT ('bert-base-multilingual-uncased')	**BiLSTM**
accuracy	94.00	~91.00
Precision - painotettu keskiarvo	94.00	-
Recall - painotettu keskiarvo	94.00	-
F1-pisteytys - painotettu keskiarvo	94.00	-

Ensimmäinen huomio kiinnittyi väärin luokiteltujen alkioiden sanoihin. Useassa väärin luokitellussa uutisotsikossa sanan keskeisin merkitys nähdään lauseyhteydestä. Tässä lauseyhteys tarkoittaa esimerkiksi uutisointia mm-kisabudjetin kasvattamisesta, joka luokitukseltaan kuuluu talouteen, mutta on väärin luokiteltu SVM-algoritmin tapauksessa

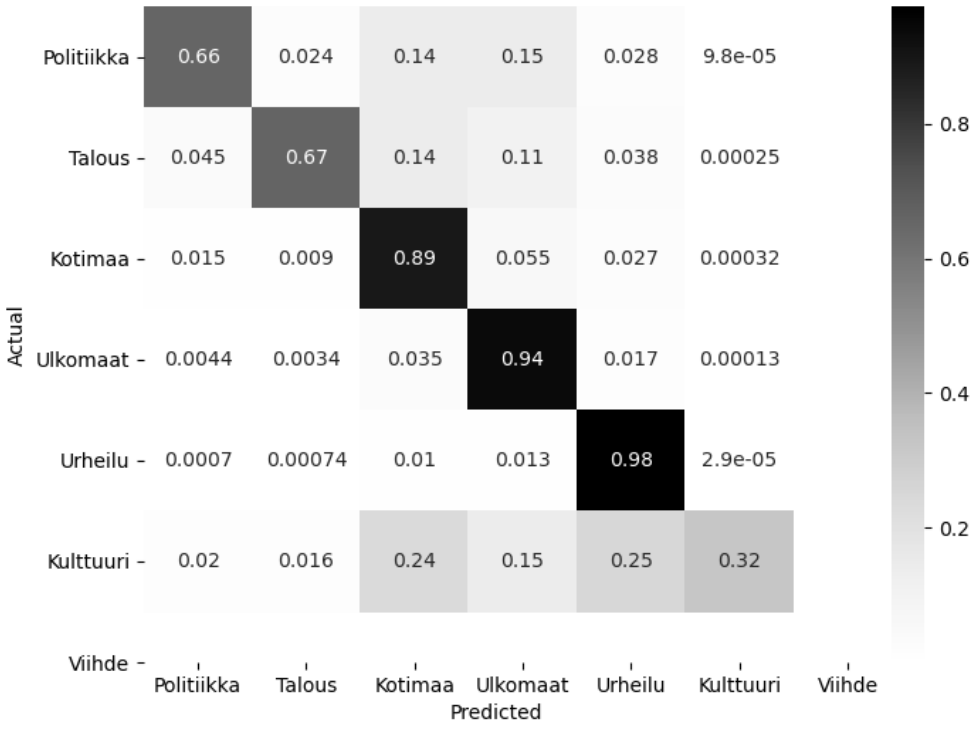
urheiluksi. Toinen vaihtoehto väärin luokitelluissa alkioissa on mahdollisuus kuulua myös toiseen luokkaan. Tarkasteltaessa väärin menneittä alkioita voidaan huomata myös työn tekijän luokittavan osan väärin menneistä myös useampaan kuin yhteen luokkaan. Tästä syystä voidaankin todeta luokittelualgoritmin mahdollisesti toimivan hyvin tosi elämässä, missä luokkien rajat eivät ole tarkasti määritellyt ja mahdollisuus on luokitella uutisotsikot useampaan kuin yhteen luokkaan.

SVM-luokittelualgoritmin tapauksessa väärin luokiteltuja alkioita yhdistää sekä lauseyhteys, että erisnimet. Erisnimien, kuten *”Cheek”* ja *”Robin”* tapauksessa SVM-luokittelualgoritmi luokittelee uutisen kulttuuriksi, vaikka uutisen oikea luokka on kotimaan uutiset. Tämä johtuu luultavimmin erisnimien vaikutuksesta luokitukseen ja siitä, että SVM on luokittelijana lineaarinen, joten se ei ota huomioon esimerkiksi lauserakenteita ja sanojen välisiä yhteyksiä. Naiivi Bayes, kuten SVM luokittele uutisotsikoita väärin tilanteissa, joissa otsikko sisältää tietylle luokalle kuuluvan kuvaavan sanan tai erisnimen. Tietenkin voidaan myös huomata luokittelun epäonnistumisen johtuvan tavallisesta luokitteluvirheestä.

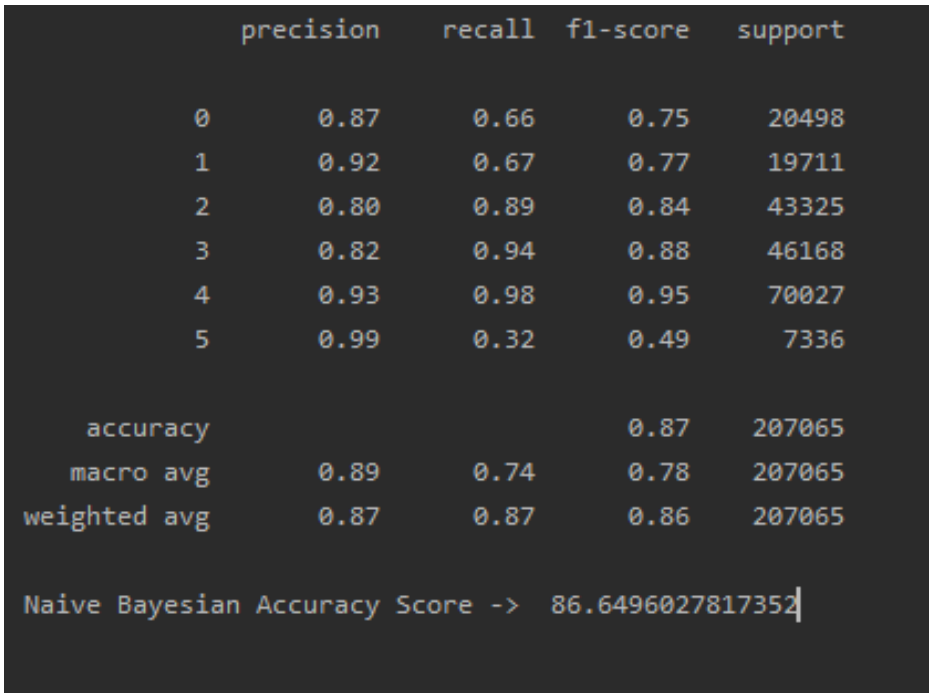
Naiivin Bayes –luokittelijan kohdalla voidaan huomata väärin menneissä alkioissa Naiiville Bayesille ja Bayes-teorialle tyypillinen ongelma, opetusmateriaalin epätasainen jakauma. Epätasainen jakauma opetusmateriaalissa aiheuttaa sen, että pienimmän opetusmateriaalin luokan alkiot luokittuvat väärin. Saatu tulos voidaan havaita myös kuvasta 11, joka esittelee Naiivin Bayes -luokittelualgoritmin konfuusio-matriisin. Konfuusio-matriisista voidaan havaita luokkien politiikka, talous sekä kulttuuri tarkkuuden olevan alhaisempi kuin muiden luokkien. Epätasainen normalisoitu tarkkuus konfuusio-matriisista johtuu edellä mainitusta epätasaisesta jakaumasta. Naiivin Bayes -luokittelijan tapauksessa Frank E. sekä Bouckaert R. toteavat tutkimuksessaan saman tuloksen. (Frank E., 2006) Työn tarkoituksena ei ole pohtia syitä ja ratkaisua epätasaiselle jakaumalle tarkemmin, vaikka se esitetäänkin havaintona. Esimerkki classification_report-työkalun käytöstä löytyy kuvasta 12. Möys classification_report-työkalun avulla voidaan havaita sama kuin Naiivi Bayes -luokittelualgoritmin konfuusio-matriisista.



Kuva 10 *SVM-luokittelualgoritmin confusion matrix*



Kuva 11 *Naiivi Bayes -luokittelualgoritmin confusion matrix*



	precision	recall	f1-score	support
0	0.87	0.66	0.75	20498
1	0.92	0.67	0.77	19711
2	0.80	0.89	0.84	43325
3	0.82	0.94	0.88	46168
4	0.93	0.98	0.95	70027
5	0.99	0.32	0.49	7336
accuracy			0.87	207065
macro avg	0.89	0.74	0.78	207065
weighted avg	0.87	0.87	0.86	207065

Naive Bayesian Accuracy Score -> 86.6496027817352

Kuva 12 *Esimerkki työssä käytettävästä classification_report-työkalusta*

Väärinluokitellut alkiot ovat alkioita, jotka luokittelualgoritmi, tässä tapauksessa BERT, SVM, Naiivi Bayes sekä Bi-LSTM ovat luokitelleet, mutta alkioin oikea (“true”) luokka on eri kuin luokittelualgoritmin luokka. Väärinluokitelluista alkioista voidaankin todeta vaikeiden tapausten sisältävän informaatiota, jonka luokittelualgoritmin pitäisi tietää etukäteen. Tällaista etukäteen tiedettyä tai opetettua informaatiota on esimerkiksi henkilöiden tai tapahtumien nimet sekä muut erisnimet.

BERT-mallin väärin menneitä alkioita tarkastelemalla huomataan, että luokitusvirhe voi johtua myös kontekstista sekä mahdollisuudesta uutisotsikolle kuulua useampaan kuin yhteen luokkaan. Useampaan kuin yhteen luokkaan kuulumisella tässä tarkoitetaan sitä, että ihminen voisi myös luokitella alkion kuuluvan kahteen tai useampaan luokkaan. Esimerkiksi uutisotsikko ”putoustähdet tekivät mukavan tilin – kososen tulot 120 000”, oikea luokka on kulttuuri, sillä kyseessä on näyttelijät sekä TV-sarja Putous. BERT-mallin tapauksessa edellä mainittu otsikko on virheellisesti luokiteltu talousuutiseksi, sillä otsikko sisältää sekä numeroita että sanat ”tilin” ja ”tulot”, jotka voidaan ymmärtää kuuluvan talousuutisiin.

Toisaalta voidaan huomata, että BERT ei oikeasti ole 100 % tarkka luokituksissaan. Esimerkiksi uutinen *”abloy kiistää yleisimmän lukkotyypin turvallisuusuhan”* on luokiteltu BERT-mallin mukaan luokkaan ”Kulttuuri”, vaikka oikea luokka on ”Kotimaa”. Virheen ei lauseen rakenteen tai sanaston perusteella voida luokitella kuuluvan edellä mainittuihin

tapauksiin, joten se luokitellaan oikeaksi virheeksi. Esimerkiksi toisesta virheestä voidaan katsoa kuuluvan otsikko *”pknfanit kiittelivät vetoa monella riittäisi kantti samaan”*. BERT luokittelee edellä mainitun otsikon luokkaan ”Urheilu”, vaikka oikea luokka on ”Kulttuuri”. Otsikko voidaan katsoa täysin vääräksi, sillä BERT on painottanut luokkaan ”Urheilu” kuuluvia sanoja, kuten ”vetoa” ja ”-fanit”. Tästä voidaan päätellä uutisaineiston vaihtelevuuden ja luokkatyyppisten sanojen esiintymisen vaikuttavan luokittelutulokseen.

Vertailtaessa BERT-algoritmia, sekä muita työssä käytettyjä tekstin luokitteluun soveltuvia algoritmeja, BERT on ehdottomasti tarkin. BERT:n tarkkuus kuitenkin näkyy sen koulutusajassa. BERT mallin koulutusaika neljällä kierroksella on noin 4 tuntia NVIDIA RTX Quattro 6000 -näytönohjaimella. SVM, sekä Naiivin Bayes -luokittelijan koulutusnopeus on huomattavasti nopeampi. Yrityskäytössä opetusaika ja projektin aikajana on kuitenkin yleensä mitoitettu sopivaksi, joten pidempää käyttöä ajatellen BERT-algoritmin käyttäminen sopii yrityskäyttöön paremmin. Nopeaan käyttöön ja väliaikaisiin ratkaisuihin SVM- ja Naiivi Bayes -luokittelualgoritmi sopivat hyvin.

Tarkkuus ei parhaimman luokittelijan (BERT) tapauksessakaan nouse yli 94 %. Voidaankin pohtia, onko kyseinen tarkkuus riittävä tämän työn tilanteessa tai yleisessä yrityskäytössä. Tämän työn kannalta tarkkuus 94 % osoittautuu riittäväksi. Tarkoituksena työssä on tutkia, minkälaisia keinoja on luokitella suomenkielistä dataa uutisotsikoiden muodossa. Tarkkuuden mittarina ei tässä ole saavuttaa 100 % tarkkuutta vaan tutkia ja vertailla eri menetelmiä. Vertailulla tässä tapauksessa tarkoitetaan tutkimusta, sopivatko työssä käytetyt metodit suomenkielisten uutisotsikoiden automaattiseen luokittamiseen. Tutkimiseen ja vertailemiseen käytettävä tarkkuus on vain osa mittareita, joilla metodien välistä vertailua voidaan tehdä. Aiemman oletuksen takia voidaankin tarkkuuden 94 % huomata olevan riittävä parhaimmaksi tarkkuudeksi, kun tarkoituksena on kartoittaa menetelmiä, suomenkilisen tekstin automaattista luokitusta varten.

Yrityskäytössä ja tarkkuutta vaativilla sovellusalueilla kyseinen tarkkuus 94 %voidaan kyseenalaistaa. Tavoitteena sovelluksissa on taata mahdollisimman automatisoitu tekstin luokittelumetodi, jossa tarve ihmiselle on mahdollisimman pieni. Kuitenkin 94 % tarkkuus vaatii ihmisen tarkastamisen lopussa, sillä 6 % alkioista on mahdollisuus mennä väärin. Muistettava on, että mikäli sovelluksen tarkoitus on edesauttaa ihmisen tekemää työtä, voidaan 6 % virhe laskea riittävän pieneksi, jotta automaattisen tekstin luokittelun käyttö on perusteltua.

Tarkkuutta vaativat tehtävät, joissa virheiden mahdollisuus tulee olla riittävän pieni, tarkkuus 94 % ei ole riittävä. Tähän voidaan todeta menetelmien vaativan vielä enemmän

esiprosessointia, parametrien hienosäätämistä tai metodien kytkemistä vielä paremman tarkkuuden mahdollistamiseksi.

Työssä ei ole tarkoitus selvittää, mitä keinoja on parantaa suomenkielisen uutisdatan luokituksen tarkkuutta. Tämän takia parametrien, metodien ja datan parantamisen tutkimiseen ei ole perehdytty. Mainittakoon kuitenkin, että tulevaisuuden tutkimuksen kohteena on mahdollista tutkia miten eri parametrien ja luokittelumetodien vaihtaminen sekä datan syvempi prosessoiminen vaikuttaa luokitustulokseen.

5. YHTEENVETO

Tekstin luokitteluun käytettäviä menetelmiä on monia. Lineaariset metodit ovat tuottaneet kilpailukykyisiä tuloksia neuroverkkojen ohella. Tekstin luokittelulla on viisi vaihetta: esiprosessointi, datan muuntaminen numeeriseen muotoon, piirteiden valinta, tekstimuotoisen datan luokitteleminen valitulla metodilla sekä tulosten arviointi.

Diplomityön tarkoituksena on tutkia, minkälaisia menetelmiä on luokitella suomenkielistä uutisdataa. Tutkittaessa menetelmiä saadaan vastaus myös toiseen tutkimuskysymykseen, josta selviää metodien olevan tehokkaita suomenkielisen uutistekstien luokittelua varten.

Työssä käytettävä data koostuu STT uutisista vuosilta 2012–2018, joka tekee yhteensä noin 400 000 kappaletta uutisotsikoita sekä luokkia. Uutisdatan sisältämä sanasto on monipuolinen ja vuosittaiset trendit sekä tapahtumat vaikuttavat selkeästi sanaston alkioihin. Uutisotsikot ovat kieliasultaan vaihtelevia, yleensä lyhyitä, pituudeltaan lauseesta aina muutaman lauseeseen omaavia kokonaisuuksia. Uutisotsikoiden yhtenäistämiseksi ja luokittelumetodien toimintaa varten data esiprosessoidaan tarvittavaan muotoon poistamalla stoppisanat, sekä tekemällä jokaiselle luokittelijalle ominaiset vektorimuotoistamiset, kuten TF-IDF.

Työssä käydään katsaus 6 eri luokittelumetodiin, niiden toimintaan ja aiempiin tutkimuksiin. Kirjallisuuskatsauksessa valitut 6 luokittelumetodia ovat SVM, Naiivi Bayes, BiLSTM, BERT, Kapselit ja KNN. Kaikkien luokittelumetodien on aiemmissa tutkimuksissa todettu kykenevän kilpailukykyisten tuloksien tuottamiseen, mutta testit suoritettiin lopulta 4 soveltuvimman luokittelumetodin välillä.

Kaiken kaikkiaan työssä testit suoritetaan luokitteluun sopivilla metodeilla, joita valitaan 4 kappaletta. Valitut metodit ovat Naiivi Bayes, SVM, Bi-LSTM, sekä BERT. Jokaisen metodin kohdalla esiprosessoitu data opetetaan metodille ja testidatan avulla lasketaan metodin tarkkuus, jotta tulokset ovat vertailtavissa keskenään. Erilaisten valintaperusteiden mukaan metodeille on valittu parametrit ja asetukset. Työssä tarkoituksena ei ole tutkia parametrien vaikutusta luokitustulokseen, vaan vertailla eri luokittelumenetelmiä keskenään ja tämän takia erilaisia parametri yhdistelmiä ei ole kokeiltu.

Parhaiten luokituksessa suoriutui muuntajiin (*transformers*) perustuva BERT-malli. BERT-mallin tarkkuus nousi korkeimmillaan 94,04 %, joka on lähellä viimeaikaisempi tuloksia tekstin luokittelun saralla. BERT-mallia on saatavana monelle eri kielelle, mutta työssä käytetään *multilingual-bert-base-uncased*-mallia, joka on opetettu 100 kielellä,

jotka omaavat laajimman artikkelimäärän Wikipedia dataa. BERT-mallin tarkkuus voidaan selittää sen kuulumisella neuroverkkoihin, jotka ovat osoittautuneet kirjallisuuskatsauksessa tehokkaiksi metodeiksi luokitella dataa. BERT-mallin etuna on sen pohjamalli, joka on opetettu suurella määrällä eri kieliä sisältävää dataa. Suuren opetusdatan määrä vaikuttaa suoraan luokitustarkkuuteen ja erilaisten sovelluskohteiden toimintaan.

Toiseksi parhaiten tehtävästä suoriutui lineaarinen SVM. Lineaarisen SVM tapauksessa on kuitenkin huomattava nopeuden vaikutus luokitustulokseen. Lineaarisen SVM opettaminen vaatii huomattavasti vähemmän opetusaikaa sekä parametrien määrittämistä. Voidaankin todeta lineaarisen SVM olevan vaihtoehtona luokittelijalle, jossa tärkeää on taata nopea käyttöönotto ja vähäiset laitteistovaatimukset. Tehtävissä, jossa lineaarinen SVM onnistuu paremmin kuin BERT on tehtävät, joissa luokitustulokset vaaditaan nopealla aikataululla. SVM tapauksessa esimerkiksi erilaiset demot ja esitykset on helpompi valmistaa lyhyellä aikataululla.

Samaan tarkkuuteen kuin SVM (n. 92 %) työssä ylsi BiLSTM-neuroverkko. Heikoiten työssä käytettävän datan luokittelemisesta suoriutui Naiivi Bayes -luokittelualgoritmi, jonka heikkoutena kirjallisuuskatsauksessa todettiin epätasaisesti jakautunut data.

Voidaan kuitenkin todeta, että varsinkin lineaarisen SVM- ja Naiivi Bayes -luokittelualgoritmin tapauksessa useasti esiintyvien uutisten eli duplikaattien poistaminen pienentää luokituksen tarkkuutta. Tämä voidaan selittää sillä, että duplikaattien poistaminen vähentää käytössä olevan datan määrää ja aiheuttaa siksi pienemmän tarkkuuden. 102

BERT-mallin suorituskykyvaatimukset (20 GB VRAM) on huomioitava, mikäli BERT-malli valitaan käytettäväksi varsinaisessa ohjelmistokokonaisuudessa. BERT-malli vaatii myös ajallisesti huomattavasti enemmän aikaa opetukseen kuin esimerkiksi toiseksi tullut SVM.

Uutisotsikoiden luokitteleminen tehdään usein käsin uutistoimiston puolesta. Voidaan kuitenkin todeta myös automaattisen uutisdatan ja uutisvirran luokittelemisen olevan tärkeä resurssi yrityksille, joissa uutisdataa ei itse luoda, vaan sitä seurataan. Uutisdata voidaan rinnastaa muihin tekstiä sisältäviin datakokonaisuuksiin. Esimerkiksi muut luokittelujärjestelmät, kuten dokumenttien tai asiakasarviointien automaattinen luokittelu voidaan suorittaa samalla datalla, mikäli halutut luokat ovat samantyyliset.

Työ onnistui hyvin aikataulullisista ongelmista huolimatta. Työssä olisin halunnut kokeilla enemmän menetelmiä tekstin luokittelulle ja luokittelumetodien parametreille. Tähän ei käytettävissä oleva aika kuitenkaan riittänyt ja huomio keskittyikin enemmän 4 kirjallisuuskatsauksessa hyvin menestyneen metodin tutkimiseen. Työ vastaa mielestäni hyvin esitettyihin tutkimuskysymyksiin, jonka avulla saa katsauksen metodeihin, joilla suomenkielistä lyhyttä tekstiä voidaan luokitella automaattisesti.

6. LAINATUT LÄHTEET

A. B. Prasetijo, R. R. (2017). Hoax detection system on Indonesian news sites based on text classification using SVM and SGD. *2017 4th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)* (ss. 45-49). Semarang: 2017 4th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE).

Ahmed, H. I. (2018). *Detecting opinion spams and fake news using text classification.* Security and Privacy 1.1.

Asha S Manek, P. D. (2017). Aspect term extraction for sentiment analysis in large. *World wide web 20.2* , 135-154.

B. Tang, S. K. (2016). Toward Optimal Feature Selection in Naive Bayes for Text Categorization. *IEEE Transactions on Knowledge and Data Engineering, vol. 28, no. 9*, 2508-2521.

Barberá, P. B. (2019). Automated Text Classification of News Articles:. *Political Analysis*, 1-24.

Bayu Yudha Pratama, R. S. (2015). Personality Classification Based on Twitter Text. *International Conference on Data and Software Engineering (ICoDSE)* (ss. 170-174). IEEE.

Bhowmick, P. K. (2009). Multi-label text classification approach for sentence level news emotion analysis. *International Conference on Pattern Recognition and Machine Intelligence* (ss. 261-266). Berlin: Springer.

Billsus, D. &. (1999). A hybrid user model for news story classification. Teoksessa *Um99 user modeling* (ss. 99-108). Vienna: Springer.

CHAN Chee-Hong, S. A. (2001). Automated online news classification with personalization. *4th International Conference on Asian Digital Libraries. Research Collection School Of Information Systems.*

Chen, J. T.;Zhong, J.;Xie, Y. C.;& Cai, C. Y. (2014). Text Classification Using SVM with Exponential Kernel. *Applied Mechanics and Materials*, 807-810.

Colas Fabrice, B. P. (2006). Comparison of SVM and Some Older Classification Algorithms in Text Classification Tasks. *IFIP International Conference on Artificial Intelligence in Theory and Practice* (ss. 169-178). Springer Link.

Cortes C., V. V. (1995). Support-Vector Networks. *Machine Learning 20*, 273–297.

Donghwa Kim, D. S. (2019). Multi-co-training for document classification using various document representations: TF–IDF, LDA, and Doc2Vec. *Elsevier Information Sciences Volume 477*, 15-29.

Einea, O.;Elnagar, A.;& Al Debsi, R. (2019). *SANAD: Single-label Arabic News Articles Dataset for automatic text categorization.* Elsevier BV.

Frank E., B. R. (2006). *Naive Bayes for Text Classification with Unbalanced Classes.* Berlin: Springer.

Goldani, M. H. (2020). *Detecting Fake News with Capsule Neural Networks.* arXiv.

González-Carvajal S., G.-M. E. (2020). *Comparing BERT against traditional machine learning text classification.* arXiv.

Goudjil, M. K. (2018). A Novel Active Learning Method Using SVM for Text Classification. *Int. J. Autom. Comput. 15*, 290-298.

Harjule, P. &. (2020). Text Classification on Twitter Data. *10.1109/ICETCE48199.2020.9091774.* , (ss. 160-164).

Hussain, Y. (2019). *PREDICTING CUSTOMER.* Tampere: Tampere University.

Joachims, T. (1998). Text categorization with Support Vector Machines: Learning with many relevant features. *European Conference on Machine Learning* (ss. 137-142). Berlin: Springer-Verlag.

Jotikabukkana, P. S. (2015). Effectiveness of social media text classification by utilizing the online news category. *2nd International Conference on Advanced Informatics: Concepts, Theory and Applications* (ss. 1-5). IEEE.

Karandikar, J.;McLeay, T.;Turner, S.;& Schmitz, T. (2015). Tool wear monitoring using naïve Bayes classifiers. *The International Journal of Advanced Manufacturing Technology*, 1613-1626.

Kim, J.;Jang, S.;Park, E.;& Choi, S. (2020). Text classification using capsules. *Neurocomputing, ISSN: 0925-2312, Vol: 376*, 214-221.

M. I. Rana, S. K. (2014). News classification based on their headlines: A review. *17th IEEE International Multi Topic Conference* (ss. 211-216). Karach: IEEE.

Maiya, A. S. (2020). ktrain: A Low-Code Library for Augmented Machine Learning. *arXiv*.

Mangal, S. B. (2014). Text News Classification System using Naïve Bayes Classifier. *Int. J. Eng. Sci 3* , 209-213.

Marcin Michał Mirończuk, J. P. (2018). A recent overview of the state-of-the-art elements of text classification. *Expert Systems with Applications, Volume 106*, 36-54.

Mähönen Mika. (2013). *Tekstin luokittelu.* Pori: Tampere University of Technology.

Nabamita Deb, V. J. (2020). A Comparative Analysis Of News Categorization Using Machine Learning Approaches. *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH VOLUME 9, ISSUE 01*.

Niusha Shafiabady, L. L. (2016). Using unsupervised clustering approach to train the Support Vector. *Neurocomputing*, 4-10.

Polosukhin, A. V. (2017). *Attention Is All You Need.* arXiv.

Prasetijo, A. B. (2017). Hoax detection system on Indonesian news sites based on text classification using SVM and SGD. *4th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)* (ss. 45-49). IEEE.

Pratama B. Y., S. R. (2015). Personality classification based on Twitter text using Naive Bayes, KNN and SVM. *2015 International Conference on Data and Software Engineering (ICoDSE)* (ss. 170-174). IEEE.

Pyysalo, A. V. (2019). *Multilingual is not enough: BERT for Finnish.* arXiv.

Rennie, J. D. (2001). *Improving multi-class text classification with naive Bayes.* MIT.

Salakoski, J. K. (2018). *Turku Neural Parser Pipeline: An End-to-End System for the CoNLL 2018 Shared Task.* Brussels, Belgium: Association for Computational Linguistics.

Sara Sabour, N. F. (2017). Dynamic Routing Between Capsules. *Advances in neural information processing systems*, 3856-3866.

Srivastava S., K. P. (2018). Identifying aggression and toxicity in comments using capsule network. *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying* (ss. 98-105). TRAC-2018.

Tiago A, A. J. (2011). Spam filtering: how the dimensionality reduction affects. *J Internet Serv Appl*, 183–200.

Toutanova, J. D.-W. (2018). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* arXiv.

U. Pujianto, M. F. (2019). Text Difficulty Classification Based on Lexile Levels Using K-Means Clustering and Multinomial Naive Bayes. *2019 International Seminar on Application for Technology of Information and Communication (iSemantic)*, 163-170.

V Korde, C. M. (2012). TEXT CLASSIFICATION AND CLASSIFIERS: A SURVEY. *International Journal of Artificial Intelligence & Applications (IJAIA), Vol.3, No.2, March 2012*, .

Vidhya.K.A, G. (2010). A Survey of Naïve Bayes Machine Learning approach in. *International Journal of Computer Science and Information Security, Vol. 7, No. 2*.

Vijayarani, S. I. (2015). Preprocessing techniques for text mining-an overview. *International Journal of Computer Science & Communication Networks*, 7-16.

Vladimer B. Kobayashi, S. T. (12.7.2012). *Text Classification for Organizational Researchers: A Tutorial.*

Wei Zhao, J. Y. (2018). *Investigating capsule networks with dynamic routing for text classification.* arXiv.

Wermter, S. &. (2002). Selforganizing classification on the Reuters news corpus. *COLING 2002: The 19th International Conference on Computational Linguistics*.

Yan Yan, Y. W.-C.-W.-C. (2017). LSTM2: Multi-Label Ranking for Document Classification. *Neural Processing Letters 47*, (ss. 117–138).

Zhang Y., L. Q. (2018). *Sentence-state lstm for text representation .* arXiv.

Zhang, W. T. (2008). Text classification based on multi-word with support vector machine. *Knowledge-Based Systems*, 879-886.

Zhao, W. P. (2019). *Towards scalable and reliable capsule networks for challenging NLP applications.* arXiv.

Zhiyong Cui, R. K. (2020). *Stacked Bidirectional and Unidirectional LSTM Recurrent Neural Network for Network-wide Traffic Speed Prediction.* arXiv.