Automated analysis of Norwegian text

Bjarte Johansen

Thesis for the degree of Philosophiae Doctor (PhD)
University of Bergen, Norway
2019

UNIVERSITY OF BERGEN

Automated analysis of Norwegian text

Bjarte Johansen

Thesis for the degree of Philosophiae Doctor (PhD)
at the University of Bergen

Date of defense: 28.06.2019

Year: 2019

Title: Automated analysis of Norwegian text

Name: Bjarte Johansen

Print: Skipnes Kommunikasjon / University of Bergen

Abstract

In this thesis we look at how we can develop automated analysis tools for Norwegian text. We look at 3 different tasks: Part-of-Speech (PoS) tagging, Named-Entity Chunking (NEC), and Named-Entity Recognition (NER).

For our work on PoS tagging, we extend the work done on the OBT+Stat tagger by training a new model to allow it to also do disambiguation of Nynorsk. We work with Googles SyntaxNet and train it for PoS tagging of Bokmål and Nynorsk, showing state of the art results at the time of the research.

We train a Support Vector Machine for NEC of Bokmål. The task of extracting names from text. Next, we develop a NER model using deep learning and provide a NER sequence tagger for Bokmål and Nynorsk. The Nynorsk tagger is the first NER model for Nynorsk that we are aware of. The best performing model is trained on both language forms. It shows better performance on both Bokmål and Nynorsk than the models we trained individually on the language forms.

At last we show how we can use NEC and NER together with Social Network Analysis tools to investigate two case studies around the news story discussing the consequence study of drilling for oil in Lofoten, Vesterålen, and Senja. In the first case study we show that it is possible to find the thematic structures of a news story by analysing the relationship between the entities in the text. In the second case study, using topic modelling, we find the topics, and who the most important persons are for each topic.

Acknowledgments

A thesis is never done alone.

I would like to thank my supervisors Bjørnar Tessem, Dag Elgesem, and Tor Midtbø for accepting me as a Ph.d-student and for the help that I have recieved along the way. I also want to thank my family for believing in me and for soothing my doubts and fears. My mother saw me struggling and told me that it was OK to fail. That lifted a weight of my shoulders and allowed me to continue. My brother has been the person who has helped me the most. He has housed me when I needed a place to stay. He has provided food and wine at his bar when I was hungry and thirsty. He is also probably the one, besides me, that has spent the most time on my thesis. I am very grateful. Another person I want to thank is Karianne. She inspired me to start writing the thesis. I also need to thank Toya. She is incredible and have given me the support I needed to finish this thesis. Lastly, I want to thank Truls. Our discussions have kept me the right kind of insane.

Contents

List of Figures

List of Tables

Acronyms

BiRNN Bidirectional RNN.

CBOW Continous Bag-Of-Words.

CNN Convolutional Neural Networks.

CRF linear-chain Conditional Random Field.

HMM Hidden Markov Model.

LDA Latent Dirichet Allocation.

LoVeSe Loften, Vesterålen, and Senja region.

LSTM Long Short-Term Memory.

NDT Norwegian Dependency Treebank.

NEC Named-Entity Chunking.

NER Named-Entity Recognition.

NLP Natural Language Processing.

OBT Oslo–Bergen tagger.

PoS Part-of-Speech.

RBF radial basis function.

ReLU Rectified linear unit.

RNN Recurrent Neural Network.

SG Skipgram.

SNA Social Network Analysis.

SVM Support Vector Machine.

Chapter 1

Introduction

In this thesis we look at how we can use automated methods for analysing Norwegian text. The general research question we are working on to understand is the question

> "How can we develop and use automatic methods for analyzing unstructured Norwegian text?"

Research in this domain is dominated by English and the research on Norwegian text is, in the best case, fragmented (De Smedt et al., 2012). Even though Norwegian is relatively similar to English, the differences are large enough that it is not guaranteed that the methods that work for English will work as well for Norwegian. The META-NET project reports that, a part from English, no other languages in Europe has a well-developed language resources for data mining and text analysis (De Smedt et al., 2012).

Automated text analysis is a sub-field of Natural Language Processing that investigates how computers can be programmed to understand written language. The field can be divided into three categories: Syntax parsing, information extraction, and language generation. Not every text analysis task fall squarely within one of these categories, but they are useful as a rough categorization of typical tasks within the field.

"Syntax parsing" covers the tasks concerned with understanding the syntactical elements of a text. This includes tasks such finding

word and sentence boundaries in a text, but also part-of-speech tagging, lemmatization, and grammatical analysis. Part-of-speech taggers try to find the category of the words in a sentence. The challenge is that many words can belong to several different categories dependent on the context of the sentence. Lemmatization is the task of finding the base form of a word and remove the inflectional endings. It is often used to reduce the dimensionality of the vector space a model has to consider for categorization or other analysis. Grammatical analysers try to find the grammar of a sentence. The reason for developing parsers is to help further analysis of text by reducing the semantic ambiguities that is inherent in natural language.

"Information extraction" covers the structuring of natural language into a system that a computer can understand. Typical tasks are Named-Entity Recognition, Relationship Extraction, and Sentiment Analysis. Each task concerns itself with finding points of interest in a text: A named-entity recognizer finds the names of persons, organizations, locations, and other entities; a relationship extractor finds what the relation between those entities are; and a sentiment analyzer works to discover the feelings an author projects in their text. For example, a film review can be positive or negative depending upon whether the author liked the film or not.

For the last category, "Language generation", researchers are interested in programming the computer to generate text that is understandable and feels natural. Here, tasks such as summarization are included–where the object is to convert a longer text to a shorter text that still holds the most relevant information. Other tasks include generating news from structured data and translation of a text written in one language to a different language.

There are two main ways of developing models for automated text analysis: Rule-based and statistical models. Historically, the rule-based models have been receiving the most attention, but since around 1996 the statistical models dominate the field (Abney, 1996). Rule-based methods define formal structures that describe how to analyze a

language, while statistical methods analyze large corpora and build a model that fits the evidence for how language is used in that corpora. The rule-based approach is based on the ideas of Chomsky (2002). He claims that there is a set of structural rules that are innate to humans and form a universal grammar that all languages follow.

Chomsky argue that "probabilistic models give no particular insight into some of the basic problems of syntactic structure." Norvig (2011), on the other hand, says that a language is the "contingent outcome of complex processes", and in that sense "can only be analyzed with probabilistic models."

The predominant idea that has taken hold the last couple of years is to model Natural Language Processing tasks as sequences to be labeled. The most popular sequence labeling techniques are variants of the LSTM BiRNN, like we describe in section 2.4.3. Though neural network architectures are heavily used within the field, they do require large sets of training data and ample computing resources to produce well-performing models. Neural networks also allow us to do little or no feature engineering as deep neural networks have the capacity to discover and encode the features as part of the training process. The negative aspect of this ability to learn features is that it becomes difficult to reason about what those features are. It also becomes difficult to know why the neural network decides the label for a particular input.

Though RNNs have been known since the 1980's (Rumelhart et al., 1986), it was not before around the 2010's that they saw their breakthrough as a technique used for natural language processing (Goodfellow et al., 2016, Chap. 10). There simply was not enough resources before that time to efficiently train and validate RNN and other deep neural network models.

Before neural networks became popular, researchers would define and build a feature vector that a model would use to learn a task—also called feature engineering. We do that in section 5.2 when we train a Support Vector Machine to do Named-Entity Chunking. Feature engineering is still popular in situations with low resources and where there are

not enough available data to use neural networks and similar algoritms. Support Vector Machines, Hidden Markov Models, and Conditional Random Fields are examples of classification algorithms that used to be popular with automated text analysis researches. Some algorithms, like the Conditional Random Field, are still used in conjunction with neural networks—as we use in 5.3 where we train a model for Named-Entity Recognition.

Research on Norwegian text has mostly been based on rule-based and hybrid approaches. Projects like the Oslo-Bergen Tagger (Bick et al., 2015) and the added statistical disambiguator (Johannessen et al., 2011) employ this approach to language analysis. In Norway, it has mostly been the computational linguistics community that has worked on developing tools for automated text analysis. Their interests have been in the structure and grammar of language and how language is used, instead of as tools for data mining. They have therefore opted to make tools that expose the uncertainties in their models and help them investigate grammatical structures.

Recently, the trend has been to take advantage of international research successes by building corpora that follow international standards. The work on the Universal Dependency Treebank for Norwegian (Øvrelid and Hohle, 2016) and the Norwegian Review Corpus (Velldal et al., 2018) are examples of this trend. Most state-of-the-art methods for tasks like Part-of-Speech and Sentiment analysis on English text require large corpora to train well-performing models, and it is easier to adapt those methods to Norwegian when the input follows the same structure.

We use many different technologies in the research for this thesis based on statistical models. Technologies like Support Vector Machines, linear-chain Conditional Random Fields, and Deep Neural Networks. We also use Social Network Analysis to research two case studies in analysis of Norwegian text. All of these methods and technologies are explained in depth in chapter 2.

Norwegian use the same script as English and is somewhat similar.

However there are also many differences. We look at some of the interesting characteristics of Norwegian text (in the context of automated analysis) in chapter 3.

Norwegian has a few challenges that has to be overcome to solve the problems that we are interested in. Norwegian has its own capitalization rules that affect how names are written. It uses compound words, and compound words cannot be split into its constituents as that can drastically change the meaning of a sentence. Polysemy—or that the same word can mean different things when the context changes—and ambiguities in the language makes it difficult in some instances to know the semantic meaning of a sentence without further context. Norwegian also has 2 written forms, Nynorsk and Bokmål. Each of the written forms also varies depending on where the authors is from and the region they live in.

In this thesis we focus on three main Natural Language Processing tasks: Part-of-speech tagging, Named-Entity Chunking, and Named-Entity Recognition.

In chapter 4 we explain the Part-of-Speech task and perform 2 studies on Part-of-speech tagging:

Training OBT+Stat in Nynorsk — There are few resources for automated analysis of Bokmål, and even fewer resources for Nynorsk. We wanted to see if we could update the statistical disambiguator for the Oslo-Bergen tagger to also be able to do part-of-speech tagging for Nynorsk as well.

Training SyntaxNet to understand Bokmål and Nynorsk — For this study, the goal was to take an off-the-shelf tool that had been developed for English and see how it performed on the Norwegian language forms. Since SyntaxNet was performing at a state-of-the-art level on English, we wanted to see if it can outperform the OBT+Stat tagger.

A well-performing Part-of-speech tagger is important for other Natural Language Processing tasks as it can help to remove ambiguities

caused by polysemy. We use Part-of-Speech as a feature for our models in chapter 5—where we research named entitites in text. We want to find the locations, organizations, persons, and other names that appear in a corpus. We perform 2 different studies on named entities:

Named-Entity Chunking — Other studies have look at what the type a name has, but their attempts do not delineate the names from the rest of the text. We investigated how we could develop a model that marks which sequences of tokens are names, also called chunking, to perhaps make it possible to use these previous attempts or investigate similar approaches in the future.

Named-Entity Recognition — In our second study we used deep learning to create a model for both delineating the names from the text and categorizing them in one step. We, again, based our study on state-of-the-art research from studies on English text. We showed that we could get better results than what has been previously achieved on Norwegian Bokmål—even though previous research only work on categorizing names. Our research represents the first attempt, that we are aware of, for a Named-Entity Recognition model for Nynorsk. The best performing model uses a joint model for both Nynorsk and Bokmål.

Named-Entity Chunking and Recognition can be used as a tool to investigate the relationship between entities in large corpora. In chapter 6 and 7 we investigate two different case studies where we analyze such networks in a news story:

The thematic structure of news stories — In this case study we present the news story on the consequence study of oil drilling in Lofoten, Vesterålen, and Senja. The consequence study has been a hot topic for many years in Norway, but became a large part of the political campaigns before the election in 2013. Given the assumption that journalists will usually put thematically relevant entities together in the same article, we wanted to see if we could find that thematic

structure through Social Network Analyis. We extracted all names in the corpus through Named-Entity Chunking and created a network based on which articles they appear together in. We found 6 different groups that we think represent the different thematic views on the study.

Who are talking to whom about what? — In the second study we used Named-Entity Recognition to find only the persons in the text in the same news story about the consequence study. We used topic modelling to automatically find the different topics of the news story. We then investigated which persons are the most important in each of the topics and which persons are the information carriers between the topics.

It could seem like we are working on very disparate topics: Part-of-Speech Tagging, Named-Entity Chunking and Recognition, and Social Network Analysis of news stories. However, to be able to do Social Network Analysis of news stories we need a chunker and recognizer to find the names and name categories. To develop a well-performing chunker and recognizer we need a Part-of-Speech tagger. We also believe that by investigating the full stack of topics we get a unique insight into the inner workings, strengths, and weaknesses of these tools. Through the two case studies we also show the usefulness of the tools that we have developed for this thesis.

Further, we discuss what other researchers have done that is similar to our research in chapter 8. In chapter 9 we discuss what we have learned from the different studies and how they relate to each other. Lastly, we come to a conclusion and discuss future work in chapter 10.

1.1 Motivation

Grimmer and Stewart (2013) says that as long as the limitations of automated text analysis methods are recognized and the validity of

the methods are demonstrated, they will revolutionize the study of political science. We believe that this also holds in other branches of the social sciences that rely on analysis of textual media. Hannigan (2015) argues that interdisciplinary cooperation between social science and natural language processing has the potential to propel the field of organizational research and content analysis forward.

However, in many cases, these methods need to be tailored to the language of the corpus that they are used on. It can in some cases be possible to use English language resources to study texts in other languages, as discussed by Lucas et al. (2015), but it is difficult to evaluate the validity of such approaches.

The main academic reason for this thesis is therefore the lack of resources for automatic language analysis of Norwegian text. A study from 2012 by META-NET showed that in Europe, no other language than English has a good coverage of language resources for information extraction and text analysis, and that the research on Norwegian text is "at best fragmented" (De Smedt et al., 2012).

Some research has shown that the linguistic distance between English and Norwegian is smaller than for other languages (Chiswick and Miller, 2005). (One researcher has even claimed that English is actually a Scandinavian language (Nickelsen, 2012).) It is however difficult to tell if the methods that work for English will work just as well for Norwegian. This is especially true for those methods that are based on grammar and the presence of specific words.

Research on other languages than English, like the research we are conducting for this thesis, can also produce insights back into the already established research by identifying blind spots and produce new questions.

A unique aspect of Norwegian is that it has 2 different official written forms that are quite similar, but have many differences. Both of the written forms also have large internal variations in how they are written and how words are formed (De Smedt et al., 2012). Though this is usually thought of as a problem, we show evidence that training on

closely related and similar languages can improve machine language learning. Having two written forms, Nynorsk and Bokmål, should be viewed as a challenge and an opportunity and not as a problem.

Norwegian as a natural language research platform, as established by META-NET, has not seen any large infusion of resources, but if we want to continue to make Norwegian relevant for technologies like voice recognition, robot assistants, and other newly developed and developing technologies we need to put time into researching tools for Norwegian text and language.

Chapter 2

Computational Methods

In this chapter we describe the methods that were used in the production of the experiments in this thesis. We use many different technologies to develop the experiments in our research, from classical Support Vector Machine and Hidden Markov Models, to Deep Neural Networks with LSTM units and Linear Chain Conditional Random Fields. For the case studies we also employ various techniques from Social Network Analysis together with Topic Modeling to investigate the entities that appear in news texts.

2.1 Support Vector Machines

A Support Vector Machine (SVM) is a type of supervised learning algorithm "where the goal is to find a decision boundary between two classes that is maximally far from any point in the training data" (Manning et al., 2008, p. 293).

Figure 2.1 shows how an example model could look after training a SVM with samples from two different classes. The hyperplane is the solid line in the middle, while the stippled lines is the margin to the hyperplane. The solid-coloured samples on the margins are the support vectors of the model.

For our research, we are interested in distinguishing between multiple

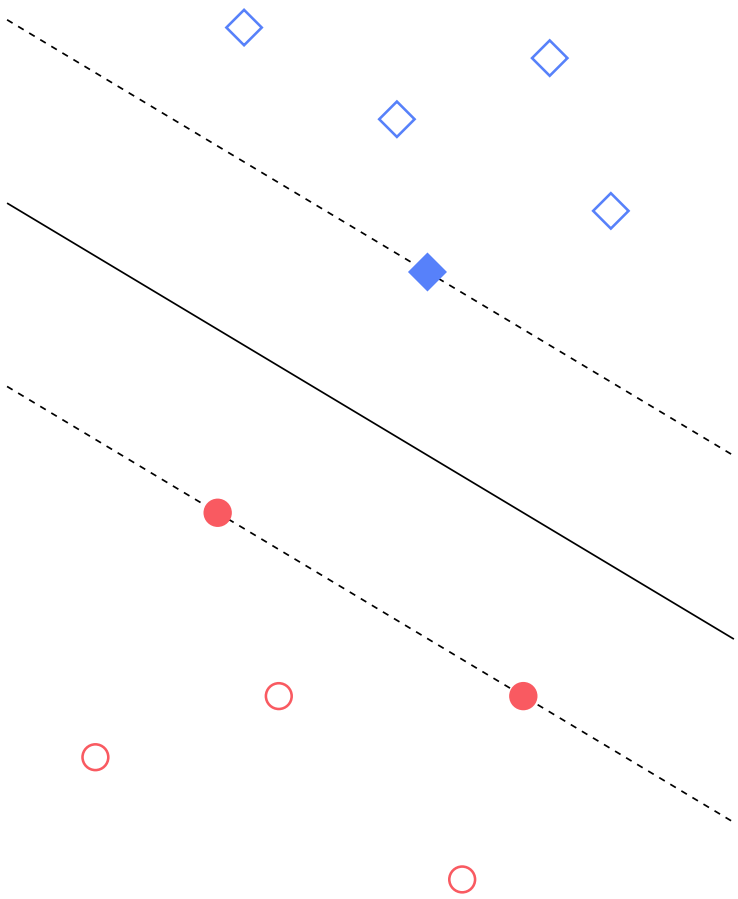

Figure 2.1: Example result of training a SVM. The circles and diamonds are two different classes of objects. The black line is the hyperplane found by the SVM, the stippled line is the margin to the hyperplane, and the solid-coloured points are the support vectors.

classes, but traditionally a SVM is only able to differentiate between two. To get around this constraint we use an extension to SVMs which supports multi-class data: the "one-versus-one" approach. The classifier builds a SVM for each pair of classes and chooses the class that is selected by a majority of the classifiers.

In the case of labeling errors there might not be possible to find a hyperplane that cleanly separates the classes of the training data. To get around this constraint it is possible to use the soft-margin method to allow for some classification errors. The soft-margin method defines C as the soft-margin parameter to the error function and controls how much a classification error is penalized (Vert et al., 2004). The size of C can therefore result in over- or under-fitting by making the SVM choose a small or large margin hyperplane.

The kernel type that we use in our research, described in section 5.2, is the radial basis function (RBF) which allows the SVM to also classify nonlinear data by lifting the data into higher dimensions where they might be linearly separable after all. It defines γ as a hyperparameter and the free variable of the kernel and decides how the points in the problem space are lifted into higher dimensions to make it easier to separate the different classes from each other. The RBF kernel should be able to find any linear separation that both a linear and polynomial kernel is able to find, though it is more expensive to compute.

In section 5.2 we train a SVM model in Named-Entity Chunking—or to delineate between named entities and the surrounding text.

2.2 Hidden Markov Models

A Hidden Markov Model (HMM) is "a tool for representing probability distributions over sequences of observations" (Ghahramani, 2001). The HMM gets its name from two defining properties. The model assumes that an observation at time t was generated by a processes whose state is *hidden* from the observer, it then assumes that this state satisfies the

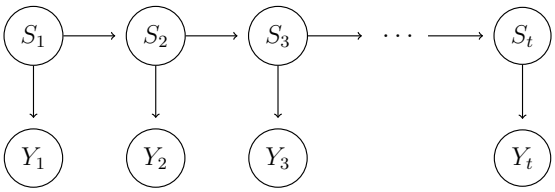

Figure 2.2: Hidden Markov model.

Markov property.

The Markov property says that given the state at a previous timestep, the current state is independent of all states prior to the previous state. This means that the state at any given time represents all of the history of a process that is needed to predict the future state of the process. HMMs are described by the equation (2.1):

$$P(S_{1:T}, Y_{1:T}) = P(S_1)P(Y_1|S_1)\prod_{t=2}^{T} P(S_t|S_{t-1})P(Y_t|S_t) \tag{2.1}$$

The equation says that the probability of a state sequence $S_{1:T}$ producing the sequence of observations $Y_{1:T}$ is equal to the probability of the first state, S_1, times the probability of the observation given the first state, $P(Y_1|S_1)$, times the joint product of the probability that each of the next states follows the previous state, $P(S_t|S_{t-1})$, together with the probability that the state produces the observation at time t, $P(Y_t|S_t)$.

We use a HMM in section 4.2.1 to train a tagger called OBT-Stat to tag text written in Norwegian Nynorsk.

2.3 Linear-chain Conditional Random Fields

A linear-chain Conditional Random Field (CRF) is a method used to classify sequences of interdependent variables (Lafferty et al., 2001). An example would be to classify the words in a sentence as a person, organization, or location. While HMMs, as described in section 2.2, assumes that the next state is only dependent on the previous state, CRF allows us to also include features from any point in the sequences. It does that by introducing a set of real-valued feature functions $\mathcal{F} = \{f_k(y, y', \vec{x_t})\}_{k=1}^{K}$ and a parameter vector $\theta = \{\theta_k\} \in \Re^K$. A CRF is then a distribution $P(\vec{y}|\vec{x})$ that takes the form (Sutton et al., 2012):

$$P(\vec{y}|\vec{x}) = \frac{1}{Z(\vec{x})} \prod_{t=1}^{T} \exp\left\{ \sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, \vec{x_t}) \right\} \tag{2.2}$$

where $Z(\vec{x})$ is an input dependent normalization function

$$Z(\vec{x}) = \sum_{y} \prod_{t=1}^{T} \exp\left\{ \sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, \vec{x_t}) \right\} \tag{2.3}$$

The vector $\vec{y}$ are the labels that the CRF is predicting, and has the form $\vec{y} = \{y_0, y_1, \ldots, y_T\}$. $\vec{x}$ are the feature vectors that are used to predict a label for some input. $\vec{x}$ has the form $\vec{x} = \{\vec{x_0}, \vec{x_1}, \ldots, \vec{x_T}\}$. The parameter vector θ is usually learned from the data through an optimization algorithm like stochastic gradient descent or Adam.

We use a CRF as the final layer in a model for Named-Entity Recognition in conjunction with a LSTM-BiRNN and other techniques in section 5.3. The LSTM-BiRNN (described in section 2.4) condenses the information and outputs a feature vector θ that the CRF uses to calculate the most probable sequence of labels for the words in the sentence.

2.4 Deep Neural Networks

In this section we describe the type of Deep Neural Networks and the accompanying methods that we use in the research for this thesis. Those include Dense layers, Convolutional Neural Networks (CNN), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM) units, Word Embeddings, cross entropy loss, and the Adam optimizing algorithm.

2.4.1 Dense layer

A dense layer in a network is a layer where every input to the layer is connected to every output (Mitchell, 1997). It still has a weight for every connection, an activation function, and a bias for every output

in the network. An example of a dense neural network can be seen in figure 2.3.

A dense layer is useful as a way to reduce the dimensionality of the output from other layers such as a RNN. The reason is that the output of a RNN would have the same size as its hidden size. For example, if we the hidden size of a RNN is set to 512 neurons, the output vector from the RNN would be 512 values as well. To reduce the dimentionality of the RNN, every output value of the RNN is connected to the neurons of a dense layer. The dense layer is set to be the same size as our desired output—normally the same size as the number of labels. Normalizing the output of the dense layer will then give a likelihood for each label in the vocabulary.

Each node in the neural network calculates the affine transformation where the inputs $\vec{x}$ are weighted by the kernel $\vec{w}$ and then summed together with a bias b. Adding a bias to the sum allows the network to change the shape of the activation function such that it can fit the input to the prediction better. The bias is either set to a specific number like 1, or trained as one of the parameters of the network. The sum is then put through an activation function:

$$f(\vec{x} \cdot \vec{w} + b)$$

The simplest function is the binary function, which models a biological neuron that is either activated or not activated by the input to the function:

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$$

A popular function is the logistic function, which maps the input onto an S-curve and limits the input to a value between 0 and 1:

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$$

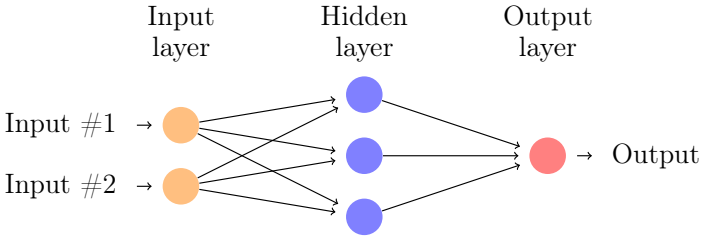

Figure 2.3: An example of a dense neural network.

The hyperbolic tangent, or tanh, is also often used. Especially with the popularity of the LSTM cell for RNNs described in section 2.4.4. It has the form:

$$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Another popular function is the Rectified linear unit (ReLU):

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$

It is used in many types of tasks from image classification to machine translation (Ramachandran et al., 2018).

We use a linearly-activated dense layer where the activation function returns the identical result to the input: $f(x) = x$. We use it to reduce the dimensionality of the output from a Bidirectional RNN (BiRNN) in section 5.3 to build a model for Named-Entity Recognition.

Another activation function we use is the softmax function. It calculates the normalized exponential and gives us a way to interpret the output from a previous layer as a likelihood for each label in our vocabulary (Goodfellow et al., 2016).

$$\phi(x)_j = \frac{e^{x_j}}{\sum_{k=1}^{K} e^{x_k}} \text{ for } j = 1, \ldots, K$$

We use it to output the likelihood for the entity labels for our Named-Entity Recognition model.

2.4.2 Convolutional Neural Networks

Convolutional Neural Networks are "neural networks that use convolution in place of general matrix multiplication" (Goodfellow et al., 2016) and are often used in image classification. Using a dense network for this task would require too many neurons to be possible to train in a reasonable amount of time. Instead of operating on every point of the

image, each neuron operates on a n-dimensional view of the input.

This technique can also be used in natural language processing. We use a 1-dimensional CNN with a ReLU function to learn character embeddings to use as part of the features when we train a model for Named-Entity Recognition (NER) in section 5.3.

2.4.3 Recurrent Neural Networks

Recurrent Neural Networks "are a family of neural networks for processing sequential data" (Goodfellow et al., 2016, Chap. 10). They work by including the result of previous input to the neural network as part of the parameters that the network accepts. This means that the neural network can take into account how previous input in a sequence affect input that appear later in the same sequence. How the RNN tracks what to keep from previous input is determined by the type of cell that the RNN utilizes. We use an LSTM as the cell in our networks. LSTMs are useful as they create paths through time and allow for information to accumulate over a long period. We describe them further in section 2.4.4.

A RNN iterates for each timestep over the following equations:

$$h_t = \mathcal{H}\left(W_{xh}x_t + W_{hh}h_{t-1} + b_h\right) \tag{2.4}$$

$$y_t = W_{hy}h_t + b_y \tag{2.5}$$

where W denotes the weight matrices and for example W_{xh} is the hidden weight matrix for the input. b is the bias vector. $\mathcal{H}$ is the hidden layer function. $\mathcal{H}$ is usually the element-wise application of a sigmoid function. h_t is the hidden state at time t, and y_t is the output at time t.

We can also put multiple cell into each their own layer of the RNN. We will then have to compute the following equation instead:

$$h_t^n = \mathcal{H}\left(W_{h^{n-1}h^n}h_t^{n-1} + W_{h^nh^n}h_{t-1}^n + b_h^n\right) \tag{2.6}$$

It says that hidden state for the current layer n at time t is the result of the affine transformation of the hidden state at the previous layer at the same time and the previous state at the same layer at the previous time. In this instance we define the first hidden state as $h^0 = x$.

Normally a RNN will run from the first element of a sequence to the last, and that is at its essence true, but since the operator (usually) controls the sequence it is possible to present the words in any order that is desired. For example in a BiRNN we train two RNNs where one RNN traverses the sequence from the first to the last item, but for the other RNN we present the sequence in reverse order. A popular technique is to concatenate the result of two such RNNs traveling in opposite directions forming a BiRNN. The idea is to capture information that can be used for classification from both the past and the future of the sequence for each timestep.

We use this feature of the BiRNN in section 5.3 to train a model for NER. We treat the words in a sentence as a sequence that we input to the BiRNN.

2.4.4 Long Short-Term Memory units

Long Short-Term Memory units introduces "self-loops to produce paths where the gradient can flow for long durations" and thereby capturing long-term dependencies (Goodfellow et al., 2016, Chap. 10).

A LSTM RNN basically works in the same way as described in equation (2.4), (2.5), and (2.6), but $\mathcal{H}$ is implemented by the following functions instead:

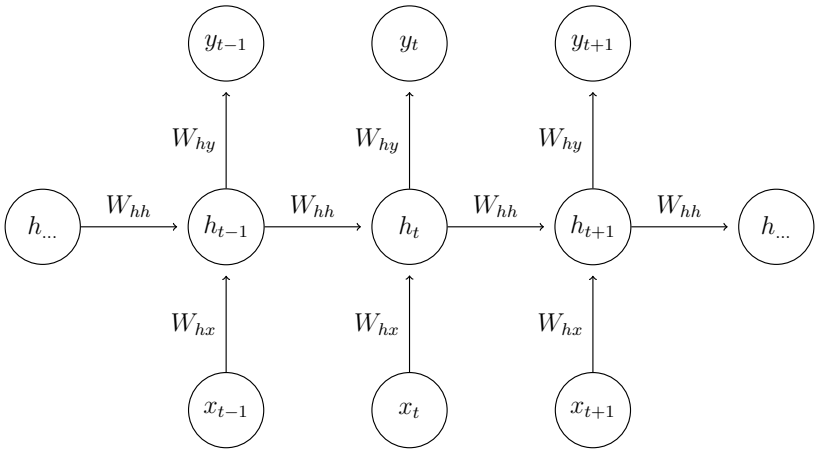

Figure 2.4: Recurrent Neural Network

$$i_t = \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i\right) \tag{2.7}$$

$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right) \tag{2.8}$$

$$o_t = \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1} + b_o\right) \tag{2.9}$$

$$c_t = f_t c_{t-1} + i_t \tanh\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right) \tag{2.10}$$

$$h_t = o_t \tanh\left(c_t\right) \tag{2.11}$$

where σ is the logistic sigmoid function. i, f, and o are the *input gate*, *forget gate*, and *output gate* , and that in equation (2.7) and (2.8) their value at time t is the addition of the affine transformation of the input vector, the previous hidden state, and the previous cell activation, with the weight for that gate. c is the stored "long term" memory, which is described in equation (2.10) as the result of the inner activation function tanh on the affine transformation of the input and previous hidden state together with the result of the input gate, and added to the result of putting the forget gate together with the previous cell activation. The hidden state (h) is then described in equation (2.11) as the result of the output gate together with the application of the inner activation function of the cell activation.

We use a LSTM cell in our NER model described in section 5.3 to capture the long term dependencies between words in a sentence.

2.4.5 Embeddings

Word embeddings, or distributional semantic models, are "mappings $V \to \mathbb{R}^D : w \mapsto \vec{w}$ that maps a word w from a vocabulary V to a real-valued vector $\vec{w}$ in an embedding space of dimensionality D" (Schnabel et al., 2015); and that means that instead of representing a word as a high-dimensional vector with the same number of dimensions as there are words in the relevant vocabulary, we map those vector onto a smaller, real-valued space. We are in other words trying to mitigate

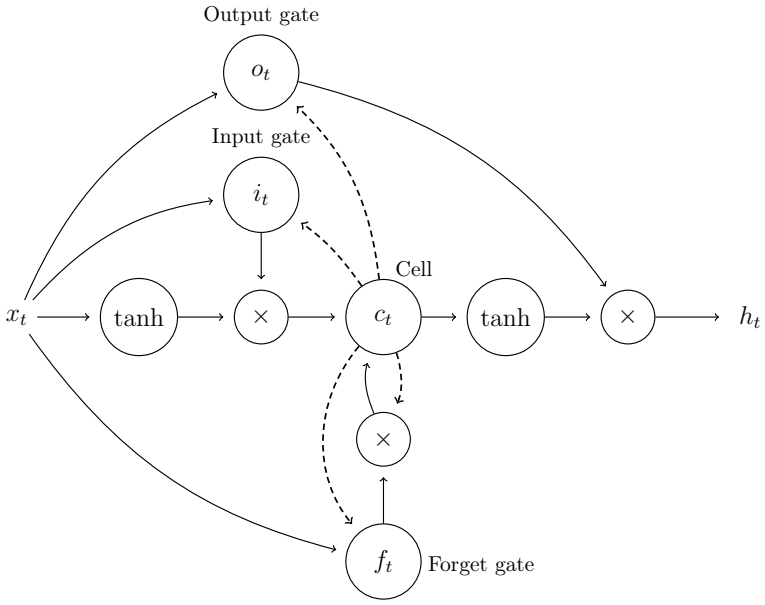

Figure 2.5: LSTM cell: A stippled line means we access the data from $t-1$. The illustration does not show the hidden weights of the model.

the curse of dimensionality: As the number of dimensions grows, the training data occupies less and less of the space and therefore becomes more sparse and we need increasingly more observations to train a well-performing model for the problem (Trunk, 1979). This is particularly problematic for language models where we are trying to model the joint distribution between many discrete random variables: "For example, if one want to model the joint distribution of 10 consecutive words in a natural language with a vocabulary of size 100000, there are potentially $100000^{10} - 1 = 10^{50} - 1$ free parameters" that need to be trained (Bengio et al., 2003).

Two models for word embeddings proposed by Mikolov et al. (2013), are the Continous Bag-Of-Words (CBOW) and Skipgram (SG) models. The CBOW architecture tries to predict the current word using the surrounding context by minimizing the loss function:

$$E = -log(P(\vec{w_t}|\vec{W_t})) \tag{2.12}$$

where w_t is the target word and $W_t = w_{t-n}, \ldots, w_t, \ldots, w_{t+n}$ is the word in context with the n words in front and behind it.

The SG model is similar, but the goal is instead to predict the surrounding words given the current word or minimize the loss function:

$$E = -log(P(\vec{W_t}|\vec{w_t})) \tag{2.13}$$

An embeddings model like SG or CBOW can be learned by training it like a simple projection layer in a neural network. It can be also be done unsupervised: For example, for an input sequence of words, each word is converted into a one-hot vector with the dimentionality of the vocabulary. Then, the layer is trained using an optimizing algorithm and one of the loss functions described above. Figure 2.6 and 2.7 shows a graphical representation of the input, projection layer, and output of CBOW and SG model.

Embeddings models are not limited to sequences of words; they can also add sub-word information as part of the calculation as shown by

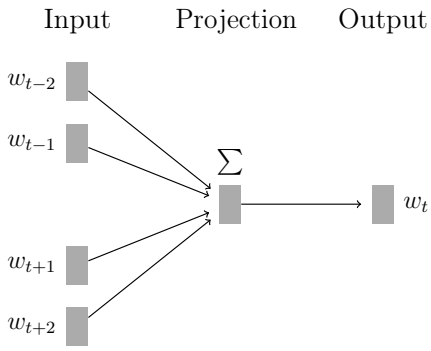

Figure 2.6: CBOW model

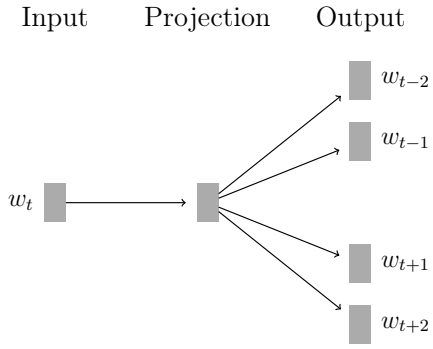

Figure 2.7: Skipgram model

Bojanowski et al. (2017). They learn representations of n-grams of characters within a word and then represent a word as the sum of the n-gram vectors. They show that this type of representation can help increase accuracy for models trained on morphologically rich languages.

We train a sub-word embeddings model on a combined Nynorsk and Bokmål corpus that we use as the first layer in the NER that we describe in section 5.3.

2.4.6 Cross entropy loss

To train a neural network the optimizing algorithm and the back-propagation step has to be provided with a loss function. A popular loss function is the cross entropy loss of the likelihood for each of the predicted labels and the ground truth (Mitchell, 1997):

$$H(p, q) = -\sum_i p_i \log q_i \tag{2.14}$$

where p_i is the likelihood of the predicted output of the network of example i and q_i is the ground truth of what the next label should be. The result of the cross entropy of two probability distributions is how many bits are needed to represent the difference between the two distributions. The smaller the difference, the more similar they are.

We use the cross entropy loss as the loss function for our optimizing algorithm when we train a model for NER in section 5.3.

2.4.7 The Adam optimizing algorithm

Adam is an algorithm for "first-order gradient-based optimization of stochastic objective functions" (Kingma and Ba, 2014). It gets its name from the fact that it uses "adaptive moment estimation" to train the weights in the model based on the local moments, instead using the global moments as the estimated error.

The way the algorithm works is by calculating adaptive learning rates for different parameters by estimating the mean (the first moment)

and the uncentred variance (the second moment).

In further detail, it first calculates the gradient for the stochastic objective of our loss function. Then it updates the first and second moment estimates based on the current timestep. It then uses the individual moment estimates of each gradient to calculate the updated parameters for the loss function. To update the network, it uses back-propagation of the errors through the network to update all the weights of the network.

To avoid the problem of exploding gradients in RNNs as described by Bengio et al. (1994), it is adviced to clip the gradients to the global norm, or to a max value, as suggested by Pascanu et al. (2013). The reason for this problem is that RNNs allow the network to keep information about the past for an unspecified amount of time. This results in "an explosion of the long term components, which can grow exponentially more than the short term ones" (Pascanu et al., 2013).

We train our NER model that we describe in section 5.3 using the Adam optimizing algorithm.

2.5 Precision, recall, and F score

In section 2.4.6 we described the cross entropy function which is used to calculate the difference between the training set of a model and the output it gives, but to measure and understand the efficacy of a model it is better to use measures such as precision, recall and the F_β score.

Precision is the percentage of retrieved documents that are relevant

$$\text{Precision} = \frac{|\text{ relevant items retrieved }|}{|\text{ retrieved items }|} = P(\text{relevant}|\text{retrieved}) \quad (2.15)$$

Recall is the percentage of relevant documents that are retrieved

$$\text{Recall} = \frac{|\text{ relevant items retrieved }|}{|\text{ relevant items }|} = P(\text{retrieved}|\text{relevant}) \quad (2.16)$$

The F_β score is the harmonic mean of the precision and recall and allows us to make a tradeoff between precision and recall

$$F_\beta = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2+1)PR}{\beta^2 P + R} \text{ where } \beta^2 = \frac{1-\alpha}{\alpha} \tag{2.17}$$

In our research we use the balanced F_β score where $\beta = 1$ or $\alpha = 1$ as a measure of the accuracy of our models. When $\beta = 1$ the formula in equation 2.17 simplifies to

$$F_{\beta=1} = \frac{2PR}{P+R} \tag{2.18}$$

A higher β will emphasize recall, while a lower β will put more weight on precision. The reason for using the harmonic mean between precision and recall instead of the arithmetic mean is because it is always possible to get a perfect recall score by having the model return all results. This means that the arithmetic mean of precision and recall will be at least 50% as we have found 100% of the relevant items. The harmonic mean, on the other hand, will always be closer to the smaller of the two values than to their arithmetic mean (Manning et al., 2008).

We use recall, precision, and the F_β score to measure the performance of all of the models that we develop in this thesis and as a way to compare our results with the results of other researchers.

2.6 Latent Dirichlet Allocation

Latent Dirichet Allocation (LDA) is a method that is used to find the topics in a corpus. LDA is "a generative probabilistic model for collections of discrete data" (Blei et al., 2003). In LDA the documents are represented as random mixtures over latent topics where each topic is a distribution of words. This means that each document has the possibility of containing multiple topics, or rather, each document has a distribution of topics within it.

According to Blei et al. (2003), LDA assumes that the documents w in a corpus D was generated given the following generative process for each document:

1. Choose $N \sim Poisson(\xi)$
2. Choose $\theta \sim Dir(\alpha)$
3. For each of the N words w_n:
   (a) Choose a topic $z_n \sim Multinomial(\theta)$
   (b) Choose a word w_n from $p(w_n|z_n, \beta)$, a multinomial probability conditioned on the topic z_n.

To actually calculate the probabilities for each word in each of the topics we need to know the number of topics in the corpus. In many occasions the number of topics is chosen based on domain knowledge or expert opinion. However, there are also some metrics available that can be used to inform an opinion on the number of topics.

We use LDA to find the topics in the case study that we describe in chapter 7.

2.6.1 Choosing the number of topics

The following 4 metrics are used to calculate how well the chosen number of topics fits the current corpus:

Arun2010 The symmetric Kullback-Leibler divergence of the Singular value distribution of the topic-term matrix and the distribution of the length of each document over the document-topic matrix. (Arun et al., 2010).

CaoJuan2009 The average cosine distance of the topics. (Cao et al., 2009).

Griffiths2004 The approximate likelihood of the words in the corpus given the number of topics (Griffiths and Steyvers, 2004).

Deveaud2014 The information divergence between all pairs of LDA topics (Deveaud et al., 2014).

Each metric measures a different score for the topics in the corpus. To use the metrics to decide on the number of topics in a corpus, one needs to run LDA analysis for the full range of number of topics that one is interested in. The metrics are calculated for each analysis and compared to see which model performs the best. Depending on which what is being researched and at the discretion of the researcher, one can also put more or less emphasis on one or more of the metrics.

2.7 Social Network Analysis

Social Network Analysis (SNA) methods are tools to investigate and analyze relational data such as "the relationship between social entities, and on the patterns and implications of these relationships" (Wasserman and Faust, 1994).

Each relationship between the entities become edges between nodes in a graph (or network) and can be used to calculate different metrics like the importance of a node and the communities that appear in it. These metrics makes it possible to quantify and measure the interactions between social agents and makes it possible to "prove theorems and deduce testable statements" (Wasserman and Faust, 1994).

An example of a social relationship—that we will investigate in the two case studies later in chapter 6 and 7—could be that some entities appear together in a newspaper article more than other entities based on the theme or topic of that article. In this instance, the nodes are the persons, organizations, and locations that appear in the story, and the edges describe that they have appeared in the same article together.

It can also be beneficial to describe how many times the entities appear together. This is the edge weight. The edge weight is often used to calculate metrics like the node strength, which is the sum of the edge

weights. This can in turn be used to calculate the importance of a node or used in community detection.

We use three ways of describing importance of nodes in a graph in our work:

Eigenvector centrality measures the importance or prestige of a node in a graph. It is based on the idea that a nodes importance is influenced by the importance of the nodes that it is connected to (Wasserman and Faust, 1994).

PageRank gives the likelihood that you will end up at a particular node given that we randomly follow the edges of a graph from any other node in the graph (Page et al., 1999).

Betweenness centrality measures the importance of a node by looking at how many paths between other nodes the given node controls. For example, a secretary for several important executives that controls who can talk to the executives would themselves become important as everyone elsewhere in the company would need to go through the secretary to get to the executives. In other words, a node gets a high betweeness score if they control many paths between other nodes in the graph (Wasserman and Faust, 1994).

Another concept in SNA and graphs is the connectivity or cohesiveness of a graph. "A graph is cohesive if, for example, there are relatively frequent [edges], many nodes with relatively large degrees, or relatively short or numerous paths between pairs of nodes" (Wasserman and Faust, 1994).

One of the methods from SNA that we are interested in is community detection—or finding highly connected subgraphs that has few edges between them.

A measure for evaluating how well a given collection of subgraphs, or a community structure, divides the graph into groups is *modularity*. Modularity was first described by Newman and Girvan (2004) and tries to maximize how many edges are contained within the communities

and split the graph into many communities where each community has a small total degree, described by the following equation:

$$q(\mathcal{C}) = \sum_{\Upsilon \in \mathcal{C}} \left[\frac{|E(\mathcal{C})|}{m} - \left(\frac{\sum_{v \in \Upsilon} deg(v)}{2m} \right)^2 \right] \tag{2.19}$$

where $\mathcal{C}$ is a community structure describing a graph and each $\Upsilon \in \mathcal{C}$ is a community, or cluster, in the graph. m is the number of edges in the graph. $E(\mathcal{C})$ is the set of intracluster edges, or edges going between the clusters. The first term in the in equation 2.19, is the fraction of edges that connect nodes in the same community. The second term describes the expected value of how many edges a node is connected to.

A problem with modularity is that some graphs do have strong communities, but there are a few highly connected nodes that drives the modularity score down. A solution to this is targeted node removal, also known as violator removal, to improve the modularity of the community structures that are found in the graph.

In our research, we have used the method proposed by Wen et al. (2011):

1. Calculate which node to remove to get the highest modularity gain.
2. Remove the node and repeat.
3. Use changepoint detection to identify when we had the largest increase in modularity to say how many nodes to remove.

Even though we are removing nodes that could hold a position of importance within a community in the graph, we believe this method helps us find the best division between the communities when we are more interested in finding the communities than we are in preserving every node in the community.

We use SNA to find the groups in a news story in chapter 6. We also use SNA to find the most important persons together with which persons appear together in a news story in chapter 7.

Chapter 3

Characteristics of Norwegian text

In this chapter we describe some of the characteristics of the Norwegian language that are important to take into consideration when working with automated analysis of Norwegian text. Norwegian is not ideally suited for automated analysis as there are stylistic choices and particularities of the language that force a semantic understanding that is not captured in the immediate structure of the text.

The Norwegian language has a large number of polysemes and it can therefore be difficult to know the exact meaning of a word, sentence, or even paragraph without the proper context around it—especially when one also takes into consideration that some grammatical structures are inherently ambiguous.

There are slightly disadvantageous rules for capitalization of proper nouns, but there are also some instances where it works in the favour of automated analysis.

In addition, there are also two official written forms of Norwegian that have similar but distinct grammar, orthography, and vocabulary. Each written form also varies depending on the dialect of the writer or the region that the writer lives in.

3.1 Capitalization and names

In the book "Skriveregler" (translation: Rules of writing) Vinje (1998) presents 19 conventions for capitalization of words in Norwegian (Haaland (2008) provides a summary of the rules in English). The conventions are, however, mostly descriptive and there are exceptions to most, if not all, of them. The main rule, however, is that proper nouns are capitalized and common names are lowercased.

However, if we look at the capitalization of organizations the rule is to only capitalize the first term in the name, for example the name "Den norske stats oljeselskap" (translation: The Norwegian State's Oil Company). Here it is only the determinant at the beginning of the name that is capitalized in Norwegian, while the rest of the terms are lowercase. However, it would be unwise to rely on this rule as it is often broken and should at this point be considered mostly a stylistic choice.

This rule is broken even by large national institutions as can be seen in the name of the Norwegian central bank "Norges Bank". If the rule had been followed, the second term should have been lowercased instead of capitalized. It could be that in this case they are trying to avoid the ambiguity between "being Norway's bank" and having the name "Norway's Bank", but the rule is broken nontheless.

This type of ambiguity does affect Norwegian, as exemplified in the difference between the sentence "Presten viser liten respekt for kirken og dens historie" and "Presten viser liten respekt for Kirken og dens historie." The only difference being the capitalization of "kirken". The translation of the first sentence would be "The priest shows little respect for the church and its history," but could both refer to a particular church, the concept of churches, or the faith it represents. The second sentence is translated in the same way, except now, it would refer to the Church of Norway instead.

Another notable rule is that titles should not be capitalized unless they refer to the institution the title represents. For example "Sysselmannen" versus "sysselmannen" (translation: the governor), the first

refer to the governmental institution and office of the governor, while the second refers to a specific person who hold the office as governor.

Though the rules like the ones for titles and capitalization of common names versus proper nouns can help models for automated analysis delineate between names and the rest of the text, Vinje (1998) shows that while capitalization is an indicator for when there is a name present, it is not enough on its own to identify all names.

3.2 Compound words

Compound words are very common in Norwegian text and account for around 10% of all words in running text (as cited by Johannessen and Hauglin, 1996). This is also true for short texts; Johannessen and Hauglin (1996) selected a random newspaper article and found 47 compounds in a 440-word article. Tough most of them already were part of the lexicon they used, as many as 12 of them were new to it. Most compound words are nouns (75%), approximately 15% are verbs, and 6% adjectives (as cited by Fjeldvig and Golden, 1985).

In Norwegian, there can be a semantic difference between two sentences if you use a compound word or use two separate words. For example, the difference between "røykfritt" and "røyk fritt", the first translates to "no smoking" while the second to "smoke freely" (Språkrådet, 2009).

This semantic difference between compound and split words can in some cases also happen to names. An organization like "Luftforsvaret" (translation: the Air Force) is the result of combining the two words "Luft" (translation: Air) and "forsvaret" (translation: Armed Forces). If we would write "Luft forsvaret" instead, it would translate to "Air out the Armed Forces". (The lowercasing of "forsvaret" is correct in this instance if we are referring to the Armed Forces in general and not the institution.)

Compound words are therefore important to consider when we

analyze Norwegian text. The immediate solution to the problem is to try and split the compound word into its individual constituents, but as we have seen, we then loose the semantic meaning of the compound.

However, there are cases where splitting compounds is actually helpful. Fjeldvig and Golden (1985) were interested in improving the usability of search for Norwegian text. They wanted to make it easier for someone searching for a general topic like "arv" (translation: inheritance) to also find documents that contain information about "*arv*eavgift" (translation: inheritance tax), "*arv*erett" (translation: inheritance regulation), or "*arv*elov" (translation: inheritance law).

Johannessen and Hauglin (1996) worked on an automatic morphosyntactic tagger for Norwegian and developed a compound analyzer to recognize the morphology of new compounds using a lexicon and rule-based approach. Though they do not actually split the word, they instead analyze what the constituents of the compound are to improve the analysis of the word and its context.

In our research on NER in section 5.3 we deal with compound words in a different way. We train a sub-word embeddings model on n-grams of words and in that way our model learns how to analyze compound words.

3.3 Polysemy and ambiguity

Like other languages, Norwegian can be a difficult language to automatically analyze semantically. Lie (1982) showed that Norwegian sentences can contain combinatory coordination over the clauses in the sentences. For example a sentence like

Det var merkelig at hun var der og han ikke så henne

can be translated into the two following sentences in English

1. It was strange that she was there and he did not see her.

2. It was strange that she was there, and it was strange that he did not see her.

This means that is was ambiguous and one cannot know, without further context, if it was strange that "she" was present or not based on this sentence.

Also simple transitive sentences can be ambiguous. Øvrelid (2004) investigated how we can disambiguate these type of sentence. For example

> Brevet skrev jenta
> The_letter wrote the_girl
> (Translation: The letter was written by the girl)

Any native speaker would instantly recognize that it was not the letter that wrote the girl, but the girl that wrote the letter. A model would have to capture the information about how a girl is different from a letter to give the correct parsing of such a sentence.

Norwegian also has many polysemes: words that mean different things in different contexts. An example would be a word like "historie" which could both be translated to "story" or "history" depending on the context (Jónsdóttir, 2003).

This also affects lemmatizations, as discussed by Johannessen et al. (2011): A word like "årene" is both the definite plural of "år" (translation: year), "åre" (translation: oar), and "åre" (translation: vein).

For our work with named entities there are also Norwegian given names that are polysemic that we need to consider. They can be quite difficult to understand without a wider context.

For example, the sentence "Bjørn er farlig" can be translated to both "Bears are dangerous" or "Bjørn is dangerous" as Bjørn can be the given name of a person as well as a designator for an animal. It could therefore be important to capture some of the context to disambiguate between the terms which are part of a name and those that are not.

In our work, we assume that this type of ambiguity does not happen that often, so we do not directly control for it. To properly control for

it, we would need to conduct a study on how Norwegian given names affect the ambiguity of the sentences they appear in.

For our Named-Entity Chunking (NEC) model, we control for problems with polysemy through adding part of the context around the word as part of the feature vector that we use for classification. For the NER model, we used sub-word and character embeddings together with a BiRNN to get a model that is better equipped at learning what it should focus on to find the correct category. However, we still have the problem that we usually only focus on the text at a sentence level and we cannot disambiguate sentences where we do not know if "Bjørn" refers to a bear or to the person named "Bjørn".

3.4 Two written forms and regional variances

Norwegian has two written forms: Nynorsk and Bokmål. Nynorsk is mostly used outside of the larger cities in the western parts of Norway, and Bokmål is used in most of the rest of the country. The reason Norway ended up with 2 written forms was that after the dissolution of the union with Denmark a growing national movement wanted Norway to have its own language instead of using the Danish written language.

Eventually two competing standards emerged through the work of Knud Knudsen and Ivar Aasen (Myking, 1997). Though Bokmål is decidedly more used than Nynorsk today, both of them are recognized as standard written forms of the Norwegian language.

The largest difference between the two language forms is that Nynorsk is based on the dialects of the common people, while Bokmål is a reformation of the Danish language into a more natural Norwegian. While the two written forms are very similar, they do differ through orthography, grammar (to some extent), and word choice. Nynorsk is reported to have a more verbal feel, while Bokmål is considered to be more formal in its expression (Brunstad, 2009).

There is no authorized standard *spoken* form of Norwegian (Sandøy,

2011) and Norway has many dialects. The dialect a person speaks can affect the spelling, grammar, and choice of words within the same written form.

For example, normally, Bokmål and Nynorsk both have three grammatical genders; female, male, and neutral. An exception is the Bergen dialect which only uses two grammatical genders: common and neutral (Bordal, 2015).

In the Bergen dialect, a name like "Tariffnemnda" (translation: The Tariff Committee) could therefore also be spelled "Tarriffnemnden" depending if the writer speaks the Bergen dialect or not.

It can also affect word order in some cases, as can be seen in the dialect from Kåfjord. They change the word order of some question types compared to the rest of the country (Westergaard, 2005).

There has been an attempt to unify the two written forms into a common form called Samnorsk, but proponents of the new form failed politically to convince users of the two written forms to adapt it (Leira, 2003).

Despite these differences, we have found evidence that the two written forms are not so different that our models cannot generalize over them given enough data. In section 5.3 we train a model for NER using a combined corpus of Nynorsk and Bokmål and by using a sub-word embedding model we are able to get better result by combining the two written forms than by training on them separately.

Chapter 4

Part-of-speech

In this chapter we present the work that we have done on Part-of-Speech (PoS) tagging of Bokmål and Nynorsk. PoS tagging is the task of finding the word class, or part-of-speech, for each word in a sentence.

PoS are categories of words that share the same grammatical properties. Examples of PoS are nouns, verbs, adjectives etc. PoS tagging can help computers reason about text by disambiguating the meaning of words in context.

This is a simpler task than finding the full grammar of a sentence, which can become very complicated. PoS is usually one of the first steps in an information extraction process that is dependent on finding data in unstructured text.

One of the reasons to have information about the PoS is to disambiguate between words that are spelled the same, but belong to different word classes. In the sentence "He held a fork in his left hand", the word "fork" is a noun, while in the sentence "Her next move will fork the king and queen", it is a verb (and will likely lead to a winning chess game).

Having a well-performing PoS tagger can help in other tasks like NER, for example in the sentence "Rusten bil" (translation: Rusty car); knowing that "Rusten" is not a noun, but an adjective can help the NER process deciding that it does not refer to the last name "Rusten".

PoS can also be a valuable tool in dependency tree parsing, relation extraction, and other natural language processing tasks. We use PoS as a feature in our NEC and NER models, explained in chapter 5.

4.1 Literature review

Velldal et al. (2017) train a model for the UDPipe tagger for Nynorsk, Bokmål, and a combined data set. They achieve a $F_{\beta=1}$ score of 97.07% for the Bokmål model, 96.80% for the Nynorsk model, and the combined model achieve a score of 96.49% on the Bokmål data set and 96.27% on the Nynorsk data.

Hagen et al. (2000) made a tagger based on a morphological constraint grammar; the Oslo–Bergen tagger (OBT). As they are interested in using the tagger as a tool for linguists to search for specific grammatical structures, OBT reports all ambiguities that it finds. They report a precision of 96.0% and a recall of 99.0%, which results in a $F_{\beta=1}$ score of 97.5%, for Bokmål (Bick et al., 2015). For tagging Nynorsk they reports a precision of 93.6% and a recall of 98.7, with a $F_{\beta=1}$ score of 96.2. However, they say they have found the correct tag if it is in the list of possibly ambiguous results that the OBT finds.

Johannessen et al. (2011) add a statistical disambiguator to the Bokmål part of OBT based on a HMM approach. They achieve an $F_{\beta=1}$ score of 96.56%, but without any ambiguities in the output.

Solberg et al. (2014) develop the first publicly available treebank for Norwegian and train a model for dependency parsing. They report an unlabeled and labeled attachment score of 92.84% and 90.31 for Bokmål, and for Nynorsk they report 92.12% and 89.54%.

Marco (2014) use the FreeLing open source text processing tool to create a PoS tagger and uses a HMM to find the tags; they achieve an $F_{\beta=1}$ score of 97.3% for Bokmål.

Using the Universal Dependency data set (Øvrelid and Hohle, 2016), Google was able to train SyntaxNet to tag Bokmål with PoS at an $F_{\beta=1}$

score of 97.44% (Google, 2016b).

Hellan and Bruland (2015) presents six applications that build on a computational grammar. One of those applications is a PoS tagger for Bokmål. The tagger calculates the PoS of a sentence by looking for known items and use the information about their inflectional properties to deduce the grammar of the rest of the sentence. They do not provide an evaluation of their tagger.

Hellan et al. (2013) use the same computational grammar to create a tool for grammatical error detection.

Andor et al. (2016) use a "simple feed-forward neural network that operates on a task-specific transition system." They achieve a average F_1 score of 97.37% over 7 languages. This is the same model that we train for Bokmål and Nynorsk in section 4.3.

Bohnet et al. (2018) develop a model for part-of-speech tagging by adding another LSTM-BiRNN on top of two LSTM-BiRNN layers: One layer acting on the character embeddings of a sentence, while the other layer acts on the word embeddings. They concatenate the output from each layer for each word and feed that to the top layer. They achieve the current best results on the CoNLL 2017 shared task (Zeman et al., 2017), with an average F_1 score of 93.40% on 54 the treebanks of 54 different languages.

4.2 Training OBT+Stat in Nynorsk

For a long time the OBT has only been able to grammatically disambiguate tokens for Bokmål. Using the Norwegian Dependency Treebank (NDT) (Solberg et al., 2014), we train a new model for the HMM based tagger, Hunpos (Halácsy et al., 2007), used in the statistical part of OBT (Johannessen et al., 2011), to learn Nynorsk. HMMs are explained in section 2.2. By using this new model, the OBT is now able to also grammatically disambiguate tokens in Nynorsk with an accuracy of 94.43% on PoS tagging.

In this section we

1. explain how the OBT+Stat tagger works in section 4.2.1.
2. show how we trained and evaluated the performance of OBT on Nynorsk and present the results in section 4.2.2.
3. discuss the results and compare them to what other researchers have achieved in section 4.2.3.
4. lastly discuss what we want to do with this research in the future in section 4.2.4.

4.2.1 OBT+Stat

OBT+Stat consists of a multi-tagger that tokenizes a text and finds all word classes a token can belong to. The tokenized text is then evaluated by a constraint grammar rule engine that removes all word classes that are impossible given the context for each sentence. The result from the rule engine is then compared with the result from Hunpos. If the rule engine and Hunpos agrees on the same tag, OBT+Stat reports that tag. If they do not agree, the most likely tag that the rule engine reported is returned instead. The statistics for the most likely tag is pre-calculated and stored in a lookup table. OBT+Stat only uses the downcased version of the current word as the observation for the HMM.

4.2.2 Evaluation

We used the NDT data set to train a new model for Nynorsk that we can use with the OBT+Stat. Part of the NDT data set was withheld to be used for evalution of the resulting model. We also used the NDT to calculate the lookup table for the statistics for the most likely tags. We show the types of tags and their distribution in table 4.2 and the distribution of the types of text in table 4.1.

Newspaper texts	82%
Government reports	7%
Parliament transcripts	6%
Blogs	5%

Table 4.1: Types of text.

Type	N	Type	N
<anf>	2145	interj	234
<komma>	11452	konj	11088
<parentes-beg>	603	prep	42991
<parentes-slutt>	597	pron	17807
<strek>	2481	sbu	9846
adj	29101	subst	74336
adv	10962	symb	132
clb	17879	ukjent	1182
det	20586	verb	45507
inf-merke	4169		
Total	303098	No. of tags	19

Table 4.2: The tags.

To train the Hunpos model we randomized the sequence of the sentences in the data set. 3/4 of the data set is used to train the model and 1/4 to verify the result.

The reason for not sampling from each of the different types of text was convenience because of time constraints. We already had a script to do this ready from previous research.

On the verification set we were able to get to an accuracy score of 94.43% when measuring on how well it performed on PoS tagging.

4.2.3 Discussion

The result of the work that we present here is that it is now possible to use the OBT+Stat to provide disambiguated PoS tags for Nynorsk.

The Nynorsk tagger does not achieve as good results as the Bokmål tagger. This is not only because the OBT for Nynorsk does not perform as well as OBT for Bokmål, but also because of the difference between how the NDT is tagged and how OBT+Stat tags data. The OBT sometimes makes mistakes on punctuation and reports an abbreviated word where there should not be one. Also, some compound phrases are tagged as one token by the OBT, but are always treated as separate tokens in the NDT corpus. In all instances of these types of mistakes, or any case where the result is different from OBT, we report an error—even in the instances where the error is because of a difference in tagging style. This means the actual accuracy could be higher than the reported accuracy.

There is also the problem that some of the words in the data set are tagged as "ukjent" (Translation: unknown). The tag is reserved for words in other languages that are not a part of Nynorsk and are therefore not considered a proper part of the Nynorsk grammar. Bokmål is considered a different language and words in Bokmål can sometimes be tagged as "ukjent". Nynorsk and Bokmål does however share many common words and inflections and can thus be a source of error.

Lastly, it could improve the tagger to include a context window

around the current token as part of the observation for the HMM. though the HMM does get information about the previous observations through the state transitions of the HMM, it is difficult for a HMM to learn long chains of dependencies. More information at the decision point could improve the results of the tagger. Since the input is already tagged by a rule-based tagger we could also use the information about the tags it reports as input to the HMM.

The OBT+Stat model is however not able to achieve the same high accuracy score as Velldal et al. (2017).

4.2.4 Future work

Future work should go into unifying how the OBT+Stat tags data and how the NDT is tagged. This would allow us to more accurately assess the performance of OBT+Stat and other taggers against each other.

We could also look at adding a context window to the input for the HMM and include the tags that the underlying rule-based tagger reports.

4.3 Training SyntaxNet to understand Bokmål and Nynorsk

We use Google's open source neural network framework, SyntaxNet, to train a fully automatic PoS tagger for Norwegian Bokmål and Nynorsk. Using SyntaxNet, we are able to get comparable results to what other taggers achieve when tagging Bokmål and Nynorsk with PoS. Both taggers are available and released as open source. This research has previously been published in Johansen (2016).

We show that using an off the shelf, state of the art, learning platform allows us to create PoS taggers for both of the Norwegian written languages that are at, or exceed, what other researchers are able to do on the same task. Both PoS taggers are available online

at `http://github.com/ljos/anna_lyse` and are released under an open source license.

For this research we

1. show how we ran the training and experiments for our taggers and present the results in section 4.3.2.
2. discuss the results and compare them to what other researchers have achieved in section 4.3.3. We also present some of the issues and problems with this way of doing PoS tagging.
3. lastly we discuss what we want to do with this research in the future in section 4.3.4.

4.3.1 SyntaxNet

SyntaxNet is a "feed-forward neural network that operates on a task-specific transition system." It is not recurrent and uses beam search together with a CRF to globally normalize the learning model. They also perform full back-propagation for all neural network parameters based on the CRF loss (Andor et al., 2016).

4.3.2 Evaluation

Recently Google released the source code to SyntaxNet, their neural network framework for syntax learning. They provide a detailed explanation for how to use SyntaxNet for learning new languages and examples on the web page for the source code for SyntaxNet (Google, 2016a). We decided to run a simple grid search using the NDT for both the Bokmål and Nynorsk version which follows the Norwegian Reference Grammar (Solberg et al., 2014).

The task we are trying to achieve is to correctly classify the PoS for every token in the data set.

The experiment followed a very simple setup. For each language we split the data set into a training set containing 50% of the sentences, a

test set containing 25%, and a verification set containing the remaining 25%. As the data set contains sentences from different sources like newspaper text, government reports, parliament transcripts, and blogs, we first randomized the order of the sentences to get a fair distribution of each type of text in each of the data sets.

We then performed a grid search of the parameters to find the best performing parameter values for training a well-performing SyntaxNet model. SyntaxNet have many different parameters that we can change, but we focused on the layer size, learning rate, and momentum as these are the ones that the people behind SyntaxNet recommends. Layer size control how many neurons are in each layer, learning rate is the value for how fast each neuron learns by acting on the weight update of the back-propagation algorithm, and momentum helps the update gradient of back-propagation keep moving in the same direction. Even though we were able to run some of these training sessions in parallel it still took many days to run through all of the variations in a grid search.

Language	$F_{\beta=1}$
Bokmål	97.54%
Nynorsk	96.83%

Table 4.3: Results of training SyntaxNet.

The results from the grid search are available in table 4.3. We were able to get an $F_{\beta=1}$ score of 97.54% for Bokmål and 96.83% for Nynorsk.

4.3.3 Discussion

As one can see from table 4.3 we have been able to get good scores for both of our taggers. If the results are compared to the other taggers we presented in section 4.1, it can be seen that we are slightly better than all of the previous attempt at creating a PoS tagger for both Bokmål and Nynorsk.

There is one caveat: comparing to the OBT is somewhat problematic as it also report ambiguities and ours do not. We argue that to compare

them one should look at the precision of OBT and not the F-score—the precision measures when OBT is able to unambiguously tell if a tag is the correct one or not.

We can also see that the tagger from Google gets almost the same score as we do; we believe this is because the Universal Dependency data set is the same data set we are using, just with a translated tag set.

There are also some problems with the current implementation of the taggers we have presented here. The first issue is that the tagger only accepts tokenized text, which means we cannot run the tagger on just plain text documents; they need to be pre-processed first.

The second problem is that it is difficult to say what we can learn about the Norwegian language from the taggers. The model that we train is a neural network and it is therefore not always possible to extract information from the model about why it is choosing the output that it selects.

Further, if one is interested in also capturing the ambiguity in the language and not only be presented with what the machine calculates to be the correct answer, the approach chosen by the OBT is better. If OBT detects a word that can be tagged in multiple different ways it will present all of them if it is not able to chose. Our approach will always choose one definite answer, correct or not.

4.3.4 Future work

SyntaxNet does not just support PoS tagging. It is also possible to train it for dependency parsing. It would be quite trivial to adapt this project to also do this, but we did not attempt it as we did not have the time to run another round of training.

There are many ways to experiment with SyntaxNet to see if it is possible to improve on the current taggers. We mostly followed the standard setup and tested different changes to the parameters that the creator behind SyntaxNet suggested. One could for example change

the features that the tagger looks at instead of just the standard set.

We also believe that the performance of the models would increase if we had larger data sets to learn from. We see evidence of this in that the biggest difference between the tagger for Bokmål and Nynorsk is that the data set for Bokmål is larger—and it performs better.

Further, this project can help improve results in NER and other chunking tasks for both Bokmål and Nynorsk by providing a more accurate base to build from.

Chapter 5

Named Entities

In this chapter we present the research we have done on finding named entities in Norwegian text. In section 5.2 we present the problem of NEC and our solution. NEC is the task of finding the sequences of names in a text. We employ SVM to train a model that can accurately detect the names in the text.

In section 5.3 we introduce the problem of NER and our solution. NER is, in its simplest form, the task of finding the type of named entity a name represents. We, like other international researchers, also have our model delineate the names from the rest of the text.

The reason for implementing both NEC and NER is that previous research on Norwegian text has only concerned itself with discovering what category a name is based on pre-chunked text, and does not demarcate the names from raw text. This is what prompted us to implement a NEC model. We wanted to see what kind of system would perform best: implementing a separate NEC model and NER model or implementing a combined approach like we do in section 5.3.

NER is the foundation of many other natural language processing tasks like Relation Extraction and Named-Entity Linking. It is also often used in combination with search to say something about who and what a large corpus mentions. It is not meant to be used for questions like "Which of these texts mentions the prime minister, Erna Solberg".

This is not a very difficult task as one can simply gather all the different ways she would be mentioned in the type of text that is being research and search for those.

What NER can do instead is to used to answer the question "Which entities are mentioned in our corpus and what are they?" or "Does the corpus mention persons or organizations the most?" or "Where is the reporting in this selection of articles placed?" Without NER these types of questions are prohibitively expensive to research manually for large corpora.

5.1 Literature review

Bick (2000) developed an early Danish NER base on constraint grammar parsing. They report an error rate of ∼5%. It is unclear how their measure relates to the more standard way of reporting accuracy with F-scores. Bick (2004) improved the first model and achieved an $F_{\beta=1}$ score of 93%. It is however unclear how they arrive at this score as they originally report on different error rates of the model and then say that these numbers translate to the given F score. They do not tell us how they translated these numbers.

Bick (2003) also used a constraint grammar approach for Portuguese. On a test corpus of ∼40000 tokens they report an $F_{\beta=1}$ score of 91.85%.

Jónsdóttir (2003) did some early work on chunking and recognition for Norwegian Bokmål. They used a ruled-based approach through the use of constraint grammar rules. The approach did provide good recall scores (>90%) for NER, but the precision did not reach satisfactory results (<50%). Jónsdóttir does not provide the corresponding numbers for their NEC.

Nøklestad (2009) and Haaland (2008) also worked on NER for Norwegian Bokmål texts. Nøklestad uses a Memory-Based Learning approach while Haaland uses Maximum Entropy Models. The main challenge with the approach implemented by Nøklestad and Haaland

is that they are dependent on previously name-chunked text to work correctly. Haaland provide a $F_{\beta=1}$ score of 81.36%, while Nøklestad achieve a score of 82.53%.

Husevåg (2016) explores the role of named entities in automatic indexing based on text in subtitles. They show that the distribution of named entities are not the same for all types of text and that Norwegian text has a significantly lower name density than English for non-fiction text. They also argue that NER is an important tool for indexing as named entities are a common search request.

Kokkinakis (2004) created a NER for Swedish and showed that they could get good results on a test corpus of 45962 tokens. They got a $F_{\beta=1}$ score of 90.50%.

Dalianis and Åström (2001) use a rule-based approach to NER for Swedish and show a $F_{\beta=1}$ score of 61%.

Mickelin (2013) also worked on NER for Swedish. They use SVMs to train their model and achieve a $F_{\beta=1}$ score of 20%.

Olsson (2008) developed a tool for annotating NER data an showed that their tool decreases the number of documents an annotator needs to review and still get good results.

Kokkinakis et al. (2014) converted and adapted the NER described by Kokkinakis (2004) to the Helsinki Finite-State Transducer Technology platform (HFST). HFST is a pattern matching tool (Karttunen, 2011). Their NER tags 8 different categories: Person, location, organization, artifact, work, event, measure, and temporal. They report a precision of 79.02%, recall of 70.56%, and a $F_{\beta=1}$ score of 74.55%.

Kapočūtė-Dzikienė et al. (2013) use CRF to train a NER model for Lithuanian. They achieve an $F_{\beta=1}$ score of 89.5%.

Chiu and Nichols (2015) implemented NER for English using LSTM-BiRNNs, and is the research that we have tried to implement for Norwegian, except that we are using sub-word embeddings, represent the character and case information differently, and work with Norwegian text instead of English. We also combine two different written forms of the same language to increase performance.

Rama et al. (2018) present a new corpus consisting of Norwegian clinical trials annotated with entities and relationships. The entities are categorized into 10 different categories, while there are 5 different categories for relationships. They build two different models, one entity extraction model and one model for relationship extraction. The entity extraction model achieves a F_1 score of 84.1%. The relation extraction model achieves a F_1 score of 76.8%. They use SVMs for both models. The entities that they describe are not all fully *named* entities. They are also interested in finding family members addressed as, for example, "bestefar" (translation: grandfather) and nouns that refer to the patient in question, such as "pasienten" (translation: the patient).

Stadsnes (2018) trained and evaluated different word embeddings models and came to the conclusion that while fastText skipgram embeddings performed better when recognizing analogies, word2vec CBOW embeddings were better for synonym extraction.

Peters et al. (2018) implemented NER for English using a novel approach they call ELMo, which "is a deep contextualized word representation that models both complex characteristics of word use (e.g. syntax and semantics) and how these uses vary across linguistic context (i.e. to model polysemy)." They achieve a $F_{\beta=1}$ score of 92.22% on English text.

5.2 Named-Entity Chunking

NEC is part of the NER process and is the task of identifying which parts of a text are names. This task is usually done as an implicit part of the recognizer, but because previous attempts at NER for Norwegian text focus only on the recognition, this research represents an attempt to develop an explicit chunker. An explicit NEC can also help to evaluate the performance of a NER model and point in a direction for how to improve such models in the future.

The research shows that if we only focus on finding names and not on

discovering their type as well, we are able to accurately (>95% F_1-score) find the names in Norwegian text using SVMs. This research was first presented in Johansen (2015).

NEC is the task of demarcating which segments of a text are parts of a named entity and which are not. A named entity is a specific person, place, event, etc. Chunking is different from the process of NER in that the objective is to find the entities and not to also find the type of the found entities.

Maurits	Escher	was	a	Dutch	artist	.

Table 5.1: An example sentence.

The example in table 5.1 illustrates the task of Named-Entity Chunking. The example contains 7 tokens, but only one entity, namely *Maurits Escher*. Chunking the sentence means to discover that the terms *Maurits* and *Escher* are part of the same named entity.

The term *Dutch* is also part of the sentence, and even though it refers to a nation, it is not considered, by itself, as an entity. *Artist* also appears, which on the other hand is an entity, but it is a general category and does not refer to a single individual (or thing) and is therefore not a *named* entity.

The reason for developing an explicit chunker instead of as an implicit part of a NER tool is that previous research on recognition for Norwegian focused on only categorizing the names in pre-chunked data (Haaland, 2008, Nøklestad, 2009). The exception is Jónsdóttir (2003), who do chunking as a part of the recognizer.

We use a SVM, as described in section 2.1, to train a model for NEC. To train and classify new examples of text, we need to convert the text to a vector representation. We convert the text by taking each token and processing the surrounding context.

This type of research is important for languages like Norwegian as it shows that these types of methods used in this research generalizes over different languages given the right feature selection and training material.

The research we present here proposes a solution to named-entity chunking in the context of Norwegian text. It does that by:

1. Running experiments to show that SVMs are able to, given the features that we have defined, provide state of the art performance on NEC. See section 5.2.2.

2. Comparing our solution to what others have done before in section 5.2.4.

1. Concluding on the results in section 5.2.5.

2. Lastly we discuss any potential future work in section 5.2.6.

5.2.1 Tagging

Kudo and Matsumoto (2001) presents 5 main ways of tagging chunks in text. For the experiment in this research, we chose the IOB2 method of tagging to demarcate the named entity chunks. It uses 3 tags to identify tokens

I A token inside a chunk.

O A token outside any chunk.

B A token at the beginning of a chunk.

The reason for choosing the IOB2 tagging scheme is because it provides more of each tag in the data set than the other systems, which makes it easier for the SVM training algorithm to learn from the data set. We could also have used the IOE2 scheme, which is equivalent to IOB2, but tags the end of chunks instead of the beginning.

The other tagging methods either only tags the ends/beginning of tags between the boundary of two chunks (IOB1/IOE1) or introduces extra tag(s) (Start/End) and therefore leaves fewer instances of some tags and gives the SVM fewer instances to learn from.

5.2.2 Experiment

During this research we did multiple experiments to test whether a difference in how the terms are categorized affects the accuracy of the classifier. We chose to use a SVM classifier as they have shown themselves to work well for text classification (Joachims, 1998).

In the example sentence in table 5.3 we show an example sentence and features for our classifier. Each word is accompanied by its PoS, lemma (the canonical form of a word), and direct translation. These data are the result of running the sentence through the OBT.

The data are used to transform each word in the sentence into a feature vector that can be used to learn a model for the data or predict new instances. Table 5.2 illustrates the features that we selected together with an example from one of the words in the sentence in table 5.3.

The features that we chose were the lemma of the current word and the 2 words on each side, the PoS of each word, if the word is capitalized, and the 4 last characters of the word.

We constrained the use of the surrounding words to two words on each side. This was because we wanted to capture some of the context for each word, but at the same time we wanted to keep the cost of training the model at a level that was possible within the available resources. We also chose to use the PoS for each word as the PoS contains hints to whether a word is part of a name or not. For example, a name is often classified as a noun instead of a proper noun by the OBT.

We use capitalization as a feature because we want to downcase the lemmas to keep the number of possible values low. The reason for keeping the last 4 characters of the word is grounded in a feature of Norwegian last names: Many Norwegian last names has the same ending, like Johansen and Evensen or Fjellheim and Norheim.

We use the same data set as Nøklestad (2009) and Haaland (2008) where the terms are tagged with their type (person, organization, etc.)

Features	Vector
lemma - 2	industri
lemma - 1	ha
lemma	ingen
lemma + 1	problem
lemma + 2	.
PoS - 2	noun
PoS - 1	verb
PoS	det
PoS + 1	noun
PoS + 2	clb
Capitalized?	0
Last 4	ngen

Table 5.2: The defined features and example vector.

PoS	noun	verb	det	noun	clb
Sentence	Industrien	har	ingen	problemer	.
Lemma	industri	ha	ingen	problem	.
Translation	The_industry	has	no	problems	.

Table 5.3: Example PoS, sentence, lemma, and direct translation.

and their grammatical class. Since the grammar in the data set was tagged with an older version of the OBT, we cleaned the data and aligned the tags with the new version of the OBT. the OBT "is a robust morphological and syntactic tagger developed at the University of Oslo and at Uni Computing in Bergen" (Tekstlaboratoriet and Uni Computing, 2014).

An overview of the data set for this research is available in table 5.4. It consists of 210 newspaper articles, 46 magazine articles and 9 works of fictions with a total of 230453 tokens (words, punctuation, symbols, etc.). There are a total of 7505 entities in the data set.

We tagged each token in the data set with the IOB2 tagging scheme. The categories with the number of tags are specified in table 5.5. Each token, together with the surrounding context, was then transformed into a feature vector.

The SVM library we used for this research does not support a string vector as input, only a sparse numerical matrix, so we built a tool to convert between our text vectors and this format.

To test different parameters of the SVM learning algorithm we did a grid search over a the variables C and γ. For the C value we used the range $2^{-5...15}$ where the power increments by 2 at each step. For γ we used the range $2^{3...-15}$ where the power decrements by 2 at each step.

For every parameter option we did a 5-fold cross validation to check the result of the learned model. To be able to accurately test the classifier after it had been learned, we also removed 20% of the training data to use for testing by randomly selecting 20% of the instances from each class. The reason for selecting randomly from each class instead of the total data set was to avoid randomly selecting only the dominating class, Outside, and ensuring that we had enough test instances for each class.

To distribute the calculations over many machines and improve the time it takes to test all combinations of C and γ we used GNU Parallel (Tange, 2011): a shell tool for executing jobs in parallel.

Resource	Sources	Tokens	Entities
Newspaper articles	210	107814	4474
Magazine articles	46	63763	1916
Works of fiction	9	58876	1115
Sum	265	230453	7505

Table 5.4: Description of data set.

Category	Count	Percent
(B)eginning	7505	3.26%
(I)nside	2583	1.12%
(O)utside	220365	95.62%
Total	230453	100.00%

Table 5.5: Number of terms in each category.

Precision	Recall	$F_{\beta=1}$
97.95	95.34	96.63

Table 5.6: Results of experiment.

5.2.3 Results

The results from the experiment are shown in table 5.6. There it can seen that the chunker has a higher precision than recall. This means that the chunker is usually correct when it reports that it has found a chunk, but that it is not able to find all chunks in a text. However, the F_1-score tells us, as the harmonic mean between the precision and recall, that the accuracy of the NEC is quite good.

To calculate the precision and recall of the system we only looked at chunks that are an exact match to the corresponding chunk in the data set. A partial match is therefore not only a false negative, but also a false positive, as the exact chunk found by the system is not found in the original data set.

We also discovered that if we removed capitalization as a feature from the training data, the recall of the chunker dropped significantly ($<$50%), but the precision stayed high ($>$95%). Taking into consideration the many capitalization rules of Norwegian, we believe this result shows that while capitalization is important for finding the start of entities, it is not as important for the following parts of the entities. The reason we say that capitalization is not as important for the following parts of the entities is that the precision of our chunker is still high, and a high precision indicates that most of the entities that our chunker finds are correct.

Another observation is that the final chunker that produced the results in table 5.6 has some problems with polysemy. The final chunker classifies some entities incorrectly and it seems like it has problems with names that are at the beginning of sentences. It could be that this is because of the polysemy of certain names in Norwegian, as discussed in chapter 3.The capitalization of names at the beginning of sentences is perhaps not a good indicator for a name since (almost) all sentences begin with a capitalized word.

5.2.4 Discussion

If we compare our chunker to the CONLL shared task in 2000, we can see that the score for our chunker is significantly better than the baseline precision (72.58%), recall (82.14) and F_1-score (77.07) (Tjong Kim Sang and Buchholz, 2000). The system even performs better than the best performing chunker from that competition (95.8%) (Kudo and Matsumoto, 2001). However, this is not a completely fair comparison as they are trying to solve *any* text chunking problem for English: Noun phrases, verb phrases, adjective phrases, etc. It is also focusing on English and not Norwegian. Despite this, we can use the number as a baseline for how a chunker should perform and can therefore conclude that our chunker is doing quite well.

Zhou and Su (2002) does equally well on chunking names in English text as we do on chunking Norwegian text with an F_1-score of 96.6%. However, their data set contains only 1330 instances, and it is therefore difficult to judge the generality of their chunker. Though the data set used for training our chunker is only moderately sized, it is still over 5 times as big at 7465 entities.

Comparing our chunker to the research from Nøklestad (2009) and Haaland (2008) is not completely appropriate as their research only works on pre-chunked text. They need the names in the text to be already picked out and then uses the surrounding context to discover the type of entity.

The only candidate that we know of that goes from untagged Norwegian text to NER is the work by Jónsdóttir (2003) and they report a precision of 45%, recall of 92% and a final $F_{\beta=1}$-score of 60%. If we compare this score with the F_1-score of our chunker we can see that our chunker is more precise (98%), offers better recall (95%), and is therefore also more accurate (97%). However, this comparison is also problematic since theoretically their chunker could perform perfectly and the loss in precision is from the recognizer. We still provide the comparison as it is the only work on Norwegian text that is close to our

research.

The results show that by using SVMs we are able to accurately (>95% F_1-score) find names in Norwegian text and that if one is interested in finding *just* the names in a text and not their type, it is better to implement an explicit chunker.

5.2.5 Conclusion

With this research, we have shown that we can effectively use SVMs to train a NEC model for Bokmål. Based on the result on our test data we can say that when the model is usually correct when it classifies a sequence of text as a named entity and that it is able to find most names in the text. However, because of a limited data set and the nature of the algorithm we have used, we cannot say that this will hold for a new corpus.

"By the nature of the algorithm" we mean that an SVM learn "offline." After training the SVM, the model is fixed and cannot learn from exposing it to new text. We would need to tag and provide new examples to the training set and rerun the training set to teach the model about new instances. If there is a token in a new text that the SVM has not seen before, it will have to discard that token as a component of the feature vector that it uses for classification. This is basically how most Natural Language Processing (NLP) models work.

Hopefully, we have provided enough information in the training examples and the feature vector for the classifier to make a decision on new examples where it does not know how to represent a token.

5.2.6 Future work

A problem with this approach is that it does not distinguish between different types of names and it cannot tell the difference between, for example, the name of a person and the name of an organization. This is however the focus of the research presented in section 5.3.

As mentioned in section 5.1, there are several research papers describing approaches to NER, but they need the named entities in the text to be pre-tagged and will therefore not work with untagged data. A future path for this research would be to use our NEC chunker as a pre-processing step for the already developed NER systems and see if we are able to do streaming NER on live data.

In chapter 3 we identified some characteristics of Norwegian text that applies to the task of NEC. Unfortunately, we were unable to find a good way to include what we learned into the feature vector for our chunker. We have seen that polysemy and the many capitalization rules of Norwegian does have an affect on the accuracy of our research. In the future we should try to find ways to use these characteristics to improve the accuracy of the chunker.

5.3 Named-Entity Recognition

NER is the task of recognizing and demarcating the parts of a document that are part of a name and which type of name it is. We use 4 different categories of names: Locations (LOC), miscellaneous (MISC), organizations (ORG), and persons (PER). Even though we employ state of the art methods—including sub-word embeddings—that work well for English, we are unable to reproduce the same success for the Norwegian written forms. However, our model performs better than any previous research on Norwegian text. This study also represents the first NER for Nynorsk.

We also find that when we train on a combined corpus of Nynorsk and Bokmål, which we call Helnorsk, we get significantly better results (+5 percentage points) than if we train the models separately. We believe that this shows us, together with evidence provided by Velldal et al. (2017), that it is possible to use the similarities in the two written forms to produce better models than we would otherwise be able to when the models are trained separately. We discuss this further in

section 5.3.5 and 5.3.6.

Previous research on NER for Norwegian has chosen a more granular approach to the categories of names and have included the categories "works" and "events". The reason we chose to exclude these two categories was firstly that international research on English and other languages mainly focus on the same categories as us—that means that it is easier for us to compare our research to what is being done for other languages.

Secondly, previous research on Norwegian NER does not implement the same type of model that we and international researchers have implemented. They focus solely on the task of recognizing what type of name an already segmented name is categorized as. Our research also includes the segmentation of the names as well. This makes it difficult to compare our research directly with theirs.

It would also prevent us from using the NER directly on new documents if we wanted to build new research on top of such a NER model. We would have to first segment the text through a NEC and then run the NER on the result from the NEC. As a result from our work in section 5.2, there now is a NEC available that performs quite well (>95% $F_{\beta=1}$ score) However, we want to see how well a model that use state-of-the-art algorithms developed for English will perform on Norwegian. These algorithms usually do chunking as an implicit step of the NER process.

In our study we show that our model performs better than all previous attempts at a Bokmål NER ($> +5$ percentage points). There are no other NER models for Nynorsk that we are aware of. We show that by combining Nynorsk and Bokmål, into what we call Helnorsk in our study, we get better results than if we train separate models for the two written forms. "Helnorsk" translates to "All of Norwegian", which is fitting as it combines both of the official written forms.

The steps we take to present our study are to

1. Introduce a new corpus which is tagged with named entities and their types.
2. Develop a sub-word embedding model for Nynorsk, Bokmål, and Helnorsk.
3. Implement a deep learning system designed to train a NER model based on a state-of-the-art English model.
4. Run experiments on Bokmål, Nynorsk, and Helnorsk to show how the model performs.
5. Discuss the results of the experiments.
6. Conclude on what we believe the experiments show us.
7. Present future research that we believe should be explored to answer some of the questions that we found at the end of this study.

5.3.1 Corpus

We introduce a newly tagged corpus with named entities for the task of NER of Norwegian text. It is a version of the Universal Dependency (UD) Treebank for both Bokmål and Nynorsk (UDN) where we tagged all proper names with their type according to our tagging scheme. UDN is a converted version of the Norwegian Dependency Treebank into the UD scheme (Øvrelid and Hohle, 2016).

Table 5.7 shows the distribution of the different types of text in the corpus. It consists of 82% newspaper texts, 7% government reports, 6% parliament transcripts, and 5% blogs (Solberg et al., 2014). Table 5.8 shows the number of names for each of the categories that the corpus has been tagged with. We chose to tag it with the same categories as the CONLL-2003 shared task for language-independent NER (Tjong Kim Sang and Buchholz, 2000): Location (LOC), miscellaneous (MISC), organization (ORG), and person (PER).

Resource	Percentage
Newspaper texts	82
Government reports	7
Parliament transcripts	6
Blogs	5

Table 5.7: Description of data set.

Bokmål	Tokens	LOC	MISC	ORG	PER	Total
Training	243894	3241	498	3082	4113	10934
Development	36369	409	113	476	617	1615
Test	29966	420	90	317	564	1391
Total	310229	4070	701	3875	5294	13940

Nynorsk		LOC	MISC	ORG	PER	Total
Training	245330	3482	588	2601	3992	10663
Development	31250	340	67	268	421	1096
Test	24773	300	59	246	362	967
Total	301353	4122	714	3115	4775	12726

Helnorsk		LOC	MISC	ORG	PER	Total
Training	489224	6723	1086	5683	8105	21597
Development	67619	749	180	744	1038	2711
Test	54739	720	149	563	926	2358
Total	611582	8192	1415	6990	10069	26666

Table 5.8: Number of names for each data set.

We chose this scheme despite previous research on NER for Norwegian text has chosen a more granular approach (e.g. Haaland (2008), Jónsdóttir (2003), Nøklestad (2009)) This meant that we are to be able to more easily compare our NER tagger to taggers developed for English. Previous research studies on Norwegian text are also not solving the exact same problem as we are investigating for our research. They focus solely on categorizing named entities and do not also delineate them from the text at the same time. Having fewer categories also meant that an annotator could perform the tagging faster as there were fewer choices to make when they decided the category of a name.

There are however some problems with the corpus. The corpus has only been tagged by one annotator in one pass. This means that there are probably mistakes which will affect the performance of the trained models. The type of deep learning model that is train for this research can never be better than the input it receives. After some investigation of the data set, we also decided to trust that all named entities were tagged in the original UDN corpus with the PROPN (proper noun) tag. It is entirely possible that some of names are only tagged as nouns, further degrading the performance. During the tagging we noted that, especially for the Nynorsk part of the corpus, not all parts of a name were always tagged as a proper noun. This is not necessarily wrong in a grammatical sense, but it does mean that the two written forms follow a slightly different grammatical tagging schema. Since the tagging was quite time consuming, we did not have time to investigate further or try to figure out how to correct any mistakes that were made in the named-entity or PoS tagging.

5.3.2 Method

For the NER tagger we chose to use the BIOES tagging scheme— in contrast to our work on NEC where we chose the IOB2 tagging scheme— as other researchers report that the BIOES tagging scheme performs (marginally) better on this type of task (Lample et al., 2016). The

BIOES tagging scheme uses 5 different tags, instead of the 3 of the IOB2 scheme. The tags are

B A token at **beginning** of a sequence.

I A token **inside** of a sequence.

O A token **outside** of a sequence.

E A token at the **end** of a sequence.

S A **single** token representing a full sequence.

We tagged each of the tokens in our corpus with one of these tags and the corresponding class of that token. There is an example in table 5.9.

O	O	O	O	O	B-PER	I-PER	E-PER
Folk	er	så	opptatt	av	Karl	Ove	Knausgård
People	are	so	occupied	with	Karl	Ove	Knausgård

Table 5.9: Example of tagging a sequence that mentions a person.

We then trained a CBOW and a skipgram embedding model for each of the language forms: Nynorsk, Bokmål, and Helnorsk. The models were trained on a cleaned and combined corpus consisting of texts from Wikipedia, the Norwegian News Corpus (Andersen and Hofland, 2012), and the Norwegian Dependency Treebank (Solberg et al., 2014). We used fastText to train the sub-word embeddings with a vectors size of 300 components with a minimum n-gram size of 2 and maximum of 5 for the sub-words (Bojanowski et al., 2017).

We created a gazetteer from the NER corpus by extracting all words that appear as part of a name in the corpus. The gazetteer is used as part of the input to the model so the model can tell if a token has been used as part of a name in the past.

The model that we use is a LSTM BiRNN, as described in section 2.4.3, and it is trained on sentences that we treat as sequences of words.

For each word in the sequence, we create an input vector that consists of the sub-word embedding of the word, membership in the gazetteer, the sequence of the characters of the word, and the part-of-speech of the word.

We also train a character embedding as part of the model. The character embeddings for each character in a word is run through a convolutional layer, and the output of the convolutional layer is pooled together by selecting the maximum value for each position in the vector from the character embeddings. The convolutional layer is activated by the ReLU function.

We use the sub-word embeddings, the part-of-speech, gazetteer information, and the pooled character embeddings as the input to the BiRNN layer.

The output of the BiRNN layer is then fed to a dense layer that reduces the dimensionality of the output from the BiRNN down to the number of tags in our vocabulary. The output of the dense layer is fed to a CRF, that we use to calculate the log likelihoods of the predicted tags. We then use the CRF to calculate the most likely sequence given the evidence we have seen.

We train the model using the Adam optimizing algorithm. The implementation of the model is presented in appendix A.1. We did do some manual testing of the training parameters, but because of time constraints we ended up using the hyperparameter configuration in table 5.10 as those were giving us the best results for the values that were tested.

For each model we set a batch size of 100, a character embedding size of 25, the convolution kernel was 3, the max pooling of the convolution run was set to 53 wide and the RNN depth was 1. The dropout between layers was 50% and the hidden size of the RNN was 256 neurons. The learning rate for the ADAM optimizing algorithm was 0.01.

Variable	Value
Batch size	100
Char. embed size	25
Conv. kernel	3
Pool size	53
Depth	1
Dropout	0.5
RNN hidden size	256
Learning rate	0.01

Table 5.10: Hyperparameter configuration of the model training.

5.3.3 Results

The results from training the different models are displayed in table 5.11. We trained 4 different models. One for Bokmål, Nynorsk, and Helnorsk using the CBOW embedding model. It shows that the combined Helnorsk model performs better than either of the models trained on a single written form by ∼5 percentage points (p.p.) over both forms. We then trained a skipgram model for Helnorsk which performs ∼5 p.p. above the CBOW Helnorsk model.

In the end we end up with a $F_{\beta=1}$ score of 86.73%, with a precision of 87.22% and recall of 86.25% for the combined written form. The model performs slightly better on Bokmål with an $F_{\beta=1}$ score of 87.20%, precision of 87.93%, and recall of 86.48%. The same model has an $F_{\beta=1}$ score of 86.06% for Nynorsk, 86.20% precision, and 85.93% recall.

In table 5.12 the pr. name category results are displayed. There, it can be seen that it is especially the miscellaneous (MISC) category that through its recall score is driving the results down with a score of 42.95%. The precision is also low with a score of 73.56%.

The organisation (ORG) category also performs worse than the total score with an $F_{\beta=1}$ score of 81.31%. It is the location (LOC) category, with a $F_{\beta=1}$ score of 89.44%, and especially the person (PER) category with a $F_{\beta=1}$ score of 92.04%, that is pushing the over all score upwards.

5.3.4 Discussion

When comparing the results from our research with that of the other research that has been done on the Norwegian written forms, it is evident that our model performs significantly better than what has been shown before.

Haaland (2008) and Nøklestad (2009) shows a $F_{\beta=1}$ score of 81.36% and 82.53%, respectively, for Bokmål and we have a score of 87.20%; almost 5 p.p improvement over their results. However, the comparison is not completely fair. They only try to categorize already segmented names. Our research segments and categorizes the text as part of the

Written form	Precision	Recall	$F_{\beta=1}$
Bokmål, CBOW	80.03	73.47	76.61
Nynorsk, CBOW	77.86	68.04	72.62
Helnorsk, CBOW	84.42	76.33	80.17
Sam. Bokmål, CBOW	87.06	77.42	81.96
Sam. Nynorsk, CBOW	80.78	74.76	77.65
Helnorsk, SG	87.22	86.25	86.73
Sam. Bokmål, SG	87.93	86.48	87.20
Sam. Nynorsk, SG	86.20	85.93	86.06

Table 5.11: Results of NER experiments.

		Nynorsk	Bokmål	Helnorsk
LOC	Precision	87.98	89.55	88.89
	Recall	90.33	89.76	90.00
	$F_{\beta=1}$	89.14	89.65	89.44
ORG	Precision	81.63	80.06	80.74
	Recall	81.30	82.33	81.88
	$F_{\beta=1}$	81.46	81.18	81.31
MISC	Precision	71.88	74.54	73.56
	Recall	38.98	45.56	42.95
	$F_{\beta=1}$	50.54	56.55	54.23
PER	Precision	88.91	92.58	91.11
	Recall	93.09	92.90	92.98
	$F_{\beta=1}$	90.96	92.74	92.04

Table 5.12: Pr. name precision, recall, and F_1 score for the best performing Helnorsk model.

same process.

Jónsdóttir (2003) shows a $F_{\beta=1}$ score of 60%. We cannot boast of the same precision that they have (90%) for Bokmål, but we are close with 87.93%. They do not provide any results for Nynorsk.

Rama et al. (2018) developed an entity extraction model based on SVMs and got a $F_{\beta=1}$ score of 84.1% on a corpus of clinical texts. They are interested in finding nouns, and not only named entities, such as "bestefaren" (translation: the grandfather), and it is therefore difficult to compare our study with theirs.

Chiu and Nichols (2015) achieves a $F_{\beta=1}$ score of 91.62% on the CoNLL-2003 data set and 86.28% on the OntoNotes data set. Both are English data sets. The CoNLL-2003 data set is somewhat comparable to our data set with 35089 entities over 302811 tokens (Tjong Kim Sang and De Meulder, 2003), while ours is 26666 entities over 611582 tokens for the Helnorsk data set. The OntoNotes data set is 104151 over 1388955 tokens and is much larger than the data set we have available for Norwegian. We see here that the ratio between tokens and entities in OntoNotes is ∼7%, and in CoNLL-2003 it is ∼12%, while for the Helnorsk data the ratio is ∼4%.

This supports the conclusion by Husevåg (2016) that Norwegian has a much lower density of named entities compared to English. Since deep learning models require large amounts of data to generalize effectively over the data set, it is possible that this is a problem for training a model for NER on Norwegian text.

We saw in table 5.12 that the worst performing name category is the miscellaneous category. This is also the category with the fewest names, showing us that lower amounts of data gives us worse performance. If one looks at how many names there are for each category, in table 5.8, and compare to the performance on each category, it shows that the score is higher if there are more examples of names.

Peters et al. (2018) is the latest state-of-the-art NER for English, as of writing, and achieves a $F_{\beta=1}$ score of 92.22% on the CoNLL-2003 data set. Though we are not able to reach the same score, we are only

trailing by ∼5 p.p. Right now, there are many avenues to try out for research on Norwegian text to reduce that gap. In section 5.3.6 we discuss the ideas that we believe are the most promising and the most immediate.

5.3.5 Conclusion

The results of this research show that it is possible to train a deep learning model to learn how to find named entities in Norwegian text and reach close to (∼5 p.p.) the results of state-of-the-art models for English text. Our model achives a $F_{\beta=1}$ score of 86.73 on the combined Bokmål and Nynorsk corpus.

We also show that it is plausible that Norwegian is harder to train for NER because Norwegian has a lower density of named entities compared to English, as shown by Husevåg (2016) and corroborated by evidence presented in our research. However, this requires further research to arrive at a conclusion.

We also show that we can get better performing models for both the written forms, Bokmål and Nynorsk, if we use (sub)word embeddings and train on a combined data set instead of training a separate model for each written form of the language. We do not know if this way of combining Nynorsk and Bokmål into one training set will transfer to other natural language tasks.

We do see some problems like a worse result for Nynorsk compared to Bokmål, which we cannot immediately explain. However, Velldal et al. (2017) has shown similar results as us when they trained a PoS tagger using a combined corpus instead of treating the two written forms as distinct languages.

5.3.6 Future work

There are many possible avenues for improving on this research in the future. The first thing we would like to try would be to do a

hyperparameter search to see if there are other parameter settings that could improve the results further.

Next, we should investigate if we can train and use the ELMo embeddings presented by Peters et al. (2018) for Norwegian. They report a relative increase of 21% on NER for English using their new embedding model.

More time should be spent on analyzing and cleaning the corpus. For now, only 1 annotator has gone through and annotated the data set with NER tags.

We would also like to investigate why the miscellaneous category is performing so much worse than the other categories. This could be because we have more mistakes there or that the category is too broad; and it is difficult for the model to find a good delineation between the names in the category and the rest of the corpus.

We would also like to further test the hypothesis that a model trained on both written forms performs better than if we train two separate models. Is it just because we have more training data, and despite introducing noise, it performs better; or is it the model that is able to generalize better over the wider data set? Does the performance increase hold for other natural language processing tasks? Is it just Nynorsk and Bokmål that exhibits this behavior, or can we include other similar languages like Swedish and Danish? How close do the languages have to be to show this type of performance increase?

We hope to investigate more of these ideas in the future.

Chapter 6

Case 1: The thematic structures of news stories

We use a semi-automatic method for discovering the thematic structure of a highly polarized news stories using Social Network Analysis. We do that by connecting the named entities that appear in the text by the newspaper articles they appear together in; and analyze the resulting graph. The news story we are investigating as a case study is the story about undergoing a study of the consequences of drilling for oil in the Loften, Vesterålen, and Senja region (LoVeSe). We find 6 different groups that we believe represent the different themes of the story.

By a theme we mean a subject matter that is general to the type of story we are investigating. In our case, we are investigating a news story about oil drilling and politics. We should therefore expect to see themes such as "environmental concerns", "the political left", and "the political right".

The purpose of the research is to see if we are able to detect the thematic structure in a highly polarized news story by analyzing the graph between the entities that appear in the newspaper articles covering the story. We use NEC to find the names in the articles and network analysis to analyze the relationship between the entities in the news story.

We gathered all the newspaper articles that were released in 2013 about a highly polarized political topic in Norway: Oil drilling in the Lofoten, Vesterålen, and Senja region. LoVeSe are 3 areas in northern Norway that, though potentially very rich in oil, has never been explored for oil deposits. Before the Norwegian election in 2013, the current government proposed that Norway should perform a study of the consequences of drilling for oil in LoVeSe. This spurred a debate between the political parties in Norway and became an important part of the election campaigns in 2013. More information about the news story can be found in section 6.1.

We base the research on the observation that a single newspaper article, that is written in the context of a wider story, will often concern itself with a specific theme within that story (Allern, 2015, p. 196). Allern (2015, p. 50) says that journalists will often try to give a balanced view of a particular theme within a story by including multiple sources that will have an opinion on the theme of an article. The journalist will then, ideally, follow up with another article where they describe another angle or theme of the story.

If we can assume that this is generally true of articles written for a news story, it means that if we look at which entities appear in the same articles we can get an overview of the thematic structure of the news story and how the groups within the themes of the story interrelate. We will also be able to say something about which entities are the most important for a given theme, and by investigating who or what these entities are, we can give an educated guess as to which specific theme the structure represents.

Our research shows that we are able, in this case study, to discover the thematic structure in the story by using Social Network Analysis on the network that is created by the participants appearing in the same articles. We do that by:

1. Introducing a case study on the LoVeSe news story.

2. Showing that the case study contains some interesting features that we can use to identify the themes in the news story.

3. Concluding on the result and discussing any future work.

6.1 LoVeSe

We chose the news story about oil drilling in Lofoten, Vesterålen, and Senja because it is a highly polarized topic with two clearly defined groups: For or against oil drilling. The debate has been ongoing since the 1970s and was also discussed before the election in 2009 (Hjorthen and Kjølleberg, 2009). Our case study only focuses on the year before the election in the autumn of 2013. A more detailed description of the data set can be found in section 6.2.

At the time of writing, the discussion is not about starting to drill for oil in LoVeSe, but whether to allow a consequence study of what would and could happen if Norway started to drill for oil in the region.

The opposition argues that any chance of disturbing the ecologically delicate region and the spawning ground for the Atlantic cod is too risky, and that therefore a consequence study is not needed. The Atlantic cod is an important resource for the region and any income from oil, though perhaps substantial in the short term, is not worth potentially losing a the long term of a sustainable cod fishing industry. The identity of the region is also strongly tied to the cod fishing industry as the people in the region have been fishing and drying cod there for more than 1000 years. It has been, and continues to be, an important source of income for both the region and the country.

The proponents of the consequence study says that it is not possible to know if it is impossible to drill for oil in northern Norway just because there is a delicate ecology unless a study is performed. They argue that it might be possible to prevent any large scale ecological disaster

trough careful management and close inspection. They point to the low accident rate of the Norwegian oil industry as evidence for the Norwegian oil industry's ability to protect the environment from a potential ecological disaster.

The majority of the politically elected officials are for carrying through the consequence study; it is only a vocal minority of the politicians that are against it. The minority consists of several smaller political parties in the middle of the political spectrum. Because of the political climate in Norway, both the left and the right are dependent on the middle minority to get a majority for their politics. This means that the middle minority is able to block the study as this is an important issue for them. They can also use the issue as a bargaining tool in negotiations with the left or the right—depending on who has the most power or the most to offer at the moment.

6.2 Method

During the research, we did several experiments. This section describes the setup for those experiments and a description of the corpus that we used.

The data set consists of 984 articles selected from the online edition of the largest newspapers in Norway in the period from 2012—2013: Adresseavisen (AA), Aftenposten (AP), Bergens Tidende (BT), Dagbladet (DA), Fædrelandsvennen (FV), Nordlys (NL), Stavanger aftenblad (SA), and Verdens Gang (VG). The selection process consisted of downloading all newspapers publish in the period and using the search term `(lofot* OR vesterål* OR senja) AND (olje* OR konsekvens* OR petroleum* OR vern)` . The search returns 1685 articles. We then manually check all the matching articles to select only the articles that are reporting directly on the subject.

A problem with just searching for the articles answering to the search string is that for example, the leader of the Social Liberal Party

(V), Trine Skei Grande, would, during the pre-election period in any interview setting, start talking about LoVeSe. This means that an article that is mainly about health care could also contain a quote about LoVeSe from Trine Skei Grande.

We could have used an automatic approach like topic modelling, but a manual approach only required light skim reading of each article and made it possible for us to become more familiar with the corpus. Since we are not aware of anyone else who have tried this type of approach on Norwegian news text, we believe it was important to have a familiarity with the corpus that we would not get from more automated approaches. In the future we believe that it would be possible to (nearly) fully automate this process.

On the remaining articles we used NEC to pick out all of the names which resulted in 36069 names where 4170 of them are unique.

Newspapers	8
Articles	984
Names	36069
Unique	4170
Edges	281356

Table 6.1: Overview of the pre-processed data

Some of these names could still be for the same entity. Like "Ola Borten Moe" and "Borten Moe". We are able to take this into account when creating the graph. Within the same article we can say that if the tokens of one entity is the same as the end tokens of a different entity that they are the same entity. We can not, however, do this over different articles because people may have the same last name.

If we find "Ola Borten" we do not say that he is the same as "Borten Moe". There is also the problem that we don't distinguishing between the same name referring to different entities in different articles. We assume that any name that is equal to a name in a different article is the same name. This could potentially be a problem as it is common for a person in Norway to use the name of the place they are from as their

last name. However, since we are selecting the name with the longest sequence of terms for each name in each article—and that journalists writing articles try to distinguish between the names of entities within articles—we believe that this will not be a big problem for our data set.

Based on the data for each article we created a graph where the edges of the graph denote that they appear in an article together. The edges are weighted with how many times they appear together in the whole corpus. There are 281356 edges in the graph.

Some problems that we found are that not all of the articles in the corpus are solely about the LoVeSe news story. The NEC also introduces some noise into the graph as not every name it finds is actually an entity.

There are also some names that are found that are not part of the story itself. This includes the author and photographer of the article and the names of the newspaper it appeared in. We try to remove these names as best as we can. We also merge the different ways of writing the names of well known politicians and political parties.

Since we are expecting entities that represent the same theme to appear together, we believe that the main groups that appear in the news story should also represent the major themes of the story.

As we are only interested in the main groups, we remove all but the 1% highest weighted edges. The weight of an edge is the amount of times those nodes appear together in the corpus. We also only keep the 1% strongest nodes, leaving us with 94 nodes and 1467 edges. This does not only remove minor actors that could attribute to noise in the graph, but also removes noise introduced by misclassification by the NEC. This is because even though there are potentially many misclassifications, each misclassification appears rarely and is therefore weakly connected.

The graph we created has a very high degree of connectivity, which means that it is not easy to distinguish between the groups. We believe this is because the case study is specific enough that some nodes will be mentioned in most settings—like *Lofoten*.

To combat this problem, and get groups that have a better separation,

we used violator removal. This is a technique that iteratively determines which node has the worst impact on the modularity of the grouping it has found, removes that one from the graph, and re-runs the algorithm. The modularity score is a measure of the separation of the groups in the graph found through community detection.

In the end we have created a graph consisting of 76 nodes and 321 edges. Through community detection, we find 6 different groups in the network. We then identify the names through searches in the corpus, to understand the context around how they appear in the text, and select a label for each group.

We believe that the 6 groups and the interactions between them represent the thematic structure of the news story; as identified (in aggregate) by the journalists writing the articles.

6.3 Results

Some nodes are highly connected to the other nodes in our graph. Because of the subject of our case study we find that nodes such as *Lofoten*, *Vesterålen*, and *Senja* are connected to most other nodes. This means that we do not get a very good separation between the groups in the graph. We use violator removal to find the nodes that have the most impact on modularity and our ability to separate the groups in the network. In figure 6.1, you can see elbow plot over the modularity gain depending on how many nodes are removed. We end up removing the following nodes from the graph:

> Lofoten, Vesterålen, Senja, Stortinget, H, Ap, Norge, Frp, Sv, Jens Stoltenberg, Krf, V, Sp, Oslo, Nord-Norge, Erna Solberg, Ola Borten Moe, and Nordland.

These nodes are highly connected nodes that contribute negatively to the modularity of the communities in the graph. In other words, it

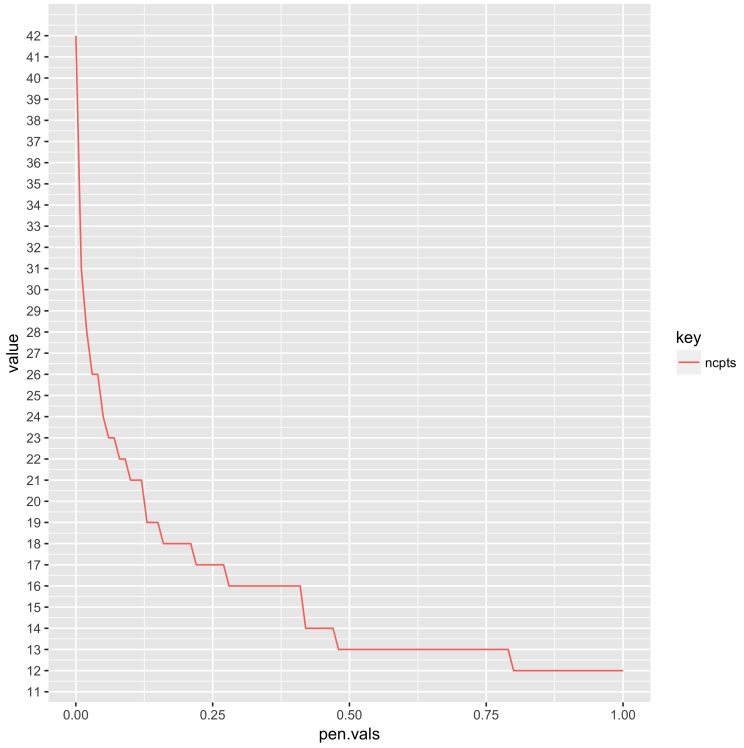

Figure 6.1: Elbow plot of the violator removal process.

makes the distance between the groups smaller and more difficult to detect. Removing them means that we can more effectively delineate between the groups.

We are not saying that these entities are unimportant for the different groups that appear in the news story, or that they are not important for the story. On the contrary, we expect them to be highly important for the discussion as they appear together with more of the other entities then any other nodes. One of the people we remove is "Ola Borten Moe" who was the minister of Petroleum and Energy in the time period of the corpus. The reason we are removing the highly connected nodes is because these nodes are so connected in their network that it is not possible to distinguish which group they belong to—or perhaps they belong to more than one or all groups.

We then run community detection on the final graph, which gives us the groups that are described in table 6.2. We can see that we have 6 different groups that represent different themes within the LoVeSe news story. The theme for each groups is

1. Oil production
2. Job creation
3. The political left
4. The political right
5. Oil politics
6. Environment

The label for each theme is based on our understanding of the corpus and by looking up the entities in the articles and in other sources to decide what theme the group represents.

We then contract the nodes in each of the groups into one node. You can see the resulting graph in figure 6.2.

From the network over the groups, we can use the page-rank algorithm to calculate the importance of each of the groups in the news

1 - Oil production	Importance:	0.25
Barentshavet	Fn	Statoil
Eu	Nordsjøen	Usa
Sverige	Nho	Norskehavet
Regjeringen	Europa	Jan Mayen
Oljedirektoratet	Mørefeltene	Kina
Petoro	Skagerrak	Rystad Energy

2 - Job creation	Importance:	0.20
Bergen	Stavanger	Tromsø
Trondheim	Bodø	Svolvær
Troms	Finnmark	Harstad
Fagforbundet	Gerd Kristiansen	Sør-Trøndelag
Vestlandet	Lo	Roar Flåthen

3 - Political left	Importance:	0.14
Mdg	Marit Arnstad	Audun Lysbakken
Kristin Halvorsen	Liv Signe Navarsete	Rødt
Bård Vegar Solhjell	Sandra Borch	Senterungdommen
Trygve Slagsvold Vedum	Rasmus Hansson	

4 - Political right	Importance:	0.15
Hans Olav Syversen	Jan Tore Sanner	Knut Arild Hareide
Nydalen	Siv Jensen	Trine Skei Grande
Bent Høie	Ola Elvestuen	Per Sandberg
Ketil Solvik-Olsen	Terje Breivik	Arne Strand

5 - Oil politics	Importance:	0.13
Nordland VI	Eskil Pedersen	Troms II
Auf	Helga Pedersen	Sogn og Fjordane
Sogn	Akershus	Hordaland
Møre og Romsdal	Trond Giske	Rogaland
Nordland VII		

6 - Environmental	Importance:	0.12
Natur og Ungdom	Silje Lundberg	Lars Haltbrekken
Bellona	Kjell Ingolf Ropstad	Frederic Hauge
Naturvernforbundet		

Table 6.2: The groups we found in the network.

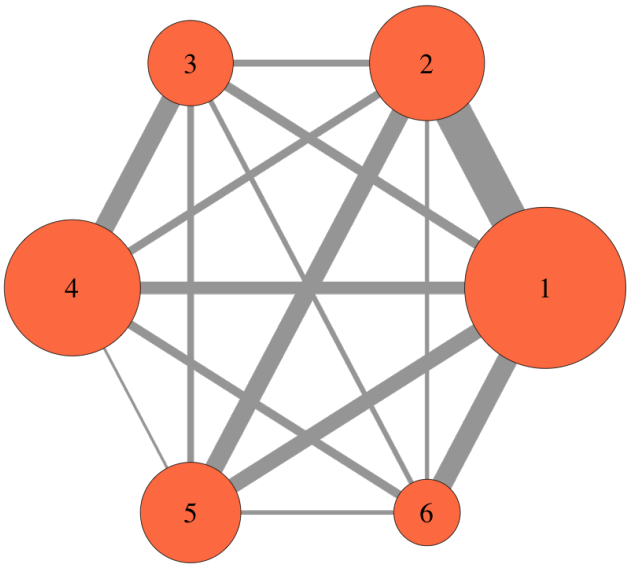

Figure 6.2: Graphical representation of the network between the groups.

story. We can see that the most important theme is oil production. The second most important is job creation, while each of the political themes; the left, right, and general oil politics, are about at the same level of importance. The least important is the environmental theme.

6.4 Discussion

If one takes a closer look at the groups in the news story it can be noted that even though the controversy in the news story is the environmental impact of oil drilling in the LoVeSe region, it is the environmental theme that is the least important to the story according to our findings. We are not able to determine what the reason for this would be from looking only at the graph. One reason could be that the entities in the environmental group have such a clear message that the newspapers spend less time on those views. Using page rank means that to be evaluated as important in the graph the node has to be connected strongly to many nodes.

In the environmental group we find entities like "Natur og Ungdom" (NU), known in English as "Young Friends of the Earth Norway". An environmental group youth organisation. Silje Lundberg; the leader of NU at the time. Lars Haltbrekken; leader of the Norwegian Society for the Conservation of Nature (Naturvernforbundet) and the organisation. Fredric Hauge; the leader of Bellona, and the organisation. The person that sticks out is Kjell Ingolf Ropstad, which was the environmental spokesperson for the Christian Democratic Party (Krf). Krf had a very strong position on the issue and said:

> Krf kan ikke sitte i en regjering som åpner for oljeboring i Lofoten, Vesterålen, og Senja – Kjell Ingolf Ropstad to Klassekampen.
>
> Translation: Krf can not be part of a government that opens up for the drilling of oil in Lofoten, Vesterålen, and Senja.

Another party that had a similar and as strong opinion was "Venstre" (V), the Liberal Party, and we might have expected to see a representative from that party in the environmental group as well. However, after investigating the corpus we found that Ropstad spent more of his time criticising the incumbent government policies on LoVeSe, and thereby prompting the environmentalists to give supporting statements or otherwise comment on the issue. For the representatives from V, that we can find in the graph, Trine Skei Grande, leader; Ola Elvestuen, first deputy leader; and Terje Breivik, second deputy leader, they had a different agenda. Though the protection of LoVeSe was an important issue for V, they would always connect the issue to either the negotiation for the new government or to other issues important to their collaborating partners.

A possible hypothesis that explains the observation is that Ropstad is setting up a front towards the incumbent government, while V (and the other members of Krf) are using the issue as a point of contention in the discussion of the new government.

Supporting evidence for the hypothesis is that Miljøpartiet De Grønne (Mdg), The Green (environmental) party, together with their leader Rasmus Hansson, is also not a part of the environmental group. They were actually at the time criticized for being too elusive on environmental questions by Bellona. They appear together with the group we have labelled as the political left—which includes the incumbent government. As Mdg was rising in the polls, it could mean that they were positioned as a part of a new alliance for a government on the left by the journalists.

6.5 Conclusion and Future Work

As we have shown, we are able to detect the groups in the case, but not fully automatically. As we saw in section 6.3, we had to remove some of the locations and other entities from the data set that introduced noise.

Some of the entities that we removed could have been prevented by using Named-Entity Recognition (NER) instead of chunking so we would be able to tell names of places from names of people and organizations. We also saw that the chunker tends to make some mistakes and is not as accurate as we would like.

Despite these limitations, we were able to find 6 different groups that we believe represent the different themes of the news story. To figure out which specific theme the different groups represent we had to manually analyze parts of the corpus—as is common with automated methods.

However, the research all depend on the assumption that in an article, a journalist will, generally, present a theme of a bigger story, and that the subjects of the story group into these themes. We could not find empirical verification that this is actually how journalists work. We based our work on a normative description of how journalists work and then found structures that match that description. We then describe what we have found through investigating the corpus in relation to those structures, but the argument that we have actually found the thematic structures that we believe are there would be stronger if we had empirical proof that journalists actually work in this manner.

A natural tool to start investigating how journalists use themes to write about news stories could be Social Network Analysis. Further investigation would be needed, but a starting point would be to ask how a graph of a news story would look like if journalists do actually work in the manner that we have assumed in our study. What are other ways a journalist might work that could also produce a similar graph?

There are many questions like these that automated methods, like the ones that we have implemented for this research, can help investigate empirically and that we want to look at in more detail in the future.

Chapter 7

Case 2: Who are talking to whom about what?

We show how we can use NER, topic modelling, and SNA to describe who appear in a news story, what topics they are discussing, which of the persons in the news story are talking together about the same topic, and who are the information carriers between the topics. We do that by creating a network where the nodes are the names we find through NER and the edges are the topics as discovered through topics modelling.

In this case study we use the same corpus as the case study in chapter 6, but instead of trying to find the thematic structure that is present in the news story, we focus on who the main actors are in each of the topics in the news story.

In our research, we say that a topic is different from a theme in that a topic is concerned with a specific subject within a news story, while a theme represents the overarching issues that are present in society. In the LoVeSe story, a topic could be "the environmental impact of oil drilling in Northern Norway", which follows the theme of environmental concerns.

We look at how different people are the most important characters of each of the sub-topics in the LoVeSe story, and then we look at which persons are the most important information carriers between topics

using Social Network Analysis. We use Latent Dirichlet Allocation to automatically detect the topics in the corpus and Named-Entity Recognition, as described in section 5.3, to find the persons who appear in the corpus. We determine which documents in the corpus belongs to which topic and use that to define the network.

7.1 Method

We calculate the number of topics across the corpus according to how we describe it in section 2.6.1: We use 4 different metrics that tells us something about how well a model of n number of topics represents the corpus that we are investigating. We calculate the occurrence of a topic within a document in the corpus by saying that a document contains a topic if the topic contributes at least 1 paragraph to the document. We choose 75 words as a reasonable length of a paragraph. To reduce the dimentionality of the documents, we lemmatize the words, convert to lower case, remove numbers and punctuation, and filter out stop words from the text. LDA is then run on the term-frequency matrix of the documents.

We employ Named-Entity Recognition as described in section 5.3 to find the persons in each document in the corpus. The NER classifier expects the input text to be pre-tokenized and PoS tagged. we use the PoS tagger provided by Straka and Straková (2017). We use this tagger instead of the tagger we presented in section 4.3 as our tagger requires more setup and was not ready for use on a new corpus outside of the pre-configured development, training, and test corpus we were conducting the research on.

This leaves us with the person–document and document–topic matrices. From there we define that a person discussed a topic with another person if they appear in the same document containing that topic. This gives us a co-occurrence matrix describing a person–person relationship for each topic in the corpus.

The co-occurrence matrices gives us a network between the persons that are discussing each of the topics, and we use this network to discuss who are the most important persons in each of the topics. We also look at how the importance of these persons change over the topics; and given each topic, who are the information carriers between the topics.

We have in this case a specific view of what we mean by a discussion. We are aware that—for most of the time— a news story is written by a journalist and not the person making a statement. What we build the method on is that we assume that when a news story mentions two persons they have addressed the same topic(s) and that at least one of the persons have been made aware of the other persons response before making a comment. This means that the news story is a representation of a public discussion between individuals.

7.2 Results

To choose the number of topics for the corpus, we decide that an upper bound of 10 topics should be a fair number for the small corpus that we are working with. For each $n > 1$ we produce a model and calculate the 4 metrics. The results are shown in figure 7.1. Even though 3 of the metrics show that a model of 10 topics is the best fitting model, we believe that we should try to look at the point where all of the models agree the most, and Deveaud2014 is basically saying that a 10 topic model is close to useless. If we look at the graph we can see that the metrics agree the most on 6 topics. After looking at some of the documents that score high for each topic, and from the knowledge we have of the corpus, we believe having 10 topics for the corpus is basically overfitting to stylistic patterns, and that 6 topics is a better generalization and closer to the semantic topics of the corpus.

In figure 7.2, the top 10 words for each of the topics in the corpus as found by LDA analysis, are shown. After investigating a selection of the corpus, we find that the 6 different topics can be described with

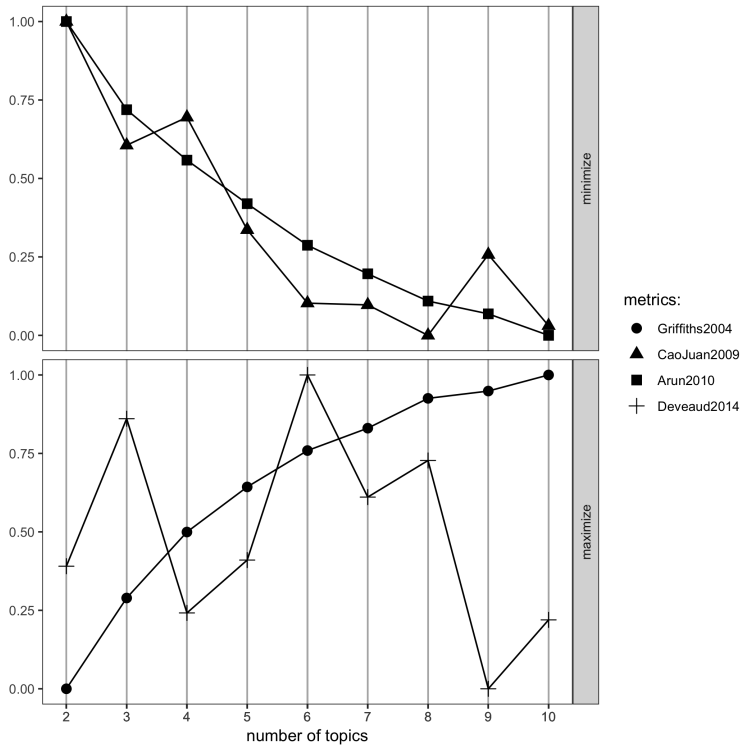

Figure 7.1: Metrics for choosing the number of topics.

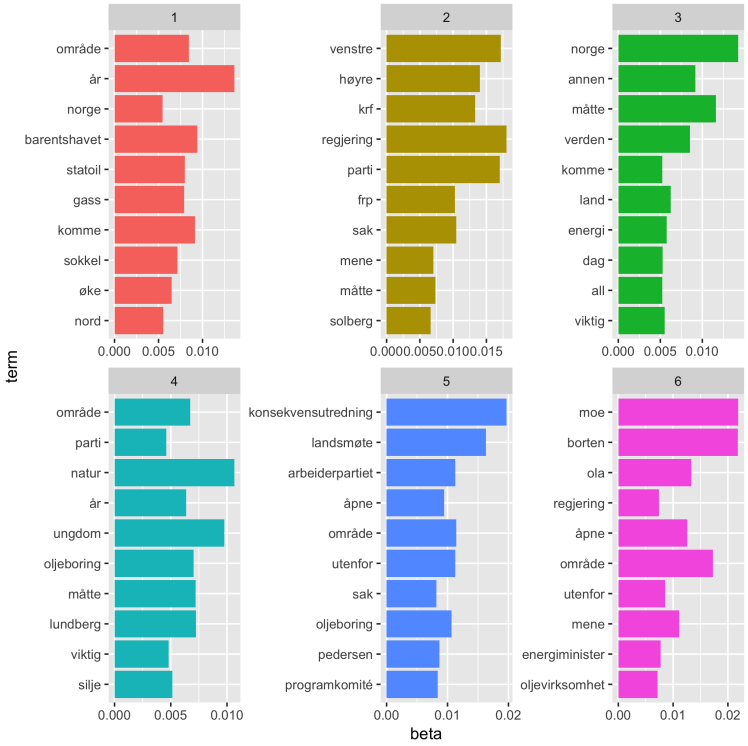

Figure 7.2: Top 10 words for each topic found by LDA analysis.

the following labels

1. Energy security
2. Election results
3. The new Minister of Energy and Oil
4. Environment
5. Feature story on oil in Northern Norway
6. The current Minister of Energy and Oil

We will further describe each of the labels and topics in section 7.3.

We also created a person co-occurrence matrix for each of the topics and calculate the eigenvector centrality scores for each of the topics. The result for the top 9 highest scoring person for each topic are displayed in table 7.1 and 7.2. The eigenvector centrality is a measure of the influence of a node in a graph and we believe that it represents the most important persons in our graph.

We also calculate the eigenvector centrality, or importance, of the full graph—resulting in the scores in table 7.3.

For each topic pair we show who are the top 3 persons with the highest betweenness score; or that they serve as an intermediate point between disparate regions in the combined graphs. The scores are shown in table 7.4.

In section 7.3 we will discuss how we interpret the results: Who the most important persons are, which topics we found, and who the information carrier between the topics are.

7.3 Discussion

It is important to note that we are not measuring the size or relevance between the different topics or groups in the news story as we did in case study 1, described in chapter 6. We are looking at what are

Topic 1			
Nina Jensen	1.00	Siv Jensen	0.64
Rasmus Hansson	0.43	Erik Solheim	0.36
Betzy Kjelsberg	0.36	Erna Solberg	0.34
Jens Stoltenberg	0.34	Ola Borten Moe	0.34
Tore Jensen	0.33		
Topic 2			
Erna Solberg	1.00	Trine Skei Grande	0.80
Knut Arild Hareide	0.72	Siv Jensen	0.69
Jens Stoltenberg	0.33	Ola Borten Moe	0.33
Per Sandberg	0.29	Ketil Solvik-Olsen	0.24
Jan Tore Sanner	0.24		
Topic 3			
Tord Lien	1.00	Lise Rist	0.32
Ola Borten Moe	0.30	Åslaug Haga	0.28
Odd Roger Enoksen	0.28	Helge Lund	0.28
Stanley Wirak	0.26	Thorhild Widvey	0.26
Terje Riis-Johansen	0.26		

Table 7.1: 10 most important persons according to the eigenvector centrality of topic 1, 2, and 3.

Topic 4			
Silje Lundberg	1.00	Jens Stoltenberg	0.80
Erna Solberg	0.66	Siv Jensen	0.50
Trine Skei Grande	0.40	Audun Lysbakken	0.33
Eirin Sund	0.32	Knut Arild Hareide	0.31
Sigurd Enge	0.31		

Topic 5			
Erik Karlstrøm	1.00	Tor Kjetil Wisløff	0.88
Harald Karlstrøm	0.77	Martin Vahl	0.55
Marius Karlsen	0.45	Odd Bakken	0.45
Hans-Ulrik Wisløff	0.45	Tor Kjetil	0.45
Martin jr	0.45		

Topic 6			
Ola Borten Moe	1.00	Borten Moe	0.73
Jens Stoltenberg	0.46	Ola Borten	0.37
Bård Vegar Solhjell	0.36	Frederic Hauge	0.27
Silje Lundberg	0.25	Knut Arild Hareide	0.25
Lars Haltbrekken	0.15		

Table 7.2: Eigenvector centrality of topic 4, 5, and 6.

Full news story			
Ola Borten Moe	1.00	Borten Moe	0.84
Jens Stoltenberg	0.61	Erna Solberg	0.36
Liv Signe Navarsete	0.36	Bård Vegar Solhjell	0.33
Knut Arild Hareide	0.31	Nina Jensen	0.29
Siv Jensen	0.28		

Table 7.3: Eigenvector centrality of full graph

Topics			
1 + 2	Ola Borten Moe	Erna Solberg	Jens Stoltenberg
1 + 3	Jens Stoltenberg	Ola Borten Moe	Erna Solberg
1 + 4	Ola Borten Moe	Jens Stoltenberg	Ole Paus
1 + 5	Jens Stoltenberg	Ola Borten Moe	Eskil Pedersen
1 + 6	Ola Borten Moe	Jens Stoltenberg	Bård Vegar Solhjell
2 + 3	Erna Solberg	Jens Stoltenberg	Ola Borten Moe
2 + 4	Erna Solberg	Ola Borten Moe	Trine Skei Grande
2 + 5	Jens Stoltenberg	Ola Borten Moe	Erna Solberg
2 + 6	Jens Stoltenberg	Ola Borten Moe	Erna Solberg
3 + 4	Jens Stoltenberg	Ola Borten Moe	Erna Solberg
3 + 5	Jens Stoltenberg	Ola Borten Moe	Eskil Pedersen
3 + 6	Ola Borten Moe	Jens Stoltenberg	Bård Vegard Solhjell
4 + 5	Jens Stoltenberg	Eskil Pedersen	Ola Borten Moe
4 + 6	Ola Borten Moe	Jens Stoltenberg	Borten Moe
5 + 6	Jens Stoltenberg	Ola Borten Moe	Bård Vegard Solhjell

Table 7.4: Top 3 persons with the highest betweeness score for each topic pair.

the sub-topics within the LoVeSe news story, and who are the most important persons for each of those topics. We also want to see who are the information carries between the stories.

In section 7.2 we determined that the news story consists of 6 topics:

1. **Energy security** The main topic of the news story. The topic describes the reasons for why the government should open for drilling for oil in the LoVeSe region like energy security and the potential financial benefits for Norway.

2. **Election results** The political right was the election winner in 2013 and this topic is about the different parties (H, V, Krf, and Frp) that were discussing a possible coalition government at the time.

3. **The new Minister of Energy and Oil** After the political right had agreed on a coalition government, they announced their new Minister of Energy and Oil and their view on oil drilling in LoVeSe.

4. **Environment** The environmental impact of drilling in LoVeSe. Most of the environmental groups in Norway are actively working against the consequence study of drilling for oil in the LoVeSe region.

5. **Feature story on oil in Northern Norway** This topic mostly represents one feature story about the oil industry and how it affects the lives of the people in the area.

6. **The current Minister of Energy and Oil** Ola Borten Moe, as the minister of energy and oil and first deputy leader of the Centre party (Sp), went against his own party's (Sp) politics and announced that he would work towards a study of the consequences of drilling for oil in LoVeSe.

To understand who the most important person in the news story is, we extracted the person–person graph and calculated the eigenvector centralities, and we display the 9 highest scoring persons in table 7.3. There it can be seen that "Ola Borten Moe" and "Borten Moe" are the

two highest scoring persons for the graph. Both these entities refer to the same person, making Ola Borten Moe, decidedly the most important person in the news story. This is not surprising as he was the minister of energy and oil at the time and LoVeSe was an important issue for him. The next person is "Jens Stoltenberg". He was the prime minster at the time for the Labour Party (Ap). Erna Solberg was the party leader for the largest party in opposition, the Conservative party (H). Liv Signe Navarsete was the party leader for the Centre party (Sp), and had a public disagreement with Ola Borten Moe. Bård Vegard Solhjell was the minister of environment. Knut Arild Hareide was the leader of the Christian Democratic party (Krf) and were in discussion with the parties on the right about joining in a coalition government. Nina Jensen was the general secretary of WWF-Norway and the sister of Siv Jensen, the leader of the Progress party (Frp). Nina Jensen and Siv Jensen were in strong disagreement about the issue.

In the first topic and in table 7.1, energy security, we can see that the most important person is Nina Jensen. We also see that Siv Jensen, her sister, is the second most important person. Betzy Kjelsberg and Tore Jensen are also mentioned in the top 9. Betzy Kjelsberg is a famous women's rights activist at the turn of the 19th century and is Siv and Nina Jensen's grandmother. Tore Jensen is their father. Though we say that the topic is about energy security, it is also fair to say that parts of the topic is about the relationship between the two sisters and their different opinions on the debate. The reason we still believe that the topic is about energy security is that it also includes the minister of environment, Erik Solheim; the prime minister, Jens Stoltenberg; the minister of oil and energy, Ola Borten Moe; and the leader of the largest opposition party, Erna Solberg. It does however look like the relationship between the two sisters was a focus of the newspapers at the time.

The second topic and table 7.1, election results, show the winners of the election in 2013 and the people representing the parties trying to make a coalition for a new government. Erna Solberg was the leader of

the Conservative party (H), Trine Skei Grande was the leader of the Liberal party (V), Knut Arild Hareide was the leader of the Christian Democratic Party (Krf), Siv Jensen was the leader of the Progress party (Frp), Per Sandberg was the first deputy leader of Frp, Ketil Solvik-Olsen was the second deputy leader of Frp, Jan Tore Sanner was the first deputy leader of H. Jens Stoltenberg was the prime minster of the old government and leader of the Labour party (Ap).

In the third topic and in table 7.1, new minster of energy and oil, we see the most important person is the new minister of oil and energy, Tord Lien (Frp). Lise Rist, communication advisor to Tord Lien. Terje Riis-Johansen was the minister of oil and energy before Ola Borten Moe. Åslaug Haga had the position before Riis-Johansen and Odd Roger Enoksen had it before Åslaug Haga. Thorhild Widvey had it before them. When we investigate the corpus we see that all of these people are used as points of comparison with the new minister.

The forth topic, described in table 7.2, environment, shows Silje Lundberg, the leader of Young Friends of the Earth Norway and Sigurd Enge, advisor for Bellona—both environmentalists. We also see the three leaders of V, Krf, and SV—Trine Skei Grande, Knut Arild Hareide, and Audun Lysbakken. All speaking against the consequence study of LoVeSe.

In topic 5, described in table 7.2, all of the people in the case study are from a single news article. The article is a feature story on the impact of the oil industry on the people living in northern Norway.

The last topic, topic 6, described in table 7.2, unsurprisingly shows Ola Borten Moe as the most important person in the topic about the minister of oil and energy. It also shows "Borten Moe". Journalists will sometimes use only the last name of the person they are describing. We are not able to link different names of the same entity between articles with the method we are using. As the topic is about the minister in relation to the LoVeSe issue it shows Silje Lundberg, Fredric Hauge, and Lars Haltbrekken—all environmentalists. It also shows the minister of environment, Bård Vegar Solhjell. The prime minster, Jens Stoltenberg,

also shows up in the topic.

The heavy involvement of the prime minister in many of these topics—and in the full story—could be an indication of the importance of the LoVeSe issue for the election.

If we take into consideration the top 3 persons with the highest betweeness score in table 7.4, we can also say that not only does the prime minster Jens Stoltenberg show up in many of the topics, he is also an important person between the topics. Betweeness measures how many of the shortest paths between the other nodes of the graph have to go through that node. Therefore, if a person in our graph has a higher betweenness score than another person, they serve as a more important information bridge between the two topics.

The environmental issue also seems to be a big part of most of the topics as the persons representing environmental concerns, like Silje Lundberg, Nina Jensen, and Fredric Hauge, have a prominent and strong score of importance in topic 1, 4, and 6. Nina Jensen is also strongly represented in the full news story as described in 7.3. This is on the surface contrary to what we found in the first case study described in chapter 6. However, in the first case study we measured the size of environmental group and how it connected to the other groups within the full story. In this study, we find the sub-topics of the full story and measure the importance of individuals within those topics and the full story. It could still be true that the environmental group is used less by the newspapers, but their impact based on who they interact with could be much larger than what we could find using the method in the first case study.

7.4 Conclusion

In this case study we have shown that by using automated text analysis methods like LDA to find the sub-topics within a news story we can describe who the most important persons are within those topics by

extracting the person with NER and using SNA to describe the importance of each person. In the end we have a separate graph for each topic describing which persons appear together in the same newspaper articles. We can then also look at which persons are important for bringing information between topics by looking at the betweeness of the persons when we combine the graphs for each topic into graphs of pairs of topics.

7.5 Future work

One of the first things we should do is to look at how we can ensure that the data is cleaned properly. One obstacle we had during the development of the research for this case study was that the article author would become an important person as they were mentioned together with a very wide set of other persons in the graph. This is problematic as we are interested in the relationship between the opinion holders the article describes. The article author adds noise to that relationship.

Another challenge—that is especially evident in table 7.3—is that we cannot align different names of the same entity. It is difficult to automatically say that the names "Ola Borten Moe" and "Borten Moe" belong to the same person.

It is also problematic when multiple entities have the same name. We assume all references to a specific name is the same person, but this is not always true. The name "Solberg" can in one context be a reference to Erna Solberg, the leader of the Conservative party, and in another it could be the Norwegian rally driver Petter Solberg. This is a problem known as Named Entity Linking (NEL) and it is a problem we should investigate and that could improve the accuracy and reliability of the result of case studies similar to this one.

We would also like to investigate how journalists use sources in a news story. Do they always work in the same manner that we have

assumed based on the description by Allern (2015), or does it change based on the story? Is there a difference between how a magazine versus how a newspaper uses its sources? NEC, NER, and NEL together with SNA opens up new avenues and allows us to empirically test such questions through statistical analysis of the relationships between the entities in the text.

Chapter 8

Related work

In this chapter we describe research related to what we have done in the two case studies in chapter 6 and 7.

In our first study we introduce a news story and a corpus that we process using Named-Entity Chunking to find all the named entities in the text. We create a graph where the names are nodes and the text are the edges. We then use community detection to discover the groups of the story. We use the groups that we discover as a proxy for the major themes that are discussed in the story.

In our second study, we use Latent Dirichlet Allocation to find the topics in the same story as the previous case study. For each topic we collect the texts that contains that topic. We then use Named-Entity Recognition to extract the persons as the nodes of the network for each topic. We create an edge for the topic if two persons appear in the same text together. We then use the network for each topic to analyze who the most important persons are in each network. We also join the networks to see who the most important information carriers are between the topics.

Other researchers have also used text analysis tools and/or Social Network Analysis to study the interaction between entities in news stories or other types of text.

Grimmer and Stewart (2013) argue that we need to be rigorous when

we employ automated text analysis tools and methods when researching political texts, but that these methods can help to reduce the cost of analyzing large corpora.

Lucas et al. (2015) gives a broad overview of how computer-assisted text analysis can be used for comparative politics.

Van Atteveldt et al. (2008) used many different techniques such as dependency parsing; detecting the source of quote or phrase in an article; finding the subject and objects; recognizing the names of known politicians; and anaphora resolution, to find the semantic networks of a text. Even though the performance of most of their methods were quite low, at less than 65% F_1 score, they were able to answer many different hypothesis about how politicians are quoted and used as sources in newspaper articles.

Vinciarelli and Favre (2007) looked at using SNA on the transcripts from radio shows. They try to use the network to segment out stories from within a wider set of texts. Their results show that they are able to get to a reasonable performance where their segmenter can be used as a tool to get close to the true segment boundaries.

Favre (2009) has also looked at using SNA on transcripts from broadcast media. They look at trying to discover the roles of the speakers in the text: If they are the news anchor, weatherman, guest, etc.

Newman and Girvan (2004) combined topic modelling with NER into a new approach based on LDA that can learn the relationship between the topics and the entities mentioned in a text. They show that the model can be used to predict a relationship between entities in a corpus that do not directly appear together.

Davulcu et al. (2010) extracted entities, social markers, and the sentiment towards those markers from a corpus consisting of 77000 newspaper articles from Indonesia. The goal was to enhance the "understanding of counter-radical movements in critical locations in the Muslim world." They were able to use the extracted information to cluster the organizations mentioned in the text into 8 different groups

based on co-occurrence of the organizations with social markers and sentiment. With the help of social scientists, 4 of the groups were identified as either purely radical or counter-radical.

Diesner et al. (2012) used NER, Named-Entity Linking and co-occurrence to create a socio-cultural network of Sudan over time. The corpus they used consists of about 32000 text documents published by the Sudan Tribune. From analysing the network they report that though the network changes from year to year, the most important entities and their ranking remain fairly robust.

Neumann and Sartor (2016) examined a corpus consisting of police interrogations in regards to a number of interrelated police investigations on money laundering. They extracted the agents, organizations, exchanges of resources, tasks, groups, and events automatically from the data and made a co-occurrence network. They were able to uncover a complex structure of companies involved in highly professional financial transactions designed to launder money.

Sudhahar et al. (2015) analyze the network of subject-verb-object (SVO) triplets in a news corpus covering the 2012 US presidential election. They extract the SVO triplets by automatically parsing the text. The verbs in the SVO triplet are used to assign the sentiment that the subject holds towards the object in the sentence. They found that they could reliably recover the "spectrum of political positions" by analysing the claims attributed to each actor. The results show that the 2012 campaign was focused on the economy and civil rights and that Obama challenged the traditional Republican ownership of the economy.

Sjøvaag et al. (2018) analyzed the relationship between different news organizations in Scandinavia by looking at the hyperlinks between their digital outlets. They find that though the Scandinavian news network is connected, most news sites link to other news sites within their respective national borders and corporate affiliation is important in carrying information across borders. They also show that local and independent news agencies assume weaker positions in the network, and

that it is the large national and corporately owned newspapers that drive the interconnection in the network.

Sjøvaag and Pedersen (2018) investigated how certain structural features impact the presence of women in the news. They manually analyzed parts of their corpus, and then used an automatic approach on the rest. For the automatic approach they used NEC and a name list to find female and male names. They show that it is the distribution level (local, metropolitan, national) that mostly affect the presence of women in news, and not other structural features such as the circulation size, funding model, or the corporation they belong to.

Touileb and Duarte (2016) used induced information structures as key phrases for news content analysis. They showed that these structures can be used to characterize a large corpus and give an overview of the content.

Touileb et al. (2018) used NEC and Structure Induction for Mining Meaningful Snippets to find marginal politicians in large corpora. Their approach takes a seed list of known politicians and produces patterns that can be used to automatically identify and extract the names of politicians that do not appear in the list. The reason they do this is because it is often easy to find a list of national politicians and leaders, but not local and international politicians. They show that their method is a good first step and that they are able to find the names of politicians that the method was not aware of from before.

Kaplun et al. (2018) explore how sentiment and controversy are related in online news articles. They show that they cannot find a correlation between negative sentiment and controversy.

Pontiki et al. (2018) used a data-driven linguistic approach to study the targets of xenophobia-motivated behavior in Greece. They collected a corpus of over 3 million Greek news articles and 4 million twitter messages. They then extract the named entities and their syntactic relationships. Their analysis indicates that xenophobic behaviors are not dominant in Greece, and that the increase in violence towards foreigners follows the increase in violence against Greeks in general.

Tannier (2016) build a tool to identify the evolution of alliance and opposition between countries on specific topics. They use time series plots and dynamic networks to visualize the relationship between countries. They show that this kind of tool can produce compelling data-journalistic content.

DiMaggio et al. (2013) used LDA to investigate how newspapers frame government assistance to artists and arts organizations. They found that the tone of press coverage of arts funding shifted dramatically in 1989 from celebratory to controversy focused; and that it persisted throughout the 1990s. Though they say that "topic modelling will not be a panacea for sociologists of culture," it is still a powerful tool for understanding and exploring large corpora.

Elgesem et al. (2016) looked at bloggers' response to Edward Snowden's revelations of the secret surveillance program PRISM. They used LDA to find the topics that the bloggers were discussing and created a co-occurrence network over topics and analyzed which topics were discussed the most together. They then looked specifically at the term "PRISM" and the topics it appears in according to the LDA analysis and used Spherical k-Means clustering to discuss how the bloggers dealt with the trustworthiness of the reports about PRISM. They conclude that the bloggers were important contributors to civic engagement.

Elgesem (2017) combined Topic Modelling with Social Network Analysis to investigate the polarization of the discussion on blogs about the Paris climate meeting in December 2015. They find blog posts that discuss topics that they identify as related to the Paris climate meeting. They then look at the co-citation network between the blogs and analyze their interactions. Although they cannot claim that the bloggers became more polarized toward the extreme over the course of the climate meeting, they find key markers of polarization between the people that accept anthropogenic climate change versus the people who claim to be sceptical.

Chapter 9

Discussion

In this chapter, we discuss how the experiments conducted for this thesis, and their results, relate to each other. We will show that each contribution is a step towards better tools for the analysis of Norwegian text.

The focus of this thesis, as we presented in the introduction, has been the development and use of automatic analysis methods for Norwegian text. As automated analysis of text is a large research field, we decided to concentrate our research on three different tasks: Part-of-speech tagging, Named-Entity Chunking and Named-Entity Recognition. To show the efficacy of the tools that we have developed to solve these tasks, we formulate, and then investigate, two different case studies.

The two case studies both investigate the same news story: The debate around the study of the consequences of drilling for oil in Lofoten, Vesterålen, and Senja during the Norwegian election in 2013. The first case study use NEC and Social Network Analysis to find the thematic structure of the news story. The second study investigates who are talking to whom and what they are talking about through NER and SNA.

9.1 Part-of-speech

The first part of our thesis focuses on the task of PoS tagging. We first experiment with training the OBT+Stat which, at the time of the research, was only able to statistically disambiguate and tag Bokmål. Since the constraint grammar part of OBT+Stat, the Oslo-Bergen Tagger, already supported Nynorsk, we used the Norwegian Dependency Treebank to train a Hidden Markov Model. We achieved an accuracy of 94.43%.

For our second experiment on PoS, we used Google's open source neural network framework, SyntaxNet, to train it as a PoS tagger for Bokmål and Nynorsk and achieved the same results as the state-of-the-art for both Norwegian language forms. We achieved an accuracy of 97.54% for Bokmål and 96.83% for Nynorsk.

When comparing the results from our first approach to that of the second approach, it is evident that the OBT+Stat model performs worse than the SyntaxNet solution. We believe that the main reason SyntaxNet performs better than the HMM approach is that SyntaxNet has more opportunity to decide which parts of the context it uses in the decision process.

SyntaxNet is a feed-forward neural network using a context window around the current word and the tags for the previous tokens as the features used for classification. OBT+Stat uses a HMM to decide the tag by only observing the current word and hidden state. It then looks to see if it can find the same tag in the potentially ambiguous output of the OBT. If they agree on a tag, that tag becomes the output from OBT+Stat.

It is the strength of the underlying rule-based tokenizer of OBT+Stat that allows the high performance of the HMM model. Even though we only use each token for each observation, the HMM is able to get almost comparable results to the state-of-the-art SyntaxNet solution. We believe that this is because the HMM only has to choose from a small subset of tags and not from all tags for every token.

9.2 Named entities

The second part of our research focused on Named-Entity Chunking and Recognition. As stated before, for most contemporary research, NEC is a task that is implicitly done within the process of NER. The reason we wanted to build an explicit NEC model was that other previous attempts at NER for Norwegian only focused on telling what type of entity an already pre-chunked name is. An explicit NEC could therefore make it possible to use those previous attempts in future research. Another reason is that an explicit NEC can help us understand the performance of a NER model and point us in a direction for how we should work to improve NER in the future.

In our research on NEC we reported a result of 96.63% accuracy for Bokmål, while for NER we achieved an accuracy of 87.20%—a difference of almost 10 percentage points. This raises the question: What is it that drives such a large difference between finding a name and also finding its category?

A NER tagger, like the one we developed in section 5.3, is in simple terms 4 different chunkers that are combined into one single model. One chunker for each of the different name categories that we defined: Locations, organizations, miscellaneous, and persons. However, looking at the highest scoring category, we get an accuracy of 92.74% for person names. This is still much lower than what we are able to show when we chunk all names as one category.

The reason this happens could be that each of the categories of names can affect the performance of the other categories. The model has to decide on a single category or label for each of tokens in the sequences that it is labeling. Which means that if a location is mistakenly categorized as a person, we will both get a lower precision for the person category and a lower recall for the location category. Potentially, a mistake can affect the accuracy of several parts of the model.

Another challenge could be that, for Norwegian, it is easy to say if part of the text is a name, but it is difficult without the proper context

to say which category of name it is. It could also be that we are not fine grained enough, and our attempt to keep the same categories as we see in research for the English language was not the right approach. Our reasoning was that we wanted to more easily be able to compare our research to other international researchers' work. We are also doing better than the previous work that has been done for Norwegian, so it is difficult to say if the number of categories is actually a problem. We would have to conduct an experiment where we explicitly label finer and coarser and see for what level we get the best results.

Our research also supports the findings of Husevåg (2016) that showed that Norwegian has a much lower density of named entities when compared to English. Deep learning models require large amounts of data to generalize over a data set. This could mean that to train an as well-performing model for Norwegian as has been achieved for English, NER models for Norwegian require larger data sets than for English.

Unfortunately, we cannot tell whether the case studies suffer from these types of problems or not. Working with raw, unlabeled text means that it is only possible to verify what is working and what the recognizer or chunker shows. We would have to go through and manually process every text in the corpus to verify what the model does not find. However, looking at the output of our chunker, it is apparent that it does suffer from some pathological issues.

It has problems with abbreviations like "f.eks" (translation: f.ex) and in some instances short forms of political parties like "Frp" where it will attach the surrounding tokens as part of the abbreviation instead of treating them as separate tokens. This will, of course, lead to misclassification. The first case study, described in chapter 6, suffers from this mistake. In that study we have to filter out and align some of the names that the chunker finds as otherwise there is too much noise in the resulting network.

We also see a tendency to add titles as part of names. That might be because there is a tendency to see a noun that is next to other nouns as

a name—especially in the beginning of a sentence. An example would be the following sentence:

1	2	3	4	5	6	7
noun	noun	noun	verb	noun	prep	noun
Frp-leder	Siv	Jensen	stiller	krav	til	forhandlingene
Frp-leader	Siv	Jensen	poses	demands	to	the negotiations

There are several features that gives the model an opportunity to make a mistake in a sentence like this: The first token is a title and it is capitalized; it contains a named entity, but the token itself is not a named entity; and it is next to a proper named entity "Siv Jensen".

Though the token sequence "Frp-leder Siv Jensen" is definitely in the training data and it is tagged in the correct manner, the problem becomes that for the SVM to delineate correctly between the text and the names there are too few instances of this type to tag to categorize it correctly (in all instances).

An idea to fix this problem would be to add a gazetteer for titles as part of the features for the classifier. Another idea would be to introduce embeddings like we did for the NER model. An embeddings model could be able to detect the similarity between the different leader titles ("Frp-leder", "SV-leder", "Krf-leder", ...) and make it easier for the model to categorize these types of sentences correctly.

For the second case study, in chapter 7, where we use the NER model, we do not see this type of misclassification where the title becomes part of the name. Here, the NER model, uses a sub-word embedding as the first layer.

The NER model also does not have the same tendency as the chunker to have problems with abbreviations. We believe that the reason the NER model is doing better at this is because the underlying tokenizer is better at handling these types of tokens. The chunker uses the OBT

while the recognizer uses UDPipe. It is possible that we could eliminate—or at least lessen—this type of error if we changed the tokenizer that we used with the chunker.

However, these problems does not prevent us from using either of the two taggers in the case studies. Especially since we are looking for the most important and frequent entities. We can therefore remove most tokens that are misclassified as entities by removing the least frequent entities that we find as they should not be measured as important in our study. Most, if not all, misclassified tokens appear infrequently.

We also discovered that in the majority of the articles, the article will mention the author as part of the general text. This becomes a challenge in our case studies because we were interesting in researching the relationship between the subjects of the news story. The author is incidental to that relationship. Since we are looking at the network between the entities in the texts, the author would create an artificial bridge between the entities and add noise to the network. We would also see the same problem for the photographer and news agencies. This is not the fault of the chunker or recognizer, but a problem with the cleanliness of the data.

Despite these obstacles, the two case studies show how such tools can be used to research a news story empirically and from different angles.

9.3 The case studies

Since the two case studies are on the same news story there are some points that should be compared. In the first study we come to the conclusion that the least important group is the environmental group. In the second case study we found that environmentalists have a large importance in 3 of the 6 topics we found. Despite environmentalists showing up as important in half of the topics, we see that the only environmentalist that is seen in the top 10 most important persons in the

second study is Nina Jensen. This tells us that though environmentalists are consulted in regards to most topics, they are not as important as politicians. This is not surprising as the news story is about a political issue that was discussed broadly between the different political parties at the time.

There are several reasons for why Nina Jensen does not appear in the network in the first case study. It could be that in many cases the chunker was not able to find the full name and either found Nina or Jensen, but not both. As case study 2 showed: Nina Jensen mostly appeared together with Siv Jensen, her sister. It could be that we mistakenly attributed the entity "Jensen" as "Siv Jensen". It is also possible that when we look at all entities, and not just persons, Nina Jensen appears in the text too few times to make it past the initial filtering. This would be regrettable, as she is an important character in the discussion. We should note that we also see "Ole Paus" in the second study, a Norwegian singer–songwriter, and he does not appear in the first study. This could be evidence for the argument that when we look only at the persons in the story, the importance of the named entities change dramatically. In other words, when we consider all names, some persons that we want to see drown behind names like "Lofoten". "Lofoten" is a central name to the LoVeSe news story, but it does not necessarily give us much information about the relation between the entities in the story. We see evidence of this after we perform violator removal in the first case study and the 3 first entities that we remove are "Lofoten", "Vesterålen", and "Senja".

The two case studies are also working on different levels of a news story. In the first study we look at the strongest interaction between the entities that appear in the text to find the thematic structure of the text. In the end, we can only say something about the groups that we believe represent the thematic structures, and not anything about the interactions in the topics in the news story.

In the second study we find the topics and then look at the interaction of only the persons that are mentioned within each topic. We use LDA

to find the topics in the news story. LDA represents a documents as a mixture over latent topics. Since we also do not filter the resulting graph as heavily as in the first case study, we should to see other persons than only those that represent the theme of a topic within the graph of that same topic.

Chapter 10

Conclusion

Through the research in this thesis it is quite clear that the field of automated analysis of Norwegian text is underdeveloped. We have through basic research shown that we can exceed, with a large margin, what other research have done before on NER by utilizing the state of the art from English research and adapting it to a Norwegian context. Given more time, it is not unthinkable that we should be able to achieve comparable results to what has been shown to be possible for English text. We would have to overcome some problems that are not present in English text, but are in Norwegian. Problems like the lower name density of Norwegian, the high degree of polysemy, the many capitalization rules, etc. This is not an insurmountable task and through additional funding it should be possible to solve.

We are also able to get similar results on PoS tagging as other researchers through training off-the-shelf software with Norwegian data. This shows that we should expend the effort to research what the global community has developed and take advantage of that research. It is unreasonable to assume that Norwegian is not suitable for these algorithms just because the language is different from English.

The biggest hurdle to get performance at the same level as what is possible for English text is, however, the amount of data and data quality. We need to get more tagged text from more sources and we

need to ensure the quality of the data we have.

A unique aspect of Norwegian is that it has two official written forms. We found that by training a model on a combined corpus of the two written forms for Named-Entity Recognition, we could get a better performing model. We believe that this is something we should investigate further together with researchers on Swedish and Danish. Norwegian, Swedish, and Danish are relatively similar language.

The reason for developing this type of software is not only to keep Norwegian relevant, but also to make it possible to research bigger volumes of data in a shorter amount of time. Right now, the time to complete a quantitative study of a large news story over multiple media (newspapers, blogs, messages, etc.), is on the order of years. Because of this, it is possible that some researchers that are interested in this type of challenges will turn to research on English issues instead.

We argue that through the two case studies, we show that by having access to well-performing NEC and NER models we can investigate the relationship between entities in the news through the use of Social Network Analysis. We therefore believe that the two case studies validates the usefulness of the tools that we have developed for this thesis.

This type of software is also not only limited to the type of analysis that we have done for our research, but has also been used to investigating the gender balance in the media in Norway (Sjøvaag and Pedersen, 2018). We hope that our research can be a useful addition to this type of research going forward.

10.1 Future work

The most pressing concern is to develop stronger and larger training sets for Norwegian. The field should work towards ensuring that there is high quality corpora available for both written forms: Nynorsk and Bokmål. We have shown evidence for the fact that analyzers for Norwegian text

perform better when trained on both forms, instead of just one. This could be because of large variances in how both forms are written, and training on both forms increases the learning potential of the models. We have seen a performance increase when training a NER model, but does it hold for other tasks as well? Could the reason be that the corpus is bigger, or is the model truly able to generalize better over the larger data set? Can any of the other Scandinavian languages like Swedish and Danish be included into a common model? Bokmål is after all, as we explained in section 3.4, a reformed version of the Danish language's written form.

It would be a huge boon to automated analysis of Norwegian text if researchers and users did not have to worry about the differences between the two written forms. It could also make it easier to get funding as funding agencies would not have to prioritize between the two forms. They would also not have to worry about supporters of one form feeling neglected over the other.

There are many natural language processing tasks that are not developed for Norwegian or where there are few resources:

- Named-Entity Linking (NEL)
- Question answering
- Relationship Extraction (RE)
- Sentiment Analysis
- Summarization
- etc.

The work we have done for this thesis can be used to start researching some of these topics, especially NEL and RE, but for many tasks, the field is still at a basic research level. There is a substantial need for more resources; the field is already lagging behind if one looks at the

large amount of interest from outside of academia in the last couple of years.

NEL is the task of disambiguating between entities with the same names. The name "Solberg", for instance, can both refer to the Norwegian prime minister, Erna Solberg, and a world champion rally driver, Petter Solberg, depending on what the context is. This is a different task compared to NER where one is only interested in saying if the name "Solberg" in is a person, organization, location, or another type of name and not finding the exact entity that is mentioned.

RE is also related to NER since for RE the object is to find the relationship between the entities that are mentioned in the text. If we take the sentence "Solberg is the leader of the Conservative Party", the task would be to extract the structured information regarding who is the leader of which party.

It is surprising that other social scientist have not already jumped on developing this type of research. Tools like the ones that we have develop through our research, have a large impact on what type of quantitative research that can be done. NER, for example, has the potential to enable us to research questions about entities that appear in the news in Norway. Without automated tools, researchers would have to manually find and annotate the documents the researchers are interested in; and that makes it prohibitively expensive to investigate big corpora.

References

Steven Abney. Statistical methods and linguistics. *The balancing act: Combining symbolic and statistical approaches to language*, pages 1–26, 1996.

Sigurd Allern. *Journalistikk og kildekritisk analyse*. Cappelen Damm akademisk, 2015.

Gisle Andersen and Knut Hofland. Building a large corpus based on newspapers from the web. *Exploring Newspaper Language: Using the web to create and investigate a large corpus of modern Norwegian*, 49:1, 2012.

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 2442–2452, 2016.

Rajkumar Arun, Venkatasubramaniyan Suresh, C. E. Veni Madhavan, and M. N. Narasimha Murthy. On finding the natural number of topics with latent dirichlet allocation: Some observations. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 391–402. Springer, 2010.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

Eckhard Bick. Named entity recognition for danish. *I: Årbog for Nordisk Sprogteknologisk Forskningsprogram*, 2004, 2000.

Eckhard Bick. Multi-level ner for portuguese in a cg framework. In *International Workshop on Computational Processing of the Portuguese Language*, pages 118–125. Springer, 2003.

Eckhard Bick. A named entity recognizer for danish. In *Fourth International Conference on Language Resources and Evaluation*, 2004.

Eckhard Bick, Kristin Hagen, and Anders Nøklestad. Optimizing the oslo-bergen tagger. In *Proceedings of the Workshop on "Constraint Grammar-methods, tools and applications" at NODALIDA 2015, May 11-13, 2015, Institute of the Lithuanian Language, Vilnius, Lithuania*, pages 11–17. Linköping University Electronic Press, 2015.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

Bernd Bohnet, Ryan McDonald, Goncalo Simoes, Daniel Andor, Emily Pitler, and Joshua Maynez. Morphosyntactic tagging with a meta-bilstm model over context sensitive token encodings. *arXiv preprint arXiv:1805.08237*, 2018.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

Guri Bordal. Substantiv. `https://snl.no/substantiv`, July 2015.

Endre Brunstad. Kva er god nynorsk språkføring? i helge omdal og rune røsstad (red.): Språknormering–i tide og utide?(s. 91–108). *Novus forlag*, 2009.

Juan Cao, Tian Xia, Jintao Li, Yongdong Zhang, and Sheng Tang. A density-based method for adaptive lda model selection. *Neurocomputing*, 72(7-9):1775–1781, 2009.

Barry R. Chiswick and Paul W. Miller. Linguistic distance: A quantitative measure of the distance between english and other languages. *Journal of Multilingual and Multicultural Development*, 26(1):1–11, 2005.

Jason P. C. Chiu and Eric Nichols. Named entity recognition with bidirectional lstm-cnns. *arXiv preprint arXiv:1511.08308*, 2015.

Noam Chomsky. *Syntactic structures*. Walter de Gruyter, 2002.

Hercules Dalianis and Erik Åström. Swenam—a swedish named entity recognizer. *Technical Report, TRITANA-P0113, IPLab-189, KTH NADA*, 2001.

Hasan Davulcu, Syed Toufeeq Ahmed, Sedat Gokalp, H. Temkit M'hamed, Tom Taylor, Mark Woodward, and Ali Amin. Analyzing sentiment markers describing radical and counter-radical elements in online news. In *IEEE Second International Conference on Social Computing*, pages 335–340. IEEE, 2010.

Koenraad De Smedt, Gunn Inger Lyse, Anje Müller Gjesdal, and Gyri S. Losnegaard. *Norsk i den digitale tidsalderen (bokmålsversjon) – The Norwegian Language in the Digital Age (Bokmål Version)*. META-NET White Paper Series. Georg Rehm and Hans Uszkoreit (Series Editors). Springer, 2012.

Romain Deveaud, Eric Sanjuan, and Patrice Bellot. Accurate and effective latent concept modeling for ad hoc information retrieval. *Document numérique*, 17(1):61–84, 2014.

Jana Diesner, Kathleen M Carley, and Laurent Tambayong. Extracting socio-cultural networks of the sudan from open-source, large-scale text data. *Computational and Mathematical Organization Theory*, 18(3): 328–339, 2012.

Paul DiMaggio, Manish Nag, and David Blei. Exploiting affinities between topic modeling and the sociological perspective on culture: Application to newspaper coverage of u.s. government arts funding. *Poetics*, 41(6):570–606, 2013.

Dag Elgesem. Polarization in blogging about the paris meeting on climate change. In Giovanni Luca Ciampaglia, Afra Mashhadi, and Taha Yasseri, editors, *Social Informatics*, pages 178–200. Springer International Publishing, 2017.

Dag Elgesem, Ingo Feinerer, and Lubos Steskal. Bloggers' responses to the snowden affair: Combining automated and manual methods in the analysis of news blogging. *Computer Supported Cooperative Work (CSCW)*, 25(2):167–191, 2016.

Sarah Favre. Social network analysis in multimedia indexing: Making sense of people in multiparty recordings. In *Proceedings of the Doctoral Consortium of the International Conference on Affective Computing & Intelligent Interaction (ACII)*, 2009.

Tove Fjeldvig and Anne Golden. Automatisk splitting av sammensatte ord–et lingvistisk hjelpemiddel for tekstsøking. In *Proceedings of the Nordic Conference on Computational Linguistics*, pages 73–82, 1985.

Zoubin Ghahramani. An introduction to hidden markov models and bayesian networks. *International journal of pattern recognition and artificial intelligence*, 15(01):9–42, 2001.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

Google. Syntaxnet: Neural models of syntax. `github.com/tensorflow/models/tree/master/syntaxnet`, 2016a. Accessed: 2016-08-23.

Google. Parsey's cousins. `github.com/tensorflow/models/blob/master/syntaxnet/universal.md`, 2016b. Accessed: 2016-08-23.

Thomas L. Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(suppl 1):5228–5235, 2004.

Justin Grimmer and Brandon M. Stewart. Text as data: The promise and pitfalls of automatic content analysis methods for political texts. *Political analysis*, 21(3):267–297, 2013.

Åsne Haaland. *A Maximum Entropy Approach to Proper Name Classification for Norwegian*. PhD thesis, University of Oslo, March 2008.

Kristin Hagen, Janne Bondi Johannessen, and Anders Nøklestad. A constraint-based tagger for norwegian. In *Odense Working Papers in Language and Communication 19*, 17th Scandinavian Conference of Linguistics, pages 31–48, 2000.

Péter Halácsy, András Kornai, and Csaba Oravecz. Hunpos: an open source trigram tagger. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 209–212. Association for Computational Linguistics, 2007.

Timothy Hannigan. Close encounters of the conceptual kind: Disambiguating social structure from text. *Big Data & Society*, 2(2), 2015.

Lars Hellan and Tore Bruland. A cluster of applications around a deep grammar. In *Proceedings from The Language & Technology Conference (LTC)*, 2015.

Lars Hellan, Tore Bruland, Elias Aamot, and Mads H. Sandøy. A grammar sparrer for norwegian. In *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA 2013); May 22-24; 2013; Oslo University; Norway. NEALT Proceedings Series 16*, pages 435–439. Linköping University Electronic Press; Linköpings universitet, 2013.

Ingeborg Rygh Hjorthen and Even Kjølleberg. Prøveboring nord for 62. `http://www.nrk.no/nyheter/1.6620491`, May 2009. Accessed: 2015-12-02.

Anne-Stine Ruud Husevåg. Named entities in indexing: A case study of tv subtitles and metadata records. In *Networked Knowledge Organization Systems Workshop*, pages 48–58, 2016.

Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer, 1998.

Janne Bondi Johannessen and Helge Hauglin. An automatic analysis of norwegian compounds. In *Papers from the 16th Scandinavian Conference of Lingustics*, 1996.

Janne Bondi Johannessen, Kristin Hagen, Anders Nøklestad, and André Lynum. Obt+ stat: Evaluation of a combined cg and statistical tagger. *Constraint Grammar Applications*, pages 26–34, 2011.

Bjarte Johansen. Named-entity chunking for norwegian text using support vector machines. *NIK: Norsk Informatikkonferanse*, 2015.

Bjarte Johansen. Training googles syntaxnet to understand norwegian bokmål and nynorsk. *NIK: Norsk Informatikkonferanse*, 2016.

Andra Björk Jónsdóttir. *ARNER, what kind of name is that? - An automatic Rule-based Named Entity Recognizer for Norwegian.* PhD thesis, University of Oslo, May 2003.

Kateryna Kaplun, Christopher Leberknight, and Anna Feldman. A comparison of lexicons for detecting controversy. In *Proceddings of the LREC 2018 Workshop: Natural Language Processing meets Journalism III*, pages 1–5, 2018.

Jurgita Kapočūtė-Dzikienė, Anders Nøklestad, Janne Bondi Johannessen, and Algis Krupavičius. Exploring features for named entity

recognition in lithuanian text corpus. In *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA 2013); May 22-24; 2013; Oslo University; Norway. NEALT Proceedings Series 16*, pages 73–88. Linköping University Electronic Press, 2013.

Lauri Karttunen. Beyond morphology: Pattern matching with fst. In *International Workshop on Systems and Frameworks for Computational Morphology*, pages 1–13. Springer, 2011.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Dimitrios Kokkinakis. Reducing the effect of name explosion. In *Proceedings of the LREC Workshop: Beyond Named Entity Recognition, Semantic Labelling for NLP tasks*, pages 1–6, 2004.

Dimitrios Kokkinakis, Jyrki Niemi, Sam Hardwick, Krister Lindén, and Lars Borin. Hfst-swener – a new ner resource for swedish. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, 2014.

Taku Kudo and Yuji Matsumoto. Chunking with support vector machines. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, pages 1–8. Association for Computational Linguistics, 2001.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289. Morgan Kaufmann Publishers Inc., 2001.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.

Halvard Leira. Samnorsk som identitetspolitisk prosjekt. *Nytt Norsk Tidsskrift*, 20(04):379–400, 2003.

Svein Lie. Combinatory coordination in norwegian. In *Sixth Scandinavian Conference of Linguistics*, pages 84–89, 1982.

Christopher Lucas, Richard A. Nielsen, Margaret E. Roberts, Brandon M. Stewart, Alex Storer, and Dustin Tingley. Computer-assisted text analysis for comparative politics. *Political Analysis*, 23(2):254–277, 2015.

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*. Cambridge University Press, 2008.

Cristina Sánchez Marco. An open source part-of-speech tagger for norwegian: Building on existing language resources. In *Ninth International Conference on Language Resources and Evaluation*, pages 4111–4117, 2014.

Joel Mickelin. Named entity recognition with support vector machines. Master's thesis, KTH Royal Institute of Technology, 2013.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Tom M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

Johan Myking. Standardization and language planning of terminology: The norwegian experience. *Nazioarteko Terminologia Biltzarra*, pages 227–248, 1997.

Martin Neumann and Nicholas Sartor. A semantic network analysis of laundering drug money. *Journal of Tax Administration*, 2(1):73–94, 2016.

Mark E. J. Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

Trine Nickelsen. Uio-lingvist med oppsiktsvekkjande påstand: – engelsk er eit skandinavisk språk. `https://www.apollon.uio.no/artikler/2012/4-engelsk-er-skandinavisk.html`, Nov 2012. Accessed: 19.02.2013.

Anders Nøklestad. *A machine learning approach to anaphora resolution including named entity recognition, pp attachment disambiguation, and animacy detection*. PhD thesis, University of Oslo, June 2009.

Peter Norvig. On chomsky and the two cultures of statistical learning. http://norvig.com/chomsky.html, 2011. Accessed: 2019-01-04.

Fredrik Olsson. *Bootstrapping named entity annotation by means of active machine learning: a method for creating corpora*. PhD thesis, University of Gothenburg, 2008.

Lilja Øvrelid. Disambiguation of syntactic functions in norwegian: modeling variation in word order interpretations conditioned by animacy and definiteness. In *Proceedings of the 20th Scandinavian Conference of Linguistics*, pages 1–17. Helsinki: University of Helsinki, 2004.

Lilja Øvrelid and Petter Hohle. Universal dependencies for norwegian. *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, 2016.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proc. of NAACL*, 2018.

Maria Pontiki, Konstantina Papanikolaou, and Haris Papageorgiou. Exploring the predominant targets of xenophobia-motivated behavior: A longitudinal study for greece. In *Proceddings of the LREC 2018 Workshop: Natural Language Processing meets Journalism III*, pages 11–15, 2018.

Taraka Rama, Pål Brekke, Øystein Nytrø, and Lilja Øvrelid. Iterative development of family history annotation guidelines using a synthetic corpus of clinical text. In *Proceedings of the Ninth International Workshop on Health Text Mining and Information Analysis*, pages 111–121, 2018.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions, 2018. URL `https://openreview.net/forum?id=SkBYYyZRZ`.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323 (6088):533, 1986.

Helge Sandøy. Language culture in norway: A tradition of questioning standard language norms. *Standard languages and language standards in a changing Europe*, pages 119–126, 2011.

Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. Evaluation methods for unsupervised word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 298–307, 2015.

Helle Sjøvaag and Truls André Pedersen. Female voices in the news: Structural conditions of gender representations in norwegian newspapers. *Journalism & Mass Communication Quarterly*, 2018.

Helle Sjøvaag, Eirik Stavelin, Michael Karlsson, and Aske Kammer. The hyperlinked scandinavian news ecology: The unequal terms forged by the structural properties of digitalisation. *Digital Journalism*, pages 1–25, 2018.

Per Erik Solberg, Arne Skjærholt, Lilja Øvrelid, Kristin Hagen, and Janne Bondi Johannessen. The norwegian dependency treebank. In *Ninth International Conference on Language Resources and Evaluation*, 2014.

Språkrådet. Orddeling og særskriving. `http://www.sprakradet.no/Vi-og-vart/hva-skjer/Aktuelt-ord/Orddeling-og-sarskriving/`, 2009.

Cathrine Stadsnes. Evaluating semantic vectors for norwegian. Master's thesis, University of Oslo, 2018.

Milan Straka and Jana Straková. Universal dependencies 2.0 models for UDPipe (2017-08-01), 2017. URL `http://hdl.handle.net/11234/1-2364`. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Saatviga Sudhahar, Giuseppe A. Veltri, and Nello Cristianini. Automated analysis of the us presidential elections using big data and network analysis. *Big Data & Society*, 2(1), 2015.

Charles Sutton, Andrew McCallum, et al. An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4): 267–373, 2012.

Ole Tange. Gnu parallel - the command-line power tool. *;login: The USENIX Magazine*, 36(1):42–47, Feb 2011. URL `http://www.gnu.org/s/parallel`.

Xavier Tannier. Nlp-driven data journalism: Time-aware mining and visualization of international alliances. In *Proceedings of the 2016 IJCAI*

Workshop on Natural Language Processing meets Journalism, pages 52–56, 2016.

Tekstlaboratoriet and Uni Computing. The oslo-bergen tagger. `http://www.tekstlab.uio.no/obt-ny/english/index.html`, 2014.

Erik F. Tjong Kim Sang and Sabine Buchholz. Introduction to the conll-2000 shared task: Chunking. In *Proceedings of the 2nd Workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning - Volume 7*, ConLL '00, pages 127–132. Association for Computational Linguistics, 2000.

Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 142–147. Association for Computational Linguistics, 2003.

Samia Touileb and Katherine Duarte. Getting to know large newsflows: Automatically induced information structures as keyphrases for news content analysis. In *Proceedings of the 2016 IJCAI Workshop on Natural Language Processing meets Journalism*, pages 35–40, 2016.

Samia Touileb, Truls Pedersen, and Helle Sjøvaag. Automatic identification of unknown names with specific roles. In *Proceedings of the Second Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 150–158, 2018.

G. V. Trunk. A problem of dimensionality: A simple example. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(3): 306–307, 1979.

Wouter Van Atteveldt, Jan Kleinnijenhuis, and Nel Ruigrok. Parsing, semantic networks, and political authority using syntactic analysis to extract semantic relations from dutch newspaper articles. *Political Analysis*, 16(4):428–446, 2008.

Erik Velldal, Lilja Øvrelid, and Petter Hohle. Joint ud parsing of norwegian bokmål and nynorsk. In *Proceedings of the 21st Nordic Conference on Computational Linguistics, NoDaLiDa, 22-24 May 2017, Gothenburg, Sweden*, pages 1–10. Linköping University Electronic Press, 2017.

Erik Velldal, Lilja Øvrelid, Eivind Alexander Bergem, Cathrine Stadsnes, Samia Touileb, and Fredrik Jørgensen. NoReC: The Norwegian Review Corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. European Language Resources Association (ELRA), 2018.

Jean-Philippe Vert, Koji Tsuda, and Bernhard Schölkopf. A primer on kernel methods. *Kernel Methods in Computational Biology*, pages 35–70, 2004.

Alessandro Vinciarelli and Sarah Favre. Broadcast news story segmentation using social network analysis and hidden markov models. In *Proceedings of the 15th international conference on Multimedia*, pages 261–264. ACM, 2007.

Finn-Erik Vinje. *Skriveregler*. Aschehaug, 7 edition, 1998. Gjennomgått av Norsk språkråd og anbefalt for offentlig bruk av Kulturdepartementet.

Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

Haoran Wen, E. A. Leicht, and Raissa M. D'Souza. Improving community detection in networks by targeted node removal. *Phys. Rev. E*, 83, 2011.

Marit R. Westergaard. Optional word order in wh-questions in two norwegian dialects: A diachronic analysis of synchronic variation. *Nordic Journal of Linguistics*, 28(2):269–296, 2005.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajič, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gökırmak, Anna Nedoluzhko, Silvie Cinková, Jan Hajič jr., Jaroslava Hlaváčová, Václava Kettnerová, Zdeňka Urešová, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher D. Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria dePaiva, Kira Droganova, Hěctor Martínez Alonso, Hans Uszkoreit, Vivien Mackentanz, Aljoscha Burchardt, Kim Harris, Katrin Marheinecke, Georg Rehm, Tolga Kayadelen, Mohammed Attia, Ali Elkahky, Zhuoran Yu, Emily Pitler, Saran Lertpradit, Michael Mandl, Jesse Kirchner, Hector Fernandez Alcalde, Jana Strnadova, Esha Banerjee, Ruli Manurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo Mendonça, Tatiana Lando, Rattima Nitisaroj, and Josie Li. Conll 2017 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada, 2017. Association for Computational Linguistics.

GuoDong Zhou and Jian Su. Named entity recognition using an hmm-based chunk tagger. In *proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 473–480. Association for Computational Linguistics, 2002.

Appendix A

Listing of source code

A.1 Named-Entity Recognition model

```
import tensorflow as tf
import tf.contrib.crf
import numpy as np

from collections import namedtuple

def char_embeddings(chars, len_chars, n_chars, config):
    batch_size = tf.shape(chars)[0]
    max_word = tf.shape(chars)[1]
    max_char = tf.shape(chars)[2]

    char_embeddings = tf.get_variable(
        'char_embeddings',
        [n_chars, config.char_embed_size],
        initializer = tf.variance_scaling_initializer(
            distribution = "uniform"
        ),
        trainable = True
    )

    char_ids = tf.reshape(chars, [-1])
    embedded_chars = tf.reshape(
        tf.nn.embedding_lookup(char_embeddings, tf.reshape(chars, [-1])),
        [batch_size * max_word, max_char, config.char_embed_size]
    )

    embed_mask = tf.expand_dims(
        tf.sequence_mask(
            tf.reshape(len_chars, [-1]),
```

```
            max_char,
            dtype = tf.float32
        ),
        axis = -1
    )

    return embedded_chars * embed_mask

def conv_max_pool(embeddings, len_words, config):
    pad_shape = [config.conv_kernel - 1] * 2
    conv = tf.layers.conv1d(
        tf.pad(
            embeddings,
            [[0, 0], [0, 0], pad_shape],
            constant_values = 1 # <PAD>
        ),
        filters = config.pool_size,
        kernel_size = config.conv_kernel,
        strides = 1,
        padding = 'SAME',
        use_bias = True,
        activation = 'relu'
    )

    max_pool = tf.reduce_max(tf.matrix_transpose(conv), axis = 2)
    pool_mask = tf.reshape(
        tf.sequence_mask(len_words, dtype = tf.float32),
        [-1, 1]
    )

    return max_pool * pool_mask


Config = namedtuple(
    'Config',
    [
        'batch_size',
        'char_embed_size',
        'conv_kernel',
        'depth',
        'dropout',
        'h_size',
        'learning_rate',
        'pool_size',
    ]
)

class Network:

    def __init__(self, name, batch, config, v_shape):
```

```
80          words, pos, gazetteer, chars, len_chars, labels, len_words = batch
81          n_words, n_pos, n_categories, n_chars, n_tags = v_shape
82
83          batch_size = tf.shape(words)[0]
84          max_words = tf.shape(words)[1]
85
86          embeddings = char_embeddings(chars, len_chars, n_chars, config)
87          embedding_pool = tf.reshape(
88              conv_max_pool(embeddings, len_words, config),
89              [batch_size, max_words, config.pool_size]
90          )
91
92          fw = tf.nn.rnn_cell.MultiRNNCell(
93              [tf.nn.rnn_cell.LSTMCell(config.h_size)
94               for _ in range(config.depth)]
95          )
96          bw = tf.nn.rnn_cell.MultiRNNCell(
97              [tf.nn.rnn_cell.LSTMCell(config.h_size)
98               for _ in range(config.depth)]
99          )
100
101         self.dropout = tf.placeholder(tf.float32, [])
102
103         features = tf.nn.dropout(
104             tf.concat([words, pos, gazetteer, embedding_pool], axis = 2),
105             keep_prob = 1 - self.dropout
106         )
107
108         output, _ = tf.nn.bidirectional_dynamic_rnn(
109             cell_fw = fw,
110             cell_bw = bw,
111             inputs = features,
112             sequence_length = len_words,
113             dtype = tf.float32
114         )
115         output = tf.concat(output, axis = 2)
116         output = tf.layers.dense(
117             tf.nn.dropout(output, keep_prob = 1 - self.dropout),
118             units = n_tags,
119             name = "output"
120         )
121
122         log_likelihood, transition = crf.crf_log_likelihood(
123             output,
124             tag_indices = labels,
125             sequence_lengths = len_words
126         )
127
128         # Viterbi decode
129         self.predict, self.score = crf.crf_decode(
```

```
130             output,
131             transition_params = transition,
132             sequence_length = len_words
133         )
134 
135         # Cross-entropy loss
136         self.loss = tf.reduce_mean(-log_likelihood, name = "loss")
137 
138         tvars = tf.trainable_variables()
139         gradients, _ = tf.clip_by_global_norm(
140             tf.gradients(self.loss, tvars),
141             clip_norm = 5.0
142         )
143 
144         optimizer = tf.train.AdamOptimizer(
145             config.learning_rate,
146             epsilon = 0.1
147         )
148         self.train = optimizer.apply_gradients(zip(gradients, tvars))
```

Graphic design: Communication Division, UiB / Print: Skipnes Kommunikasjon AS

uib.no

ISBN: 9788230848753 (print)
9788230866757 (PDF)