Evaluating Semantic Vectors for Norwegian

Cathrine Stadsnes

Thesis submitted for the degree of
Master in Informatics: Language and Communication
60 credits

Department of Informatics
Faculty of Mathematics and Natural Sciences

UNIVERSITY OF OSLO

31st January 2018

Evaluating Semantic Vectors for Norwegian

Cathrine Stadsnes

Evaluating Semantic Vectors for Norwegian

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

Abstract

In this work, we create and make available two benchmark data sets for evaluating models of semantic word similarity for Norwegian. While such resources are available for English, they did not exist for Norwegian prior to this project. We also produce large-coverage semantic vectors trained on a selection of various corpora using several popular word embedding frameworks. Finally, we demonstrate the usefulness of the created resources for evaluating performance of different word embedding models on the tasks of analogical reasoning and synonym extraction.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my two supervisors, Erik Velldal and Lilja Øvrelid, for the continuous guidance, support and advice they have provided during my work on this thesis.

I would also like to acknowledge the developers and researchers behind the tools and resources used in this project. A special thank you goes to Kunnskapsforlaget, for providing a digital version of the Norwegian synonym dictionary *Norske synonymer blå ordbok*.

Finally, I would like to thank my family and friends for supporting me throughout my years of study and in life generally.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, vector space models that implement a distributional approach to lexical semantics have become a subject of increasing research interest in the natural language processing (NLP) community. The basic idea of distributional semantics is that the meaning of a word can be inferred from its distributional properties in a very large collection of texts, i.e., a corpus.

Traditionally, distributional semantic models represent meanings of words as vectors derived from their counts of co-occurrences with other words. However, since most words only occur in the context of just a few others, these vectors will be sparse and very high-dimensional. Thus, machine learning-based methods for generating dense and low-dimensional semantic vectors have been introduced. Such vectors are also referred to as *word embeddings*.

Word embedding models have been shown to capture rich semantic, syntactic and conceptual information about words and their meanings. For this reason, the models have proved to be useful for a variety of NLP applications. Word embeddings are commonly used to compute semantic similarity between words. For instance, in information retrieval, we want to obtain not only documents including the query words, but also documents including words with meanings similar to the query words. Furthermore, word embeddings have become even more widely used as input representations for artificial neural networks, which are employed in a range of downstream NLP tasks, such as sentiment analysis and text classification. The question of how to evaluate these models is therefore highly relevant.

As task-based or *extrinsic* evaluation can be expensive, it may be desirable to quantify performance properties of vector models prior to downstream use. There exists a range of benchmark data sets that facilitate such *intrinsic* evaluation of model performance for English. Generally, the evaluation is performed by measuring correlation between the distributional semantic models and the evaluation resource. For

example, the WordSim-353 data set of Finkelstein et al. (2002) can be used to evaluate model performance in determining semantically related words, e.g., *car* and *gasoline*. The more recent SimLex-999 data set of Hill, Reichart and Korhonen (2015) can be used to evaluate model performance in determining semantically similar words, e.g., *car* and *truck*. Furthermore, synonyms from WordNet (Miller, 1995) are commonly used to evaluate model performance in extracting synonyms. Alternatively, the TOEFL (Test of English as a Foreign Language) data set of Landauer and Dumais (1997) can be used. Moreover, the Google Analogies Dataset introduced by Mikolov, Chen et al. (2013) is popularly used to evaluate model performance in recognizing so-called analogies, like *granddaughter* is to *grandson* as *sister* is to *brother*.

In contrast, Norwegian remains an under-resourced language in the sense that many core NLP resources are still missing. This includes resources for evaluating distributional semantic models. For this reason, the first aim of this project is to create and make available two benchmark data sets that enable intrinsic evaluation of model performance for Norwegian. The first data set will be created by semi-automatically translating and adapting the existing Google Analogies Dataset (Mikolov, Chen et al., 2013) to Norwegian, for the task of analogical reasoning. Since this resource does not provide any context for the words, translation must be followed by manual inspection and post-processing. The second data set will be created by extracting words and associated synonyms from the digital version of *Norske synonymer blå ordbok*, which is an existing Norwegian synonym dictionary created by Dag Gundersen and published by Kunnskapsforlaget, for the task of synonym extraction.

The second aim of this project is to evaluate different distributional semantic models using the data sets created in the context of this thesis. Furthermore, we will attempt to isolate the effects of corpus, pre-processing of text and word embedding framework. However, our intention is not to discover the perfect word embedding model and hyperparameter optimization is thus out-of-scope of this project. Rather, we seek to demonstrate the usefulness of the created resources for ranking the relative performance of different word embedding models. As a by-product of the evaluation, we will make available large-coverage semantic vectors for Norwegian trained on a selection of corpora, such as the Norwegian Newspaper Corpus, the Norwegian Web as Corpus (Guevara, 2010) and the NBDigital Corpus, using several popular word embedding frameworks like word2vec (Mikolov, Chen et al., 2013), GloVe (Pennington, Socher and Manning, 2014) and fastText (Bojanowski et al., 2016).

1.1 Outline

Chapter 2 gives a theoretical overview of distributional semantics and methods for generating word embeddings. Moreover, we present previous work on evaluation and benchmark data sets.

Chapter 3 describes the work on creating an analogy resource for Norwegian based on the semi-automatic translation of the existing Google Analogies Dataset for English. In this chapter, we mainly focus on the manual inspection and post-processing following translation. Moreover, we explain the method used for evaluation.

Chapter 4 details the process of creating a synonym resource for Norwegian based on the digital version of *Norske synonymer blå ordbok*. We describe the extraction of words and synonyms as well as spelling variants. Furthermore, we explain the metrics we implement for evaluation.

Chapter 5 provides a description of the various text corpora used for training word embeddings for Norwegian, and the configuration of tools used for text pre-processing. Furthermore, we present different word embedding frameworks and hyperparameters.

Chapter 6 presents evaluation experiments and results for different word embedding models using the created evaluation resources. The first part describes results for the task of analogical reasoning, whereas the second part discusses results for the task of synonym extraction. Finally, we present a few supplementary experiments.

Chapter 7 concludes the work of this project and proposes suggestions for future work.

Chapter 2

Background

In this chapter, we will give an overview of the theoretical fundamentals of distributional semantics, and specifically in the form of vector space models. Furthermore, we will describe commonly used methods for generating dense semantic vectors. Finally, we will present previous work on evaluation and benchmark data sets.

2.1 Distributional hypothesis

Distributional semantics involves theories and methods for determining semantic similarities between words based on their distributional properties in large collections of text, i.e., corpora. The underlying idea is the so-called *distributional hypothesis*, which suggests that two words that occur in similar contexts tend to have similar meanings (Harris, 1954).

Consider the following sentences:

> He handed her a glass of *blacque*.
>
> *Blacque* gives me a headache.
>
> *Blacque* is made from French grapes.

Without having any prior knowledge of the word *blacque*, we can easily understand from these sentences that it is an alcoholic beverage like wine. Similarly, if we count the words co-occurring with *blacque*, we will tend to observe words like *glass*, *headache* and *grapes*. Such words also tend to co-occur with the word *wine* and we can assume that *blacque* and *wine* have similar meanings. Hence, comparing meaning is reduced to comparing contexts. As we shall see in the next subsections, however, what counts as *words* or *contexts* can vary.

2.1.1 Defining words

Various kinds of pre-processing methods can be applied to a corpus to define which linguistic entities are considered as words, e.g., tokenization, lemmatization, stop word removal and stemming. Tokenization is the task of splitting text into words or other units, called *tokens*. Lemmatization usually refers to the morphological analysis of words and returning their dictionary forms, known as *lemmas*. Moreover, stop word removal is the task of removing extremely frequent words, which often do not provide relevant context. Finally, stemming is the process of reducing inflected words to their root form or prefix.

To illustrate, a simple sentence is provided in Example 1.[1] Example 2 shows the same sentence with tokenization applied to it. Similarly, Example 3 shows the sentence after lemmatization. In Example 4 stop words have been removed and in Example 5 stemming has been applied.

1. The programmer's programs had been programmed.
2. the programmer 's programs had been programmed .
3. the programmer 's program have be program .
4. programmer program program
5. program program program

2.1.2 Defining contexts

Similarly, contexts can be defined in different ways, and may vary from entire documents, paragraphs or sentences to single words. For instance, we can define context to be the n words left and right of a target word, known as a context window. Moreover, we can define context using a bag-of-words approach. In this case, context is defined to be all co-occurring words of a target word, represented as a "bag" that ignores word order, either on a sentence or document level. Also, we can define context to be the grammatical relations of a target word with neighbouring words.

For instance, consider the following sentence:

> I eat *salad* for lunch.

The grammatical context features of *salad* would be {dir_obj(eat), prep_for(lunch)}. Furthermore, a context window of $n = 1$ would give {left:eat, right:for} and a bag-of-words approach would give {I, eat, for lunch} as context features.

[1] Example from INF4820 lecture slides: http://www.uio.no/studier/emner/matnat/ifi/INF4820/h15/slides/04_distributional_print.pdf

The type of contexts tend to dictate the kind of semantic similarity that will be captured by the distributional methods. As formulated by Schütze and Pedersen (1993), two words are *syntagmatic associates* if they are typical neighbours of each other but tend to have different grammatical roles, e.g., *drink* and *coffee* or *teacher* and *school*. This kind of relation is also referred to as similarity in domain or *relatedness*, and is often associated with larger context windows. Furthermore, two words are *paradigmatic parallels* if they have similar close neighbours to the left or right and are typically substitutable for each other, e.g., *eat* and *drink* or *wrote* and *remarked*. Such a relation is also known as similarity in content or *sameness*, and tend to be associated with smaller context windows or grammatically defined notions of context.

2.2 Vector semantics

Distributional semantic models generally represent words as vectors and so distributional methods are often referred to as *vector semantics*. The vectors are typically based on a co-occurrence matrix, also called a word–word matrix or word–context matrix, which represents how often words co-occur in some context in some training corpus (Jurafsky and Martin, 2009).

2.2.1 Count-based word vectors

Traditionally, distributional semantic models derive meanings of words by constructing vector representations of them based on *counts* of their co-occurrences with other words. Thus, such vectors are commonly referred to as *count-based*. More formally, a word w_i can be represented by a set of n context features, i.e., the context vector $\vec{x_i} = [\vec{x_{i1}}, ..., \vec{x_{in}}]$, where the value of each feature is how often the word and that feature co-occur. A given word can then be seen as a point in a coordinate system, or semantic space, where each context feature is mapped to a dimension $j \in [1, n]$.

Table 2.1 shows a simple example of a word–word co-occurrence matrix for six words. For m words and n context words, the matrix is of dimensionality $m \times n$. Each row corresponds to a target word and each column corresponds to a context word. Furthermore, each cell gives the number of times the target word and the context word co-occur. For example, the word vector, or row vector, of *vehicle* is $\vec{vehicle} = [5, ..., 0, 2, 1, 0, 2]$, meaning that we have observed *vehicle* and *car* to co-occur five times, *vehicle* and *gasoline* to co-occur two times and so on. Sometimes, m and n are equal, i.e., the context features are every word in the vocabulary. Since most words only occur in the context of just a few others, the vectors will be sparse with most values equal to zero, and at the same time very high-dimensional. Figure 2.1 shows a vector

	car	...	vehicle	gasoline	computer	technology	crash
car	0	...	5	2	0	0	2
vehicle	5	...	0	2	1	0	2
gasoline	2	...	2	0	0	0	0
computer	0	...	1	0	0	4	2
technology	0	...	0	0	4	0	0
crash	2	...	2	0	2	0	0

Table 2.1: A simple word–word co-occurrence matrix for six words.

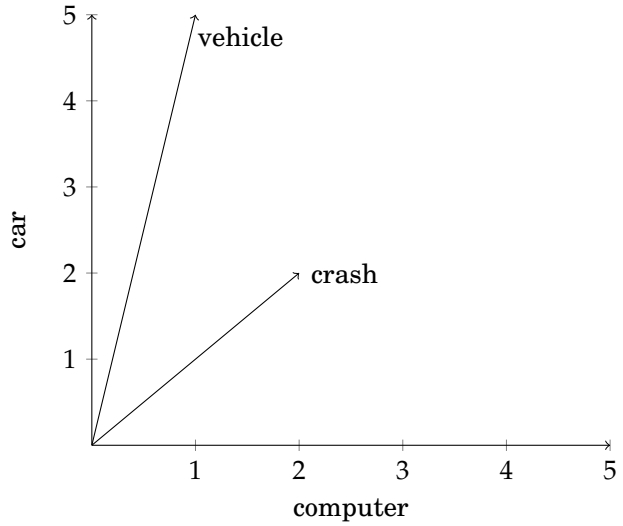

Figure 2.1: A vector space representation of the words *vehicle* and *crash* projected to two dimensions.

space representation of the words *vehicle* and *crash* from Table 2.1, projected to only two dimensions corresponding to the words *computer* and *car*.

2.2.2 Weighing words

The count-based vectors described above are prone to skewness because some words co-occur frequently with many other words (Jurafsky and Martin, 2009). For example, the word *buy* is assumed to frequently co-occur with the word *milk*, but also with many other words in the vocabulary. Hence, the observation that *milk* frequently co-occurs with *buy* might not be very indicative of the semantics of *milk* in particular.

For this reason, association measures based on some weighting, rather than raw counts, are commonly used. One such measure is the pointwise mutual information (PMI) measure which, when applied to association between words, measures how much more often than chance two words co-occur. For a target word w and a context word c, PMI is

defined as in Equation (2.1).

$$PMI(w,c) = \log_2 \frac{P(w,c)}{P(w)P(c)} \tag{2.1}$$

Positive PMI values suggest that the two words co-occur more often than by chance, whereas negative PMI values suggest the opposite. However, in many cases we do not have sufficient amounts of training data to rely on negative PMI values (Jurafsky and Martin, 2009). Therefore, the positive pointwise mutual information (PPMI) measure, which replaces all negative PMI values with zero, is more commonly used.

Furthermore, association measures can take the form of statistical hypothesis tests. One such example is the *t-test*. Here, the null hypothesis is that the two co-occurring words are independent. If the calculated *t-value* is above some threshold, we can reject the null hypothesis.

2.2.3 Vector similarity

In order to define semantic similarity between two words v and w, we need a way to measure the similarity between their vectors. The idea is that words with similar vectors are similar in meaning because they occur in similar contexts. One standard measure of vector similarity is the Euclidean distance, which computes the length of the difference between two vectors. The Euclidean distance between two vectors $\vec{v}$ and $\vec{w}$ can be computed as shown in Equation (2.2).

$$d(\vec{v},\vec{w}) = \sqrt{\sum_{i=1}^{n}(\vec{v}_i - \vec{w}_i)^2} \tag{2.2}$$

However, vectors may vary in length since frequent words tend to co-occur with more words. This will cause length bias and affect the similarity measure. One way to reduce the frequency effects is to first normalize the vectors to have unit length, i.e., $\|\vec{v}\| = 1$, by dividing each element of the vectors by the vector length, or norm. The norm of a vector is defined as in Equation (2.3).

$$\|\vec{v}\| = \sqrt{\sum_{i=1}^{n} v_i^2} \tag{2.3}$$

Another commonly used measure of vector similarity is the cosine (Jurafsky and Martin, 2009), which already accounts for length bias by normalizing the vectors. The cosine measures proximity, rather than distance, between two vectors and computes similarity as a

function of the angle between them. If we pre-normalize each vector by dividing it by its length, the cosine is the same as the dot product. The cosine between two vectors $\vec{v}$ and $\vec{w}$ can be computed as shown in Equation (2.4). The cosine ranges from 0 for orthogonal vectors, indicating that the two words are completely dissimilar in meaning, to 1 for vectors pointing in the same direction, indicating that the two words are exactly similar in meaning. Alternatives to the cosine is the Jaccard and Dice measures, both based on computing number of overlapping context features.

$$\cos(\vec{v},\vec{w}) = \frac{\sum_{i=1}^{n} v_i w_i}{\sqrt{\sum_{i=1}^{n} v_i^2}\sqrt{\sum_{i=1}^{n} w_i^2}} = \frac{\vec{v}\cdot\vec{w}}{\|\vec{v}\|\|\vec{w}\|} \tag{2.4}$$

2.2.4 Dense vectors

In recent years, another approach for representing words as vectors has been introduced. The previously described count-based vectors are both long and sparse with most values equal to zero, and for some applications it is better to represent words as short and dense vectors with most values not equal to zero. Such vectors are often referred to as *word embeddings*, as the words are embedded into a low-dimensional vector space.

There are several possible advantages of this approach. First, since dense vectors comprise fewer parameters compared to sparse vectors, they are easier to employ as features to represent words in machine learning systems (Jurafsky and Martin, 2017). Second, embeddings lead to improved computational efficiency and may generalize better to unseen data (Levy, Goldberg and Dagan, 2015). Third, embeddings may capture synonymy better (Jurafsky and Martin, 2017). There are various methods for generating dense embeddings, which will be described in the following.

Singular value decomposition

A traditional approach to generating dense vectors is to apply dimensionality reduction to the high-dimensional word–word matrix. Several such dimensionality reduction methods are available and one much used example is singular value decomposition (SVD).

SVD approximates a matrix using fewer dimensions. Jurafsky and Martin (2017) briefly explains the dimensionality reduction as rotating the axes of the original data set into a new space. In this new space, the highest order dimension captures the most variance in the original data set, the second highest order dimension captures the second most

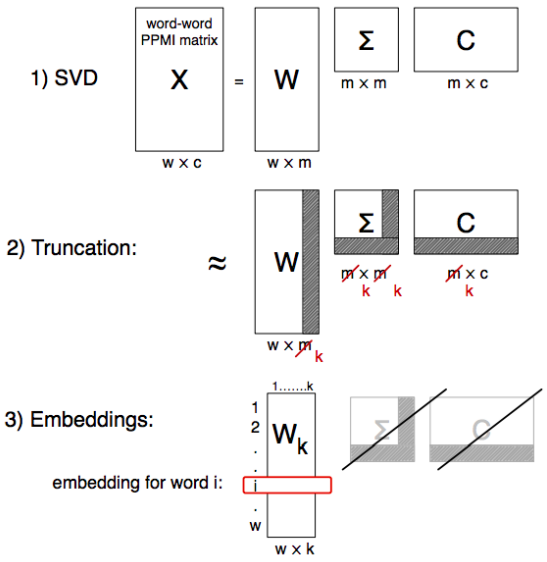

Figure 2.2: Singular value decomposition (SVD) applied to a word–word matrix.

variance and so forth. In this way, much of the original variation is captured with fewer dimension.

More formally, for w words and c context words, the $w \times c$ word–word matrix X is factorized into three matrices, e.g., W, $\sum$ and C. The latter two are discarded and W is truncated, i.e., only the top k dimensions are used. The dense, k-dimensional rows of W are used to represent words and can substitute the high-dimensional rows of X, as illustrated in Figure 2.2 (Jurafsky and Martin, 2017).

Word2vec CBOW and Skip-gram

While traditional dimensionality reduction techniques are applied to the initial full co-occurrence matrix, two more recent and popular methods for generating dense embeddings directly from the data are the Continuous Bag-of-Words (CBOW) and Skip-gram model architectures introduced by (Mikolov, Chen et al., 2013) and implemented in the *word2vec* toolkit.[2] These models learn embeddings by training a simple artificial neural network to predict neighbouring words. Thus, such models are often referred to as *prediction-based* as opposed to *count-based*. CBOW and Skip-gram models are computationally effective and pre-trained models for English are available online.

The shared intuition is that embeddings that are good at predicting neighbouring words are also good at representing word similarity because semantically similar words tend to occur in similar contexts.

[2] https://code.google.com/p/word2vec/

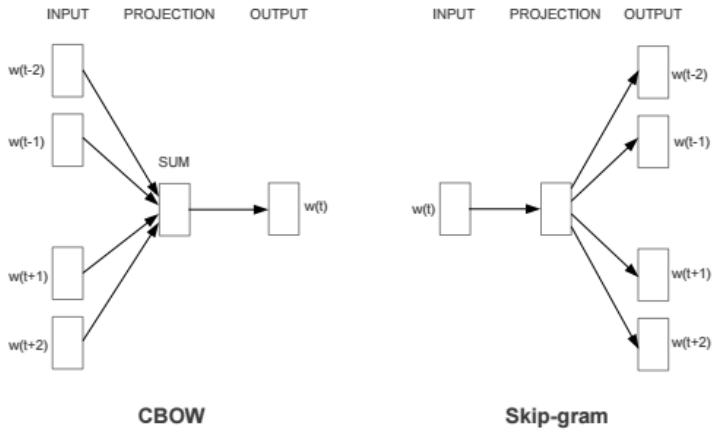

Figure 2.3: Visualization of the Continuous Bag-of-Words (CBOW) and Skip-gram model architectures of word2vec.

Hence, the neural models try to learn embeddings that are maximally similar to the embeddings of their neighbouring words and minimally similar to the embeddings of the words which do not occur close by (Jurafsky and Martin, 2017). However, while CBOW learns to predict the target word based on the context words, Skip-gram learns to predict the context words given the target word. Figure 2.3 shows a visualisation of the two model architectures (Mikolov, Chen et al., 2013).

More formally, given a target word w_t, the Skip-gram model predicts each of its n neighbouring words to the left and right, e.g., for $n = 2$ these are $[w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}]$. The target word input vector is the input for the prediction and we check whether each of the context words output vectors are the closest to it, in terms of cosine distance, among the words in the vocabulary. The cosine similarities are turned into probabilities using the *softmax* function, and the outcome determines whether we adjust the context word vectors.

Similarly, the CBOW model is also based on prediction. However, it predicts the target word w_t given its n neighbouring words to the left and right, e.g., for $n = 2$ the target word w_t is predicted from the context words $[w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}]$. Now, the input for the prediction is the average input vectors of the context words, i.e., word order is ignored, and we check whether the target word output vector is the most similar to it.

FastText

Recently, Facebook AI Research open sourced another effective method for learning word vector representations, namely fastText (Bojanowski

et al., 2016). FastText is essentially an extension of the original word2vec CBOW and Skip-gram models, but while these models represent each word by a distinct vector, and thus disregard word morphology, fastText takes into account the internal structure of words.

More precisely, fastText embeddings incorporate subword information by representing a word by the sum of the vector representations of its character n-grams (Bojanowski et al., 2016). For example, with $n = 3$ the word *calmly* is represented by the character n-grams `<ca`, `cal`, `alm`, `lml`, `mly`, `ly>` in addition to `<calmly>`, where `<` and `>` indicate the start and end of words. A vector representation is associated to each of these character n-grams and the word *calmly* is represented by the sum of these.

This model is useful for morphologically rich languages which tend to contain words that are rare in the training corpus. Because vector representations can be shared across words, the model might be able to learn reliable representations even for rare words. Also, fastText models can generate vector representations for words out of the vocabulary. Such words are represented as an average of the vector representations of their character n-grams.

GloVe

Another machine learning-based embedding model is GloVe (Pennington, Socher and Manning, 2014), short for Global Vectors, which attempts to combine global count-based models and prediction-based models with local context windows. In the GloVe model, word vectors are trained on the non-zero entries of the global word–word co-occurrence matrix and the training objective is to learn word vector representations so that the dot product between them is equal to the logarithm of the probability of the two words co-occurring (Pennington, Socher and Manning, 2014).

The underlying intuition is that the quantitative relation, i.e., *ratio*, of such co-occurrence probabilities may encode meaning in some form. As exemplified by Pennington, Socher and Manning (2014), we can study the relation between the two words $i = ice$ and $j = steam$ by looking at the ratio of their co-occurrence probabilities with other words k, denoted by P_{ik}/P_{jk}. For example, for the word $k = solid$, the ratio is assumed to be large because *solid* is much related to *ice* but not related to *steam*. Similarly, for the word $k = gas$, the ratio should be small since *gas* is not related to *ice* but much related to *steam*. For words that are either related to both *ice* and *steam* or not related to either of them, the ratio will be close to one. Pennington, Socher and Manning (2014) point out that the ratio of probabilities better distinguish relevant words from irrelevant words as well as between two relevant words compared to

raw probabilities.

2.3 Evaluation of word embeddings

Generally, evaluation of distributional semantic models is performed by giving the models a task to perform and comparing their results with some given standards. We can distinguish two types of evaluation, i.e., *extrinsic* and *intrinsic*, although the boundary between the two might be unclear. Extrinsic evaluation is performed by adding the word vector representations as features into a downstream NLP task, such as text classification, and assessing whether performance is improved. However, as task-based evaluation can be expensive, it may be desirable to evaluate the quality of the models prior to downstream use. Such intrinsic evaluation attempts to measure correlation between the characteristics of the models and some predefined criteria, typically in the form of human judgments. Ideally, the evaluation would predict how well the model performs in downstream applications. In the next subsection, we will present previous work on intrinsic evaluation of distributional semantic models.

2.3.1 Related work and benchmark data sets

There has been a long tradition of evaluation studies of distributional semantic models. One such study was performed by Baroni, Dinu and Kruszewski (2014), who compared count-based and prediction-based models on a range of tasks. Some of these tasks are new, but a majority of them have been used in evaluation studies for many years.

In their experiments, they trained several examples of both model types on a 2.8 billion token corpus for English, constructed by concatenating ukWaC, the English Wikipedia and the British National Corpus, using context windows of two and five words to the left and right of the target word. For the count-based models, they used both full and compressed vectors. For both model types, they varied the vector sizes from 200 to 500 in steps of 100. Furthermore, they considered the 300K most frequent words for target and context words. The count-based models were generated using the DISSECT toolkit[3] and the prediction-based models were trained using the CBOW model implemented in the word2vec toolkit. They tested their models on different benchmark tasks, which will be detailed in the following.

[3] http://clic.cimec.unitn.it/composes/toolkit/

Word pair	SimLex-999	WordSim-353
coast–shore	9.00	9.10
clothes–closet	1.96	8.00

Table 2.2: Example of the difference in rating scores in the SimLex-999 and WordSim-353 data sets.

Similarity and relatedness

One task of distributional semantic models is to automatically determine semantically similar words, like *car* and *vehicle*, or semantically related words, such as *car* and *gasoline*. Semantically similar words have very similar meanings, whereas semantically related words are not necessarily similar in meaning at all. Baroni, Dinu and Kruszewski (2014) tested their models on a semantic relatedness task using the WordSim-353 data set of Finkelstein et al. (2002) which contains 353 English word pairs with human-assigned similarity scores. The data set was created by human subjects estimating the relatedness of the word pairs on a scale from 0–10, where 0 describes that the words are totally unrelated and 10 describes that the words are closely related or identical. For example, the word pair *plane* and *car* got an average score of 5.77. Model performance is measured in terms of correlation between the cosine similarity scores for the word pairs computed by the model and the corresponding average human-assigned scores. Agirre et al. (2009) further split this data set into similarity and relatedness subsets.

While WordSim-353 focuses mostly on relatedness, but also similarity, the more recent SimLex-999 data set (Hill, Reichart and Korhonen, 2015) quantifies semantic similarity exclusively. This data set was created by mining the opinions of 500 annotators from Amazon Mechanical Turk[4], which is a crowdsourcing Internet marketplace. The annotators rated how similar word pairs are by moving a slider, giving words that are related, but not actually similar, low scores. Table 2.2 illustrates the difference in the SimLex-999 and WordSim-353 ratings. The word *coast* is very similar in meaning to the word *shore* as well as very much related to it. Thus, this word pair has both high SimLex-999 and WordSim-353 scores. In contrast, *clothes* is not very similar to *closet*, but clearly related. Hence, this word pair has a low SimLex-999 score but a high WordSim-353 score.

Alternatively, model performance can be evaluated on a standardized synonym test. The TOEFL (Test of English as a Foreign Language) data set of Landauer and Dumais (1997) contains 80 multiple-choice questions consisting of one target word along with four synonym candidates. The models should correctly choose the candidate that is most similar to the target word. To solve this task, the models compute cosine similarity scores between each candidate word vector and the

[4]https://www.mturk.com/

target word vector and select the one that gives the highest score. Model performance is then evaluated in terms of accuracy of correct answers.

The word similarity, or relatedness, task has been shown to be valuable. However, Batchkarov et al. (2016) suggest that this task possesses limitations because it provides only an approximation of the quality of a distributional semantic model. Furthermore, they point out that word similarity data sets assume that there is one correct score for each word pair, which can vary much across data sets as their definition of similarity might differ. Moreover, since the data sets are typically small in size, their quality measures may vary substantially.

Concept categorization

Another task sometimes assigned to distributional semantic models is to group words into semantic categories. More precisely, given a set of nominal concepts, the models should group them into categories. As exemplified by Baroni, Dinu and Kruszewski (2014), *helicopters* and *motorcycles* should be grouped to the *vehicle* class and *dogs* and *elephants* to the *mammal* class. The task presented to the models is considered an unsupervised clustering task and the word vectors are clustered into n groups. Model performance is evaluated in terms of the extent to which the clusters include words from only one correct semantic category. This task is arguably one step closer to what could be considered extrinsic evaluation, as performance also depends on the given clustering algorithm.

Selectional preferences

For a selectional preferences task, Baroni, Dinu and Kruszewski (2014) used data sets of verb–noun pairs rated by humans for how typical a noun is of a verb, either as a subject or as an object. For example, *people* obtained a high average score as a subject of *to eat* and a low score as an object, e.g., people eat but we rarely eat people. For each verb, a vector is obtained by averaging the vectors of the 20 nouns that are associated the most to it as subjects or objects. The cosine similarity between the target noun vector and the relevant vector is measured and performance is evaluated in terms of correlation of these scores with the average human-assigned ratings. Actually, this is the only task on which they found that count-based models performed comparably well as prediction-based models.

Analogical reasoning

A popularly used task for evaluating distributional semantic models is analogical reasoning, introduced by Mikolov, Chen et al. (2013) particularly for evaluating prediction-based models. Mikolov, Yih and Zweig (2013) showed that word embeddings learned by artificial neural networks capture syntactic and semantic regularities in language well. These regularities include various relations, for instance the *gender* relation between *man* and *woman* or *king* and *queen*, or the grammatical relation between *scream* and *screams* or *play* and *plays*. Such relations were demonstrated to be reflected in word vector offsets.

There are several ways of expressing analogy questions in a word analogy task, but a general notation is *A* is to *A** as *B* is to *B**. More specifically, the model is given two word pairs that share some relation, and must infer the fourth word *B** based on the other three. This is similar to answering questions like *mother* is to *father* as *grandmother* is to ?, i.e., a semantic analogy, or *young* is to *younger* as *fast* is to ?, i.e., a syntactic analogy. The model subtracts the vector of *A** from the vector of *A* and adds the vector of *B*. Furthermore, the model compares the resulting vector to the vectors of each word in the vocabulary excluding *A*, *A** and *B*, and chooses the highest-scoring word, in terms of cosine similarity, as the answer. For the two examples, the correct answers would be *grandfather* and *faster*, respectively. For evaluation, Mikolov, Chen et al. (2013) developed the Google Analogies Dataset, which contains 19,544 analogy questions covering 5 semantic and 9 syntactic relation types. Within each relation type, word pairs are provided as analogy questions and model performance is measured by accuracy of correctly answered questions.

Baroni, Dinu and Kruszewski (2014) showed that prediction-based models outperform count-based models on most of the discussed tasks, as well as being robust across various parameter settings. However, there exist studies that have come to other conclusions. For instance, Levy and Goldberg (2014) suggested that the linguistic regularities captured by the prediction-based embeddings are not a consequence of the model itself and sparse vector representations also encode relational similarities in the form of vector offsets. Similarly, these regularities can be recovered by applying vector operations and sparse vector representations may also recognize analogies well.

More precisely, Levy and Goldberg (2014) evaluated both count-based models and neural word embeddings on three analogy data sets, e.g., the MSR data set (Mikolov, Yih and Zweig, 2013), the Google Analogies Dataset (Mikolov, Chen et al., 2013) and the SemEval data set (Jurgens et al., 2012). Both the MSR data set and Google Analogies Dataset present questions like *A* is to *A** as *B* is to *B** and the model must guess the fourth word *B** from all other words in the vocabulary. In contrast,

the SemEval data set contains semantic relations, each exemplified by a few word pairs, and for each relation the model must rank a set of target word pairs according to the degree to which the relation applies. For all data sets, performance is measured in terms of accuracy. Levy and Goldberg (2014) showed that count-based vectors perform better on analogies in some of the semantic relations such as the ones related to geography, as well as in some of the syntactic relations, such as superlatives. Moreover, the prediction-based vectors perform better on most verb inflections, comparatives and family relations, among others.

Synonym extraction

Leeuwenberg et al. (2016) performed an evaluation study on the task of synonym extraction. They carried out different experiments using the word2vec CBOW and Skip-gram models trained on a 150 million word subset of the NewsCrawl corpus from the 2015 Workshop on Statistical Machine Translation[5], for both English and German. For pre-processing, they lower-cased, tokenized and applied digit conflation to the corpus. Furthermore, to ensure that the vectors are of minimal quality, they only considered words occurring at least 10 times in the corpus. They varied different parameter settings, such as the context window size, vector dimensionality and types of word vectors, and analyzed their effects on the precision of the extracted synonyms. Moreover, they proposed a new measure for computing cosine similarity relative to other similar words, which we will return to shortly.

For evaluation, they used the synonyms from WordNet[6] for English and GermaNet[7] for German, both including part-of-speech tags. A given word's part of speech is not known in the experiments, and they considered the synonyms of each word to be the synonyms of all the parts of speech it can have in WordNet or GermaNet. They evaluated their experiments in terms of precision, recall and F-measure. They defined precision to be the proportion of correctly predicted synonym word pairs from all predictions, and recall to be the proportion of correctly predicted synonym word pairs from all synonym word pairs that are present in WordNet or GermaNet.

To illustrate, we assume that the particular word *cake* is a noun only and its synonyms in WordNet are *pie*, *brownie*, *cheesecake* and *gateau*. One model may predict the synonyms of *cake* to be *pie*, *brownie*, *waffle* and *muffin*. The correctly predicted synonym word pairs are then $(cake, pie)$ and $(cake, brownie)$. As synonymy is a symmetric relation, they also considered $(pie, cake)$ and $(brownie, cake)$ for evaluation. Thus, there are four correctly predicted word pairs for *cake*. Similarly, there

[5]http://www.statmt.org/wmt15/translation-task.html
[6]https://wordnet.princeton.edu/
[7]http://www.sfs.uni-tuebingen.de/GermaNet/

are a total of eight word pairs that are considered present in WordNet for *cake*. For rare word pairs, this approach will result in a high number of false positives, i.e., predicted synonym word pairs that are not present in WordNet or GermaNet, and false negatives, i.e., synonym word pairs that are present in WordNet or GermaNet but not predicted. Consequently, precision and recall scores are expected to be low.

For both English and German, the calculated precision, recall and F-measure scores, using the 1st, 2nd and 4th most similar words as synonyms, were very low. In many cases, the most similar words might be good suggestions, but they are simply not covered in WordNet or GermaNet. In other cases, different types of relations rather than synonymy might be captured, such as antonymy, hypernymy or hyponymy. Therefore, they investigated the most similar words further. For 150 randomly chosen English words, they looked at the most similar and the second most similar words, and manually categorized them. The number of human-judged synonyms, compared to the number of synonyms given by WordNet, was around twice as large. The construction of a resource like WordNet or GermaNet requires the annotator to manually think of and add synonyms for a given word and its word senses, which can be a challenging task. In contrast, it may be easier to be presented two words and answer whether they are synonyms. This suggests that the notion of synonymy in these resources are possibly too strict to make them suitable as evaluation benchmarks and that the actual precision may be higher.

In general, Leeuwenberg et al. (2016) showed that the CBOW models gave higher precision than the Skip-gram models for both languages, and pointed out that CBOW vectors tend to be more syntactical compared to Skip-gram vectors. Furthermore, the optimal context window size was found to be around 4 for English and 8 for German. The difference in optimal context window size was assumed to be due to the difference in the distribution of synonyms in WordNet and GermaNet. More precisely, WordNet contains synonyms for nouns, verbs, adjectives and adverbs, whereas GermaNet does not include synonyms for adverbs. Possibly, adverbs require a small context window to be predicted, thus decreasing the optimal window size.

Moreover, they showed that cosine similarity alone might not be a good indicator to determine if two words are synonymous. To improve precision, they proposed a new measure, the *relative cosine similarity*, for calculating similarity relative to the top n most similar words, as shown in Equation 2.5. The relative cosine similarity will give words with a high cosine similarity compared to other words in the top n most similar words a high score. Words will get a lower score if all words in the top n most similar words have nearly equal cosine similarity scores. Furthermore, they noticed that when a synonym occurs in the top 10 most similar words, the cosine similarity is usually much higher than of the other words in the top 10. This was also found to be the case

for inflections and contrastives, but not co-hyponyms, related words or unknowns. They showed that calculating similarity relative to other words may improve precision, as some words can be filtered out.

$$rcs_n(\vec{w}_i, \vec{w}_j) = \frac{\cos(\vec{w}_i, \vec{w}_j)}{\sum_{w_c \in TOP_n} \cos(\vec{w}_i, \vec{w}_c)} \tag{2.5}$$

They also explored the advantages of using a part-of-speech (POS) tagger as a way of introducing some supervision and thus aiding the extraction of synonyms. The motivation for using POS tagging is homography, i.e., a word can have several word senses. For example, the word *phone* may be a noun, e.g., a telephone or a speech of sound, or a verb, e.g., to phone. Similarly, *call* may be a noun, e.g., a cry, or a verb, e.g., to call or to name. Homography can be a problem in synonym extraction because the vector for *phone* is trained on all the word senses of *phone* that occur in the corpus. Consequently, for the less frequent meanings, it can be difficult to find synonyms. They showed that POS tagging can improve performance by separating some of the word senses, filtering words that are not grammatically similar enough and not extracting synonyms for word categories that gave very few synonyms (Leeuwenberg et al., 2016).

Finally, they evaluated their resulting system using both intrinsic and extrinsic evaluation. They did a manual evaluation of the extracted synonyms, by taking a random sample of 200 word pairs for each language. They used two annotators per language, categorizing the word pairs as synonyms, non-synonyms or unknown. The resulting precision is lower for German than for English, but the number of found word pairs is bigger. The extracted synonyms were also used in machine translation evaluation, in the synonym module of the *Meteor* evaluation metric. They tested if the score correlates better with human judgments after adding the synonyms, and showed that they do, resulting in an improved evaluation score.

Chapter 3

Creating the Norwegian Analogy Test Set

In this chapter, we will detail the work on creating the Norwegian Analogy Test Set based on the semi-automatic adaption of the existing Google Analogies Dataset for English. We will mainly focus on the manual inspection and post-processing following automatic translation. Moreover, we will describe how to use the test set for evaluation.

3.1 The Google Analogies Dataset

As described in Section 2.3.1, the task of analogical reasoning is popularly used for evaluating distributional semantic models. Benchmark data sets typically consist of lists of two word pairs that share a relation, such as *boy girl brother sister* or *easy easier big bigger*. The models are to correctly infer the fourth word based on the other three, and performance is measured in terms of accuracy of correctly answered questions.

One such analogy data set for English is the Google Analogies Dataset proposed by Mikolov, Chen et al. (2013). This data set contains a total of 19,544 analogy questions divided into semantic and syntactic subsets. Overall, there are 8,869 semantic questions covering five types of semantic relationships and 10,675 syntactic questions covering nine types of syntactic relationships. The semantic analogies are typically about places, like *Athens Greece Baghdad Iraq*, and the syntactic analogies are generally about verb tenses or forms of adjectives, such as *dancing danced decreasing decreased*. Number of questions and examples of word pairs within each relation type are given in Table 3.1.

Mikolov, Chen et al. (2013) created the questions in each relation type in two steps. First, they manually created a list of similar

	Relation type	# Questions	Word pair 1		Word pair 2	
Semantic	Common capital city	506	Athens	Greece	Baghdad	Iraq
	All capital cities	4,524	Abuja	Nigeria	Accra	Ghana
	Currency	866	Algeria	dinar	Angola	kwanza
	City-in-state	2,467	Chicago	Illinois	Houston	Texas
	Man–woman	506	boy	girl	brother	sister
Syntactic	Adjective-to-adverb	992	amazing	amazingly	apparent	apparently
	Opposite	812	acceptable	unacceptable	aware	unaware
	Comparative	1,332	bad	worse	big	bigger
	Superlative	1,122	bad	worst	big	biggest
	Present participle	1,056	code	coding	dance	dancing
	Nationality adjective	1,599	Albania	Albanian	Argentina	Argentinean
	Past tense	1,560	dancing	danced	decreasing	decreased
	Plural nouns	1,332	banana	bananas	bird	birds
	Plural verbs	870	decrease	decreases	describe	describes

Table 3.1: Number of questions and examples of word pairs within each relation type in the Google Analogies Dataset.

word pairs, such as *bad worse* and *big bigger*. Then, they formed a large list of questions by connecting two word pairs, for example *bad worse big bigger*. Questions within the semantic and syntactic subsets are separated by an initial line identifying the relation type. For example, *: family* identifies the man–woman analogies and *: gram2-opposite* identifies the second type of syntactic relation, i.e., the antonym analogies. However, the distinction between semantic and syntactic analogy questions might be unclear. For instance, the questions including opposites and nationality adjectives can arguably be considered semantic.

3.2 Translation to Norwegian

Benchmark analogy data sets are not currently available for Norwegian. However, we can use existing resources for English and adapt them to Norwegian. As for the Google Analogies Dataset, we started this process by applying automatic machine translation, using Google Translate, which is a free machine translation service online. The Google Analogies Dataset is simply lexical, i.e., it does not provide any context for the words, and the automatic machine translation gave rather bad results. Thus, the translation had to be followed by manual post-editing. We could have added context for each analogy, e.g., *Athens is the capital of Greece and Baghdad is the capital of Iraq*, which could possibly have improved the quality of the translation. However, the combination of time it takes adding context and the inevitable manual post-editing, does not necessarily make this a more time-saving approach.

We translated the analogy data set in two iterations. First, we translated the questions within each relation type at a time. Google Translate has a limit of 5,000 characters per translation, so we further

divided the translation into blocks of questions with the same first word pair, for example all those starting with *Athens Greece* in the *Common capital city* relation type. In this step, we retained all translated questions, regardless of whether or not they were correctly translated or later to be removed. This is mainly because of the practicality of keeping track of corresponding source and target lines during post-processing of the translation. Second, we identified wrongly translated questions. Obvious errors, such as words not translatable into a Norwegian word, were easily identified. Other errors, such as words translating into the wrong tense, were cross-checked and corrected using a Norwegian dictionary, i.e., *Bokmålsordboka*, which is developed by the University of Bergen in collaboration with The Language Council of Norway.

The first relation type, *Common capital city*, translated well. The reason for this is that there is almost no ambiguity in country names and capitals. The translation required only minor editing, such as editing *Havana* to *Havanna* and *Tehran* to *Teheran*. For some reason, no instance of *France* was translated and it needed to be added manually. The *All capital cities* analogies also translated well. Most countries and capitals were translated correctly, but capitals containing *sh*, such as *Ashgabat* or *Dushanbe*, were consistently not translated with *sj*. The correct translations would be *Asjkhabad* and *Dusjanbe*. The translation of *France* needed to be added also for this relation type. In addition, common grammatical differences between English and Norwegian, for instance the difference in the use of *th* and *t*, respectively, as in *Kathmandu* and *Katmandu*, and *k* and *c*, as in *Madagascar* and *Madagaskar*, needed to be manually edited.

The currency analogy questions were almost perfectly translated, except for the word pair *Korea won*, which directly translated to *Korea vant*. Here, *vant* refers to the past tense of *win* and not the Korean currency. This example illustrates one of the weaknesses of automatic machine translation, i.e., the translation system is not provided with context. Although the translation of *won* to *vant* is a valid translation option, it is unsuitable in the context of our currency analogy questions.

The automatic machine translation made some mistakes in the *Plural nouns* relation type. For instance, it translated the plural nouns *mangoes* and *pineapples* to the singular nouns *mango* and *ananas*, respectively. Furthermore, translations of the singular form of *car* and *dream* were for some reason missing and needed to be added manually. In addition, it did not distinguish singular and plural form of *onion*, and both were translated to *løk*. Most people would say one *løk* and two *løk*, but according to the Norwegian dictionary, the correct plural form of *løk* is *løker*.

While nationality adjectives have capital letter in English, they do not in Norwegian. Google Translate almost consistently retained the capital letter when translating into Norwegian, and these adjectives

needed to be manually lower-cased. Every country was translated correctly, except from *China* that remained the same and *Sweden* that needed to be manually inserted. Furthermore, the automatic machine translation was inconsistent in the form of the adjectives. While some were correctly translated into singular forms, like *Bulgarian* to *bulgarsk*, others were incorrectly translated into plural forms, like *German* to *tyske*. We consistently edited the adjectives to having singular form. Moreover, the number of incorrectly translated adjectives were high. In fact, 14 of the 38 nationalities remained unchanged after translation, for instance *Australian*, *Cambodian*, *Irish* and *Swiss*.

The *Past tense* relation type includes analogy questions in which the automatic machine translation probably performed the worst. As a consequence, the post-editing of this relation type was by far the most time-consuming. We chose to translate the words into pairs of present and past tense, resulting in the English *-ing* verbs possibly taking the form of present participle in Norwegian. Thus, the translation required major post-editing and we needed to modify every word pair except the ones translating *listening listened* and *taking took*. We observe four repeating errors. First, the word pairs were either translated into the wrong verb tense, such as *reading read* to *lese lese*, which is simply two infinitive forms of *to read*. Second, the word pairs were translated into the wrong word class, such as *writing wrote* was translated to *skriftlig skrev*, where *skriftlig* refers to the adjective *written*. Third, the words were translated into non-corresponding pairs, i.e., pairs that are not forms of the same lexeme. For example, *decreasing decreased* was translated to *avtagende redusert*, where the two words are synonyms and *avtagende* is the present participle of *to decrease* and *redusert* the past tense. Similarly, *striking struck* was translated to *slående rammet*, where *slående* and *rammet* are synonyms for *to strike*, also taking the form of present participle and past tense. Fourth, there were word pairs where only one word is translated, either correctly or incorrectly, into Norwegian, such as *hiding hid* translated to *gjemmer hid* or *shrinking shrank* translated to *krymper shrank*. All these errors were manually corrected.

3.3 Post-processing

The remaining analogy questions required additional post-processing beyond correcting translational errors. We can categorize the questions according to the cause of the errors, e.g., linguistic differences, further divided into differences in morphology and semantics, and extralinguistic differences between English and Norwegian. We will detail these differences in the following subsections. The number of questions and examples of word pairs within each relation type in the

	Relation type	# Questions	Word pair 1		Word pair 2	
Semantic	Common capital city	506	Athen	Hellas	Bagdad	Irak
	All capital cities	4,524	Abuja	Nigeria	Accra	Ghana
	Currency	866	Algerie	dinar	Angola	kwanza
	City-in-county	2,542	Hønefoss	Buskerud	Stord	Hordaland
	Man–woman	506	gutt	jente	bror	søster
Syntactic	Adjective-to-adverb	992	munter	muntert	hel	helt
	Opposite	600	akseptabelt	uakseptabelt	vitende	uvitende
	Comparative	1,190	dårlig	dårligere	stor	større
	Superlative	930	dårlig	dårligst	stor	størst
	Nationality adjective	1,599	Albania	albansk	Argentina	argentinsk
	Past tense	1,560	danser	danset	avtar	avtok
	Plural nouns	1,122	banan	bananer	fugl	fugler
	Present tense	870	avta	avtar	beskrive	beskriver

Table 3.2: Number of questions and examples of word pairs within each relation type in the Norwegian Analogy Test Set.

resulting Norwegian Analogy Test Set, following post-processing of the Google Analogies Dataset, are shown in Table 3.2.

3.3.1 Linguistic differences

Morphology

Linguistics is concerned with the structure of language. Morphology is the study of the formation of words and involves for example the inflection of verbs. As languages can differ morphologically, the automatic machine translation may become problematic. We found some examples of such problematic differences when translating from English to Norwegian.

Initially, we discarded the last syntactic relation type, *Plural verbs*, as there is no similar morphological distinction in Norwegian. The analogy questions would result in pairs of identical words, for example the singular–plural verb pairs *decrease decreases describe describes* would translate to *avtar avtar beskriver beskriver*, and the relation type is unlikely to be inferred. We replaced this relation type with *Present tense* questions using the same verbs. For example, the analogy question *decrease decreases describe describes* was replaced by *avta avtar beskrive beskriver*, where *avta* and *beskrive* are the infinitive forms and *avtar* and *beskriver* are the present tenses of *to decrease* and *to describe*, respectively.

Furthermore, the *Present participle* relation type is not included in the Norwegian data set. In English, the present participle is a participle that ends with the affix *-ing*. It is commonly used with the auxiliary verb *to be* to form the continuous tense, e.g., 'I am *working*' or 'he was *singing*'. Even irregular verbs have an *-ing* form, e.g., *beating*, and virtually all English words with this affix are present participles. In Norwegian, the use of present participle is relatively uncommon, other

than after verbs of movement or position, such as 'han løp *gråtende* ut av rommet', meaning 'he ran *crying* out of the room'. More common is the use of present participle as an attribute to a noun, e.g., 'et *gråtende* barn', meaning 'a *crying* child', but a translation like this will not be comparable with the original word pairs.

In Norwegian, there is no dedicated adverb ending and various adjectives are identical in form of the corresponding adverb, a phenomenon referred to as *syncretism*. Examples of such adjective–adverb pairs are *amazing amazingly* translating to *utrolig utrolig*, *most mostly* translating to *mest mest* and *serious seriously* translating to *alvorlig alvorlig*. Identical word pairs are not very discriminative for inferring the adjective–adverb relation, and they are even ambiguous. Therefore, all such word pair instances were removed from the Norwegian data set. Another example of syncretism is nouns that do not change form from singular to plural in Norwegian, e.g., *child children* equals *barn barn*, *mouse mice* equals *mus mus* and *dollar dollars* equals *dollar dollar*. These plural noun pairs were also removed from the data set.

Moreover, in the *Adjective-to-adverb* relation type, some word pairs consisted of correctly translated adverbs but the associated adjectives were not corresponding. Two examples are *obvious obviously* which translated to *opplagt åpenbart* and edited to *åpenbar åpenbart*, and *complete completely* which translated to *komplett helt* and edited to *hel helt*. In addition, there is no distinction in the meaning of *fortunate fortunately* and *lucky luckily*, both translated to *heldig heldigvis* in Norwegian. The latter word pair instances were removed from the data set. After modifying this relation type, the number of analogy questions within it were almost halved. Therefore, we added fifteen adjective and adverb pairs that are not identical in form, for example *nødvendig nødvendigvis* 'necessary necessarily', *normal normalt* 'normal normally', and *tilfeldig tilfeldigvis* 'random randomly'. These additional word pairs were connected to every other existing word pairs.

Semantics

Semantics is the study of the meaning of linguistic expressions. Some words are not translatable between languages, not because we do not know the meaning of the expression, but because there exists no equivalent single word conveying the same meaning in the target language. We registered several examples of such words while translating from English to Norwegian.

For instance, in Norwegian there is no distinction between female and male grandchildren as there is in English, e.g., *granddaughter* and *grandson* in the *Man–woman* relation type are both translated to *barnebarn*, meaning simply *grandchild*. The gender relation cannot be

inferred from the word pair *barnebarn barnebarn* and we thus removed these identical word pairs from the Norwegian data set. In addition, in English the parents of a person's father or mother can be named *grandfather* and *grandmother* or *grandpa* and *grandma*. In Norwegian there exists only one term for each grandparent, if we are not to express the gender of the parent, e.g., *bestefar* and *bestemor*, respectively. As a result, the distinct word pairs in English became duplicates in Norwegian and we randomly chose to discard the word pair translating from *grandpa grandma*. In Norwegian, we can distinguish between the grandparents of your father, i.e., *farfar farmor*, meaning the father of your father and the mother of your father, and the grandparents of your mother, i.e., *morfar mormor*, meaning the father of your mother and the mother of your mother. We added these two word pairs by connecting them to all other word pairs within this relation type.

Google Translate consistently translated the superlative adjectives in definite form, e.g., *biggest* was translated to *største* which literally means *the biggest*. We chose to remove the trailing *-e*s to keep the adjectives indefinite. Furthermore, in English the words *big* and *large* express the same meaning, similarly do *high* and *tall*. In Norwegian we only have one suitable word for each expression, e.g., *stor* and *høy*, respectively. Hence, the duplicate word pairs were removed, randomly chosen to be the lines including *large largest* and *tall tallest*. Also, the lines including *tasty tastiest* were discarded because the superlative of *smakfull* or *velsmakende*, which *tasty* could translate into, can not be expressed as a single word, i.e., the adverb *mest* 'most' must be added in front. The same duplicate word pairs needed to be removed from the *Comparative* relation type, e.g., word pairs translating from *large larger* and *tall taller*. In addition, in order to avoid a third duplicated instance, the translation of *great greater* was modified from *stor større*, as suggested by Google Translate, into *flott flottere*, as the ambiguity of *great* results in several plausible translation options.

Similarly, for the opposite analogy questions, we found examples of words untranslatable to a single Norwegian word, in particular the negative statements. In English, we can make negative statements by adding negative prefixes, such as *un-*, *in-* and *dis-*, to nouns, adjectives and verbs. In Norwegian, we can add negative prefixes such as *u-*, *irr-* and *in-*. However, these prefixes can not be added to all words in this specific selection of words. For example, *konkurranseudyktig* literally means *uncompetitive*, but it is not a natural phrase. Google Translate did not manage to translate *unimpressive* or *uninformative* at all, because there are no corresponding single words in Norwegian. A direct translation would result in *uimponerende* and *uinformativ*, but these terms are not present in a Norwegian dictionary. We would rather express such words as two separate words, with a leading *not* to negate the affirmative word. But, multi-word entities are not a part of the data set, and such instances were removed.

	Relation type	English	Norwegian
Semantic	Common capital city	506	506
	All capital cities	4,524	4,524
	Currency	866	866
	City-in-state	2,467	-
	City-in-county	-	2,542
	Man–woman	506	506
Syntactic	Adjective-to-adverb	992	992
	Opposite	812	600
	Comparative	1,332	1,190
	Superlative	1,122	930
	Present participle	1,056	-
	Nationality adjective	1,599	1,599
	Past tense	1,560	1,560
	Plural nouns	1,332	1,122
	Plural verbs	870	-
	Present tense	-	870

Table 3.3: Total number of questions within each relation type in the Google Analogies Dataset and the Norwegian Analogy Test Set.

3.3.2 Extralinguistic differences

Extralinguistic differences go beyond the realm of language or linguistics, for example cultural variation. But, such differences may also have an impact on translation. The automatic machine translation of the *City-in-state* questions, e.g., *Chicago Illinois Houston Texas* was flawless and needed no post-editing. There are no differences between American cities and states in English and Norwegian, except for states including *North* or *South*, but such phrases were not present in the data set. Nevertheless, collocations of American cities and states do not apply to Norwegian in a similar way as for English due to cultural differences. Hence, we did not include the city–state pairs in the Norwegian data set. Instead, we made similar analogies consisting of Norwegian cities and counties, e.g., *Hønefoss Buskerud Stord Hordaland*, where *Hønefoss* and *Stord* are cities and *Buskerud* and *Hordaland* are counties. Like Mikolov, Chen et al. (2013), we made a list of 68 Norwegian cities, randomly chosen from Wikipedia's list of Norway's largest cities, and formed about 2,5K questions by connecting every city–county pair to 38 randomly selected other pairs.

Table 3.3 gives the number of questions within each relation type in the original Google Analogies Dataset and the Norwegian Analogy Test Set following post-processing. Table 3.4 presents the total number of relation types and analogy questions within each semantic and syntactic subset in the English and Norwegian data sets.

	English		Norwegian	
Subset	# Relation type	# Questions	# Relation types	# Questions
Semantic	5	8,869	5	8,944
Syntactic	9	10,675	8	8,863
Total	14	19,544	13	17,807

Table 3.4: Number of relation types and questions within the semantic and syntactic subsets of the Google Analogies Dataset and the Norwegian Analogy Test Set.

3.4 Evaluation

3.4.1 Accuracy

To measure the quality of various distributional semantic models, we can compute accuracy on the Norwegian Analogy Test Set utilizing the standard `accuracy()` method already available in gensim[1]. The different parameter options of this method is given in the list below.

- `questions` is a filename where the lines are analogy questions, split into types of relationship by *: section-name* lines.
- Questions containing words not present in the first `restrict_vocab` words are ignored. 30,000 by default.
- `most_similar` is a function that finds the top n most similar words.
- `case_insensitive` uppercases all words in analogy questions and vocabulary before evaluation. True by default.

The method reports accuracy for each relation type separately, in addition to the total accuracy for all relation types. Accuracy is defined as the number of correctly answered questions in a relation type over the total number of considered questions in that relation type. A question is to be considered if all of its words are present in the first `restrict_vocab` words.

More formally, let N_C denote the number of correctly answered questions in a relation type and N_T denote the total number of considered questions in that relation type. The corresponding definition of accuracy is shown in Equation (3.1).

$$accuracy = \frac{N_C}{N_T} \tag{3.1}$$

[1] https://radimrehurek.com/gensim

Chapter 4

Creating the Norwegian Synonymy Test Set

In this chapter, we will present the process of creating the Norwegian Synonymy Test Set based on the digital version of the Norwegian synonym dictionary *Norske synonymer blå ordbok*. We will detail the extraction of words and synonyms, as well as spelling variants. Also, we will describe the metrics we have implemented for evaluation.

4.1 A Norwegian synonym dictionary

Another task of distributional semantic models is to automatically determine semantically similar words, or synonyms, such as *cup* and *mug* or *taxi* and *cab*. Automatically acquiring synonyms from corpora, i.e., synonym extraction, is a challenging task. However, various natural language processing and information retrieval applications can benefit from knowledge about synonyms. The simplest evaluation measure of synonymy detection is comparison with manually created synonym dictionaries (Wu and Zhou, 2003). In this regard, we will use *Norske synonymer blå ordbok*, which is a Norwegian synonym dictionary manually created by Dag Gundersen and published by Kunnskapsforlaget.

In the initial stages of this project, we organized a meeting with Kunnskapsforlaget, which is a Norwegian publishing company. The meeting resulted in a collaboration in which they gave us access to a digital version of the synonym dictionary. Furthermore, they permitted us to use and retrieve information from it and distribute the result for academic research purposes. This has several advantages. First, we do not need to use annotators to create a resource from scratch. Instead, we take advantage of an already existing resource with large coverage and many words. Second, the resource is created by professional

lexicographers. Hence, it is therefore reliable and of higher quality than a resource created by, for example, crowdsourcing.

The dictionary's fourth book edition, published in 2013, consists of approximately a total of 28,000 headwords and 130,000 reference words. A headword begins a lexical entry in the dictionary, e.g., *abandonere* 'abandon', *ablegøyer* 'prank' and *abrupt* 'abrupt' are all headwords, as shown in the excerpt of the dictionary's first page in Figure 4.1. Each headword entry is associated with one or more lexical items, such as synonyms or reference words. Reference words are synonyms that refer to other lexical entries associated with additional synonyms, whereas synonyms do not refer further. For example, for the headword *abrupt* 'abrupt', the non-italic *avbrutt* 'disconnected', *stakkato* 'staccato' and *usammenhengende* 'discontinuous' are synonyms. The italic *plutselig* 'sudden' and *tverr* 'unwilling', prefixed by *se* 'see', are reference words.

We propose to use the digital resource of the dictionary to produce a list of words in which one can look up a particular word and retrieve a list of its synonyms. In the context of this project, i.e., for the evaluation of distributional semantic models, the synonyms in this list may be of different word senses and we do not separate them. For instance, the headword *aksent* 'accent' includes synonyms meaning a characteristic pronunciation, e.g., *intonasjon* 'intonation' and *tonefall* 'tone', as well as synonyms meaning a mark used in writing, e.g., *aksenttegn* 'accent mark' and *tegn* 'mark'. Imagine these synonyms are the only ones of *aksent* 'accent', the resulting entry would be:

```
"aksent": ["intonasjon", "tonefall",
"aksenttegn", "tegn"]
```

Translation: `"accent": ["intonation", "tone", "accent mark", "mark"]`

Additionally, we produce a version of the word list in which synonyms are grouped by word senses. Such a resource may be used for word sense induction, i.e., the task of automatically acquiring the sense of a target word, in which preserving information about word meaning is beneficial. The resulting entry would then be:

```
"aksent": [["intonasjon", "tonefall"],
["aksenttegn", "tegn"]]
```

Our primary work will be on the former word list, but both resources will be described in the following section.

4.2 XML parsing

The digital resource of the Norwegian synonym dictionary *Norske synonymer blå ordbok* takes the form of an XML document containing

1 addere

a

a

A
fra A til Å se *hel.*.
à for hver, per stk., til, til...stykket; eller.
A4 standard, ensrettet.
ab av, fra.
abakteriell se *ren*.
abalienere se *overdra*.
abandonere avstå, se *oppgi*.
abbed gardian, igumen, klosterforstander, mandritt, prior.
abbreviatur se *forkortelse*.
abc fibel, lesebok, stavebok; elementærkunnskaper, grunnlag, grunnsetninger, grunntrekk.
abderitt molbo, se *dumrian*.
abdikasjon tronfrasigelse, se *avskjed*.
abdisere frasi seg tronen, gå (av), nedlegge septeret, tre tilbake. Jf. *fratre*.
abdomen se *mage*.
aber se *feil*. Jf. *innvending*, *lyte*.
aberrant avvikende.
aberrasjon se *avvik*.
aberrerende avvikende.
abiturient artianer.
abjurere avsverge.
ablegøyer ap, apespill, apestreker (apekatt-), fynter, leven, løyer, puss, spillopper. Jf. *moro*.
ablusjon renselse.
abnorm anormal, monstrøs, patologisk, pervers, unormal, se *overdreven*, *uregelmessig*, *usunn*, *vanskapt*. Jf. *sykelig*.
abnormitet anomali, avvik, monstrositet, se *lyte*, *unntak*.
abolisjon se *opphevelse*.
abominabel se *avskyelig*.
abominasjon se *avsky*.
abonnement se *bestilling*.
abonnent leser, prenumerant, subskribent, tinger; kunde.
abonnere se *bestille*.
abonnere på holde, tinge.
abort misfødsel; misfoster; abortus provocatus, fosterfordrivelse (-drap), svangerskapsavbrytelse.
abortere se *mislykkes*; avbryte (dataprogram).
abrakadabra se *tøys*.
abrasjon slitasje.
abrogasjon avskaffelse.
abrogere avskaffe.
abrupt avbrutt, stakkato, usammenhengende; se *plutselig*, *tverr*.
abscess se *byll*.
absentere seg se *fjerne* (fjerne seg), *forsvinne*, *gå*.
absolusjon avløsning, se *forlatelse*.
absolutisme se *enevelde*.
absolutt abstrakt, allmenngyldig, betingelsesløs, definitiv, endelig, evig, kategorisk, uunngåelig; autokratisk, despotisk, diktatorisk, *egenmektig*, eneveldig, selvstendig, uavhengig, uinnskrenket; (grammatisk:) uten bestemmelse; *aldeles*, *avgjort*, *bestemt*, diametralt, faktisk, fullendt, *fullstendig*, fullt og fast, *ganske*, helt, iallfall, i høy(este) grad, loddrett (løgn), *overhodet*, sikkert, slett (ikke), *strengt*, *ubetinget*, *ugjenkallelig*, uomtvistelig, uten sammenligning, utvilsomt, visst; *endelig*, for all del, for enhver pris, *nødvendig*, partout, per fas et nefas, plent, på liv og død, *påtrengende*, under alle omstendigheter, med vold og makt.
absolvere se *tilgi*.
absorbere se *oppta*, *tilegne seg*.
abstensjon se *avkall*.
abstinens se *avholdenhet*.
abstinent avholdende, se *edruelig*, *kysk*.
abstrahere se *sammenfatte*.
abstrakt se *absolutt*, *teoretisk*; nonfigurativ.
abstrus dunkel, dyp, *merkelig*, *uforståelig*.
absurd barokk, besynderlig, fantastisk, fornuft(s)stridig, forrykt, gal, grotesk, idiotisk, latterlig, løyerlig, meningsløs, paradoksal, selvmotsigende, sinnssyk, tåpelig, ufornuftig, ulogisk, urimelig, vill.
absurditet se *paradoks*, *vanvidd*.
abundans se *overflod*.
abundant se *overflødig*, *rikelig*.
abus(us) misbruk.
accessoirer se *tilbehør*.
accouchere forløse.
accoucheur fødselshjelper.
acquit se *kvittering*.
action handling.
ad se *angående*; *gjennom*, *om*; *(inn)til*.
ad acta
legge ad acta henlegge, legge bort.
adam se *mann*.
adamsdrakt
i adamsdrakt (i Adams drakt) se *naken*.
adaptere se *tilpasse*.
adaptivitet tilpasningsevne.
ad calendas Graecas se *aldri*.
addendum (pl. -da) se *etterskrift*, *tilføyelse*.
addere se *oppregne*, *oppsummere*.

Figure 4.1: First page of the Norwegian synonym dictionary *Norske synonymer blå ordbok*, containing headwords and synonyms.

XML element	Frequency
Synonymordbok	1
artikkel	28,432
Diskriminator	306
Gram	1,454
Henvisning	12,589
HenvisSynonym	31,910
henv-ord	44,737
hode	28,432
JfrHenvis	954
JfrSynonym	240
Kommentar	116
Kropp	28,168
oppslagsord	28,432
sort-order	53
Sub	1
SynonymGruppe	39,565
Synonym	85,000
UnderArtikkel	4,115
Uttrykk	4,115
variant	721

Table 4.1: Frequency of XML elements in the digital resource of *Norske synonymer blå ordbok*.

elements, tags and attributes of which we want to extract information. We use an XML parser included in the ElementTree library[1] for Python to extract this information, allowing us to store representations of the document in data structures such as Python dictionaries and lists. An overview of the XML elements and their frequencies is presented in Table 4.1 and brief descriptions of each element are given in the bulleted list below.

- `Synonymordbok` is the root element.
- `artikkel` encloses each headword and its synonyms, possibly including additional elements.
- `Diskriminator` indicates discipline, style or other clarifications.
- `Gram` denotes word class or other grammatical information.
- `Henvisning` encloses a `henv-ord` element. It is used interchangeably with `HenvisSynonym`.
- `HenvisSynonym` encloses a `henv-ord` element. It is used interchangeably with `Henvisning`, and generates the prefix *se*.
- `henv-ord` is a reference word. It encloses the headword referred to, where additional synonyms can be found.

[1] https://docs.python.org/3.5/library/xml.etree.elementtree.html

- `hode` encloses an `oppslagsord` element. Some also include `Gram`, `variant` or `Kommentar` elements.
- `JfrHenvis` encloses a `henv-ord` element that is related to the headword, but not a part of the synonym list, i.e., not within a `SynonymGruppe` tag. It generates the prefix *Jf*.
- `JfrSynonym` encloses a `henv-ord` element that refers to related words. It generates the prefix *Jf*.
- `Kommentar` encloses explanations, spelling variants, etc.
- `Kropp`, except for a `hode` element, the `artikkel` elements include a `Kropp` element with one or more `SynonymGruppe` elements, as well as one or more `UnderArtikkel` elements. `artikkel` elements without a `Kropp` element have one or more `UnderArtikkel` elements.
- `oppslagsord` is a headword. It is enclosed by an `artikkel` element and thus the number of `oppslagsord` and `artikkel` elements should be identical.
- `sort-order` specifies the alphabetisation of compound headwords.
- `Sub` is a chemical formula.
- `SynonymGruppe` encloses a block of synonym elements, grouped by word sense.
- `Synonym` encloses regular synonyms that do not refer to other elements.
- `UnderArtikkel` encloses `Uttrykk` elements that have own synonym groups related to them.
- `Uttrykk` is an expression with own synonym groups related to it.
- `variant` is a spelling variant of an `oppslagsord` or `Uttrykk` element.

However, not all these elements are relevant for our purpose. In fact, only `oppslagsord` (headword), `variant` (spelling variant), `Synonym` (synonym), `henv-ord` (reference word) and `SynonymGruppe` (synonym group) elements are extracted and further processed. The headwords of interest are associated with a `Kropp` (body) element containing one or more synonym groups, i.e., lists of synonyms grouped by word sense. The intuition is that every element within a synonym group is considered to be a synonym of the given headword. In contrast, reference words enclosed by a `JfrHenvis` tag are observed only outside synonym groups and tend to belong to different parts of speech than the headword, like the reference word *overbevise* 'convince' and the headword *overbevist* 'convinced'. Hence, such words are not included in the synonym lists. Furthermore, headwords can be associated with

```
<artikkel id="4826482">
  <hode>
    <oppslagsord>fiksjon</oppslagsord>
  </hode>
  <Kropp>
    <SynonymGruppe>
      <Synonym>innbilning</Synonym>
      <HenvisSynonym><henv-ord>blendverk</henv-ord><
          /HenvisSynonym>
      <HenvisSynonym><henv-ord>dagdrøm</henv-ord></
          HenvisSynonym>
    </SynonymGruppe>
    <SynonymGruppe>
      <Synonym>skjønnlitteratur</Synonym>
      <Synonym>romanlitteratur</Synonym>
    </SynonymGruppe>
  </Kropp>
</artikkel>
```

Listing 4.1: XML encoding of a headword and its synonyms.

one or more `UnderArtikkel` (sub article) elements. However, such elements are not observed within the headword's synonym groups. In addition, they include expressions, often consisting of multiple words, which have own synonym groups associated to them. Therefore, we do not consider these elements as synonyms either.

Within a synonym group we find two kinds of synonyms, i.e., "regular" synonyms and reference words. As already stated, reference words are synonyms that refer to other headwords where additional synonyms can be found, whereas regular synonyms do not refer further. The XML block encoding the headword *fiksjon* 'fiction' is shown in Listing 4.1. We observe that *innbilning* 'imagination', *skjønnlitteratur* 'fictional literature' and *romanlitteratur* 'literary genre of novels' are regular synonyms, whereas *blendverk* 'illusion' and *dagdrøm* 'daydream' are reference words. Looking up the latter will lead us to additional synonyms.

Initially, we considered expanding a headword's synonym list with the synonyms of its reference words. For some headwords, the expansion resulted in reasonable synonym lists, e.g., if the headword did not include a large number of reference words or a particular reference word did not include many synonyms. However, most synonym lists became excessive with synonyms too far apart in meaning from the headword and belonging to different parts of speech. Consequently, we do not expand synonym lists with synonyms of reference words and both kinds of synonyms are processed equally.

Homonym headwords, i.e., headwords having the same spelling but different meanings, are additionally associated with a `Gram` element, i.e., a part-of-speech (POS) tag. For example, the homonym headword *rett*, referring to a panel of judges or something properly positioned to mention two of the word meanings, occurs twice as a headword, disambiguated with a noun tag (s.) and an adjective tag (adj.), respectively. However, because there is no consistent use of POS tags, i.e., they are only used for homonym headwords, we do not benefit from this grammatical information and do not distinguish word meanings.

Parsing the XML document results in a Python dictionary, in which each entry is a pair of a headword and a list of its relevant synonyms. For the particular headword *fiksjon* 'fiction' from Listing 4.1, the dictionary entry is:

```
"fiksjon": ["skjønnlitteratur", "romanlitteratur",
"innbilning", "blendverk", "dagdrøm"]
```

Translation: `"fiction": ["fictional literature", "literary genre of novels", "imagination", "illusion", "daydream"]`

At this point, headwords and synonyms are added to the data structure regardless of whether they are later to be discarded, due to being multi-word expressions, e.g., the headword *motsette seg* 'oppose oneself', or expanded, due to having spellings variants, e.g., the synonym *frambringe (frem-))* 'give rise to' may be spelled with the prefix *fram-* or *frem-*. Thus, we need to process the dictionary further.

4.2.1 Spelling variants

Many Norwegian words have spellings variants, i.e., a different spelling or form of the same word. In the XML document, spelling variants are coded either explicitly by the use of `variant` tags or implicitly by the use of certain types of parentheses. We consider spelling variants to be synonyms, as they can mutually replace one another. Consequently, we must extend our dictionary by adding spelling variants. Both headwords and synonyms are observed with spelling variants, and we will separately discuss the expansion of them in the following paragraphs.

Headwords Headwords may be associated with one or more `variant` elements, referring to its alternate spellings. For example, the headword *deltaker* 'participant' may also be spelled *deltager*. Similarly, *dokke* 'doll' can be spelled *dukke* and *endelse* 'ending' can be spelled *ending*. When parsing the document, we initialize an additional list

of headwords that are associated with a `variant` element, paired with a list of its spelling variants:

```
"deltaker":  ["deltager"]

"dokke":  ["dukke"]

"endelse":  ["ending"]
```

Considering that spelling variants are not consequently coded, as shown in Listing 4.2 and Listing 4.3 only one of the headwords have a `variant` tag, and that synonymy is a symmetric relation, we symmetrize the resulting list. Furthermore, we will use this list to keep track of spelling variants and to aid us in extending the synonym dictionary with variants of headwords and synonyms.

The initial spelling variant list is unprocessed, i.e., headwords and spelling variants are added regardless of being multi-word expressions or including punctuations. Thus, it must be further processed. Headwords that additionally include parentheses, e.g., *unntak(else)* 'exception', *atskillig(e)* 'significantly' and *innenriks(k)* 'domestic', are processed so that the initial entries are replaced by their expansion. E.g., the entry *unntak(else)* is replaced by the two entries *unntak* and *unntakelse*, and their lists of spelling variants are updated accordingly:

```
"unntak":  ["unntakelse", "unntagelse"]

"unntakelse":  ["unntak", "unntagelse"]
```

In general, multi-word headwords or spelling variants are discarded, like *et cetera*, *all right* and *ta det kuli* 'take it easy'. In addition, words including hyphens at the start or the end are discarded, such as the prefix *øye-* and suffix *-ånd*. Spelling variants including parentheses are expanded, e.g., the spelling variant *jus(s)* 'jurisprudence' is replaced by *jus* and *juss*. Similarly, spelling variants including commas are expanded, e.g., *fremføring, fremførelse* 'performance' is replaced by *fremføring* and *fremførelse*.

For each headword present in the spelling variant list, its synonym list is extended with its variant words. For instance, the synonym list of the headword *flintskallet* 'bald-headed', encoded in Listing 4.2, is extended with the variant word *fleinskallet*, encoded in Listing 4.3. Moreover, we also add *flintskallet* to the synonym list of *fleinskallet*.

Furthermore, if two (or more) words are spelling variants of each other, their synonym lists need to be identical except for the respective spelling variants. Some headwords that are spelling variants of each other, such as *fighte* 'to fight' and *faite*, have synonym lists already identical. For those who have not, e.g., *flintskallet* 'bald-headed' and *fleinskallet*, we modify their resulting synonym lists to be the union of their initial synonym lists, i.e., the set of synonyms present in either lists or in both, including their spelling variants:

```
<artikkel id="4827778">
  <hode>
    <oppslagsord>flintskallet</oppslagsord>
    <variant>fleinskallet</variant><?Pub Caret -1?>
  </hode>
  <Kropp>
    <SynonymGruppe>
      <HenvisSynonym><henv-ord>bar</henv-ord></
          HenvisSynonym>
    </SynonymGruppe>
  </Kropp>
</artikkel>
```

Listing 4.2: Example of headword with spelling variant.

```
<artikkel id="4827520">
  <hode>
    <oppslagsord>fleinskallet</oppslagsord>
  /hode>
  <Kropp>
    <SynonymGruppe>
      <Synonym>skallet</Synonym>
      <HenvisSynonym><henv-ord>bar</henv-ord></
          HenvisSynonym>
    </SynonymGruppe>
  </Kropp>
</artikkel>
```

Listing 4.3: Example of headword that is another headword's spelling variant.

```
"flintskallet": ["fleinskallet", "skallet",
"bar"]

"fleinskallet": ["flintskallet", "skallet",
"bar"]
```

Translation: `"bald-headed": ["bald-headed", "bald", "bare"]`

Other spelling variants are not headwords themselves, e.g., the spelling variants of the headword *nærtakende* 'huffy', i.e., *nærtagende* and *nærtaken*, are not initially entries in the synonym dictionary. Similarly, the spelling variant of the headword *topp-punkt* 'vertex', i.e., *toppunkt*, is neither. Consequently, such words are added as new entries to the synonym dictionary, including the synonym list of the original headword, the headword itself and the possible other spelling variants:

```
"nærtagende": ["nærtakende", "nærtaken",
"følsom", "sår"]

"nærtaken": ["nærtakende", "nærtagende",
"følsom", "sår"]
```

Translation: `"huffy": ["huffy", "huffy", "sensitive", "tender"]`

```
"toppunkt": ["topp-punkt", "verteks"]
```

Translation: `"peak": ["peak", "vertex"]`

Finally, wherever a headword with spelling variants occurs in other headwords' synonym lists, these synonym lists are extended with the spelling variants. For example, the synonym list of the headword *bar* 'bare' includes the headword *flintskallet* 'bald-headed' and thus its spelling variant *fleinskallet* is ensured to be added.

Other headwords include parentheses, indicating spelling variants without the headword having `variant` tags, e.g., *albu(e)rom* 'elbow-room', *rampet(e)* 'mischievous' and *fas(c)inere* 'fascinate'. Such headwords must be expanded similarly as the headwords associated with `variant` tags, as described above. The headwords are replaced by adding the expansions as new entries to the synonym dictionary, e.g., *albu(e)rom* is replaced by *alburom* and *albuerom*, in which inherit the synonym list of the headword, and the expansions are added to each other's synonym lists. For the particular headword *albu(e)rom*, the expansion results in the entries:

```
"albuerom": ["alburom", "armslag",
"bevegelsesfrihet", "plass"]

"alburom": ["albuerom", "armslag",
"bevegelsesfrihet", "plass"]
```

Translation: `"elbow-room": ["elbow-room", "elbow-room", "freedom of movement", "space"]`

Furthermore, every occurrence of the original headword in other headwords' synonym lists is replaced by its expansions. E.g., in the synonym list of the headword *armslag* 'elbow-room', *albu(e)rom* 'elbow-room' is replaced by *alburom* and *albuerom*:

```
"armslag": ["alburom", "albuerom", "bevegelsesfrihet", "plass"]
```

Translation: `"elbow-room": ["elbow-room", "elbow-room", "freedom of movement", "space"]`

Other headwords include commas, also indicating spelling variants, e.g., *hjemme, heime* 'home' and *kokke, kokkelere, kokkerere* 'cook'. The synonym dictionary is updated similarly as for headwords including parentheses. For headwords including parentheses or commas, the spelling variant list is updated with the expanded spelling variants, e.g., with the entries:

```
"albuerom": ["alburom"]

"alburom": ["albuerom"]

"hjemme": ["heime"]

"heime": ["hjemme"]
```

Headwords including parentheses and commas, e.g., *slåpen, slåpet(e)* 'lanky' are also expanded similarly. Moreover, we strip *(-)* from headwords such as *hand(-)*, *(-)isme* and *post(-)*. Headwords including hyphens at the start or end, such as *kjempe-* and *-aktig*, are discarded. Similarly, multi-word headwords including parentheses, e.g., *rådføre seg (med)* 'confer with' and *bestrebe seg (på)* 'endeavour', are discarded.

Synonyms We also observe synonyms with spelling variants. Some synonyms include nested parentheses, e.g., *kolonial(vare(r))* 'groceries' and *(land(e))vinning* 'conquest'. Such synonyms are replaced and expanded with all spelling variants, e.g., *kolonial(vare(r))* is replaced and expanded with *kolonial*, *kolonialvare* and *kolonialvarer*. Synonyms including single parentheses are expanded similarly, e.g., *foto(grafi)* 'photography' resulting in *foto* and *fotografi*. Additionally, if the synonym includes a hyphen, e.g., *(av-)svekke* 'invalidate' and *topp(-punkt)* 'peak', the synonym is replaced and expanded with and without the content within the parentheses. For instance, the synonym *(av-)svekke* is expanded with *av-svekke* and *svekke*.

Synonym	Translation	Expansion
framstilling (frem-)	representation	framstilling, fremstilling
rettfram (-frem)	straightforward	rettfram, rettfrem
etterligning (-likn-)	imitation	etterligning, etterlikning
hamning (havn-)	paddock	hamning, havning
høytflyvende (-flygende)	high-flying	høytflyvende, høytflygende
middelmåtig (-mådig)	mediocre	middelmåtig, middelmådig
brobygger (bru-)	mediator	brobygger, brubygger
lausunge (løs-)	bastard	lausunge, løsunge
antakelig (-tag-)	probably	antakelig, antagelig
medfølelse (-kjensle)	compassion	medfølelse, medkjensle
overfladisk (-flatisk)	superficial	overfladisk, overflatisk
svartsjuke (-syke)	jealousy	svartsjuke, svartsyke
snarvei (-veg)	shortcut	snarvei, snarveg
sjølgod (selv-)	smug	sjølgod, selvgod
sjukant (syv-)	heptagon	sjukant, syvkant
utdanning (-else)	education	utdanning, utdannelse
frakoplet (-kobl-)	disconnected	frakoplet, frakoblet
klenodie (-ium)	treasure	klenodie, klenodium
atskillelse (ad-)	separation	atskillelse, adskillelse
avkopling (-kobling)	relaxation	avkopling, avkobling
respektfull (-fylt)	respectful	respektfull, respektfylt
mesterstykke (-verk)	masterpiece	mesterstykke, mesterverk
jamgod (jevn-)	equal	jamgod, jevngod

Table 4.2: Examples of synonyms containing frequent patterns of spelling variants, their English translation and expansion.

Some spelling variant patterns are more frequent in Norwegian than others. To mention a few, words prefixed by *fram-* are often also spelled with the prefix *frem-*, words including the infix *-tag-* may be spelled as *-tak-*, and words suffixed by *-else* are possibly spelled with the suffix *-ing*. Synonyms with such patterns are indicated by parentheses and hyphens, e.g., *framstilling (frem-)* 'representation' and*avtagende (-tak-)* 'decreasing', and must be further processed. We have manually identified various frequent spelling variant patterns. Examples of synonyms including such patterns, and their expansions, are presented in Table 4.2.

In contrast to headwords with spelling variants, we only expand synonyms locally, i.e., the synonym *foto(grafi)* 'photography' is replaced by *foto* and *fotografi* only where the synonym explicitly occurs with parentheses. The intuition is supported by the synonym *varm(blodig)*, which is expanded to *varm* 'warm' and *varmblodig* 'warm-blooded' in the synonym list of the headword *sensuell* 'sensual'. In the specific context of this headword, both *varm* and *varmblodig* are suitable synonyms:

```
"sensuell":  ["erotisk", "heit", "kjødelig",
"libidinøs", "lidenskapelig", "sanselig",
"seksuell", "temperamentsfull", "varm",
```

"varmblodig", "vellystig", "voluptuøs",
"yppig", "lysten", "het"]

Translation: "sensual": ["erotic",
"hot", "bodily", "libidinous", "passionate",
"physical", "sexual", "temperamental",
"warm", "warm-blooded", "lascivious",
"voluptuous", "lush", "lustful", "hot"]

However, *varm* is also a synonym of the headword *tropisk* 'tropical'. But, in the context of *tropisk*, we do not accept extending the synonym list with *varmblodig*. Consequently, none of the synonyms with spelling variants, in which are context-dependent, are added to the global spelling variant list.

4.2.2 Extracting synonyms

As already stated, synonyms are initially added to the dictionary without being processed, i.e., the synonym lists include multi-word expressions and words including punctuations. As the basic units of our semantic vectors will be single words, the synonym lists need to be further processed.

In general, multi-word synonyms are discarded. However, if the entries are of the form *gruble (over)* 'ponder over', *fight (fait)* 'fight' and *ligne (likne)* 'resemble', i.e., with a single word outside of the parentheses, the synonym entry is replaced by that word. In addition, if the word within the parentheses is a spelling variant of the added word, the spelling variant is also added to the synonym list. E.g., *fight (fait)* is replaced by *fight* and *fait* and *ligne (likne)* is replaced by *ligne* and *likne*. If there are multiple words within the parentheses, they are added similarly if they are all spelling variants, e.g., *lyve (ljuge, lyge)* 'lie'. Synonyms including hyphens at the start or end are discarded, e.g., *dupleks-* 'duplex'.

The resulting synonym dictionary is a list of headwords in which we can look up a particular word and retrieve a list of its synonyms. The synonym lists are simply flat lists, including every synonym of a headword, regardless of their word meaning. The frequency of headwords and synonyms in the resulting Norwegian Synonymy Test Set are shown in Table 4.3. A graph representation of the distribution of headwords with 0–10 synonyms is given in Figure 4.2. Additionally, we obtain a version of the synonym dictionary in which synonyms are grouped by word sense. This resource will be described in the next subsection.

	Frequency	Average
Headwords	27,601	-
Synonym tokens	111,111	-
Synonym types	32,303	-
Synonyms per headword	-	4.03

Table 4.3: Frequency of headwords and synonyms in the Norwegian Synonymy Test Set.

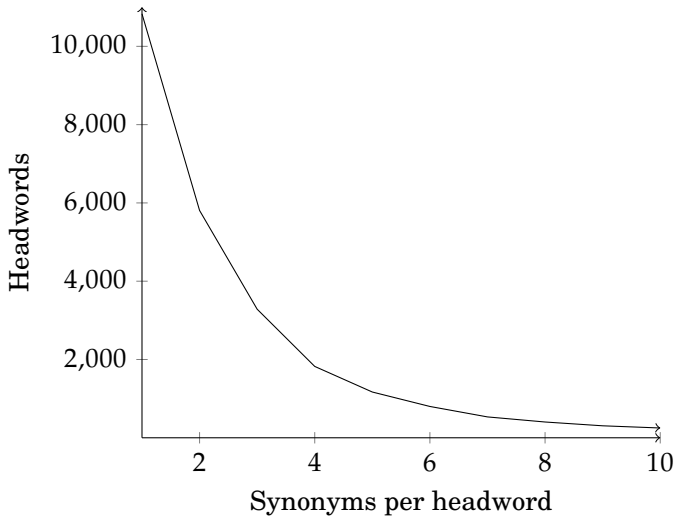

Figure 4.2: Distribution of number of synonyms per headword in the Norwegian Synonymy Test Set.

4.2.3 Synonym groups

Actually, the synonyms of the resource detailed in the previous subsection were initially grouped by word meaning, but the nested synonym lists were flattened out in the end. However, the nested version of the resource might be used for tasks such as word sense induction. But first, it needs further processing.

Originally, spelling variants of headwords are added as separate lists to the synonym list of the given headword. For the particular headword *pseudonym* which has the spelling variant *psevdonym*, the entry is:

```
"pseudonym": [["psevdonym"], ["alias",
"navneskjul"], ["skjulenavn"]]
```

Translation: `"pseudonym": [["pseudonym"], ["alias", "alias"], ["nickname"]]`

However, this spelling variant should not really be added as a separate synonym group. Although it is considered a synonym of the headword, it does not necessarily represent a distinct word meaning from those of the existing synonym groups. Alternatively, it can be added to all synonym groups. This is not quite correct either, as the spelling variant does not represent all word meanings. A final approach is to include additional information to distinguish spelling variants from synonyms:

```
"pseudonym": {variants: ["psevdonym"],
synonyms: [["alias", "navneskjul"],
["skjulenavn"]]}
```

In this way, we avoid incorrectly adding spelling variants to the synonym groups, but acquiring information about them or looking them up can be inconvenient. Nevertheless, we decide not to include spelling variants in the synonym lists. Instead, we keep them in a separate dictionary, similar to the one described in Section 4.2.1. For each headword with spelling variants, we only keep one of them as an entry in the synonym dictionary. Their synonym lists are identical, and by utilizing the spelling variant dictionary, we can avoid such redundant entries. The headword we retain is added as an entry to the spelling variant dictionary, paired with a list of its spelling variants. For example, the headword *pseudonym* is kept in the synonym dictionary, added as an entry to the spelling variant dictionary and paired with its spelling variant *psevdonym*:

```
"pseudonym": [["alias", "navneskjul"],
["skjulenavn"]]
```

```
"pseudonym": ["psevdonym"]
```

The spelling variant dictionary may be used as follows: if a particular target word, say *sammenlikning* 'comparison', is not an entry in

	Frequency	Average
Headwords	26,610	-
Synonym tokens	101,208	-
Synonym types	31,261	-
Synonyms per headword	-	3.80
Synonym groups per headword	-	1.27

Table 4.4: Frequency of headwords and synonyms in the synonym dictionary with synonym groups.

the synonym dictionary, we can look it up in the spelling variant dictionary. Then, we discover that *sammenlikning* is a spelling variant of *sammenligning*, in which is an entry in the synonym dictionary, and we retrieve its synonym list. Similarly, if our distributional model is to find a synonym of the headword *parallell* 'parallel', it may give us the word *sammenlikning*. We find that *parallell* is an entry in the synonym dictionary but *sammenlikning* is not present in its synonym list. Thus, we look it up in the spelling variant dictionary and register that our model did correctly identify a synonym of *parallell*. Moreover, all occurrences of spelling variants associated with a headword are removed from other headwords' synonym lists. Frequencies of headwords and synonyms in the resulting synonym dictionary with synonym groups are presented in Table 4.4.

4.3 Evaluation

4.3.1 Precision and recall

Commonly, precision and recall are used as evaluation metrics for classification and information retrieval tasks. In general, precision is the fraction of correct instances of the returned instances, whereas recall is the fraction of correct instances returned of the total number of correct instances. In a typical classification task one makes a prediction for every instance presented. However, in the task of synonym extraction we do not make a prediction for headwords we do not have embeddings for. Hence, we adapt the metrics in order to evaluate our word embedding models.

We define precision to be the number of headwords of which we predict a correct synonym among its k most similar words, over the number of headwords for which we have embeddings in addition to having embeddings for at least one of its synonyms. Furthermore, we define recall to be the number of headwords of which we predict a correct synonym among its `k` most similar words, over the total number of headwords in the synonym dictionary. In this way, precision and recall can ultimately be regarded as two variants of accuracy, e.g., one not

considering and one considering headwords we do not have embeddings for, respectively.

More formally, let H_S denote the number of headwords of which we predict a correct synonym, H_E denote the number of headwords for which we have embeddings in addition to having embeddings for at least one of its synonyms, and H_G denote the total number of headwords in the synonym dictionary. The resulting equations for precision and recall are then:

$$precision = \frac{H_S}{H_E} \tag{4.1}$$

$$recall = \frac{H_S}{H_G} \tag{4.2}$$

Chapter 5

Training word embeddings for Norwegian

In this chapter, we will present the various corpora used for training word embedding models for Norwegian, as well as the configuration of tools used for text pre-processing. Moreover, we will describe different word embedding frameworks and hyperparameters.

5.1 Corpora

Word embedding models require large amounts of training data in order to learn reliable word representations. In this project, we train word embeddings for Norwegian on a selection of large unannotated corpora, which represent different types of texts and domains. These corpora include the Norwegian Newspaper Corpus, the Norwegian Web as Corpus, and a digital text collection from the National Library of Norway (Nasjonalbiblioteket), hereinafter referred to as the NBDigital Corpus. These corpora will be further described in the following subsections.

5.1.1 The Norwegian Newspaper Corpus

The Norwegian Newspaper Corpus (NNC) represents modern Norwegian language in both its official written standards – Bokmål and Nynorsk. The text collection began in 1998 and NNC is a so-called monitor corpus, i.e., a dynamic corpus which grows in size over time. NNC is a compilation of texts collected from the web edition of 24 Norwegian newspapers, with a growth of approximately 230,000 words per day. The corpus contains over 1 billion Norwegian Bokmål words and 60 million Norwegian Nynorsk words, which makes it the largest corpus of Norwegian.

NNC can be obtained from Språkbanken's repository[1], which is a collection of Norwegian language technology resources at the National Library of Norway (Nasjonalbiblioteket). Pre-tokenized texts are available for articles dated 1998–2011. Articles dated 2012–2014 are available as separate XML documents. Raw text is extracted from these XML documents and tokenized by applying UDPipe (Straka, Hajič and Straková, 2016) v.1.1. The extraction and tokenization was performed by Løvold (2017) in the context of his MSc project.

The resulting corpus used in this project is a concatenation of four sub-corpora, e.g., the pre-tokenized Norwegian Bokmål texts from 1998–2011 and the extracted and tokenized Norwegian Bokmål texts from 2012, 2013 and 2014. Sentence, token and type counts for this corpus are presented in Table 5.1.

5.1.2 The Norwegian Web as Corpus

The Norwegian Web as Corpus (NoWaC) (Guevara, 2010), is a large web-based corpus of Norwegian Bokmål developed at the Department of Linguistics and Scandinavian Studies at the University of Oslo. The corpus was built by crawling, downloading and processing web documents in the period between November 2009 and January 2010.

The procedure used to collect the data in NoWaC is based on the methods described by Ferraresi et al. (2008). First, a list of URLs containing documents in Norwegian Bokmål was collected through queries to various search engines. Second, these documents, a total of 6,900, were used to start a crawling job, limited to the *.no* top-level Internet domain. Third, the downloaded documents were further processed, e.g., by applying tokenization, lemmatization and POS tagging, resulting in a linguistic corpus.

NoWaC v.1.0 can be obtained from the Department of Linguistics and Scandinavian Studies' website.[2] The corpus is distributed as a single file following the OBT format, which is the output format of the Oslo–Bergen Tagger (Johannessen et al., 2012). In this format, a sentence is enclosed by a start and end tag, with one token per line. For each sentence, we extract the tokens and concatenate them, separated by whitespaces, into a line, resulting in a tokenized and sentence segmented corpus. Core corpus statistics are shown in Table 5.1.

5.1.3 The NBDigital Corpus

The NBDigital Corpus (NBDigital), which is also available from Språkbanken's repository, comprises 21,483 digitalized books and

[1] https://www.nb.no/sprakbanken/show?serial=sbr-4&lang=en
[2] http://www.hf.uio.no/iln/om/organisasjon/tekstlab/prosjekter/nowac/

Corpus	# Sentences	# Tokens	# Types, full-forms	# Types, lemmas
NNC	87,515,520	1,527,414,377	9,016,282	7,886,045
NoWaC	33,501,699	687,209,465	6,097,166	5,507,706
NBDigital	45,714,400	813,922,111	20,756,201	19,855,097

Table 5.1: Counts of sentences, tokens and types for the various corpora, e.g., NNC, NoWaC and NBDigital.

other types of texts produced by 10,636 different authors and public institutions. The collection includes texts in several languages, such as French, Swedish and Norwegian Bokmål and Nynorsk.

NBDigital is distributed as a compressed .tar.gz file. The decompressed file includes 4,740 folders grouped by author, each of which contains all texts of the specific author. An additional 5,896 texts do not include author information. Hence, they are structured outside of the folders and are considered produced by distinct authors. Each text is a plain text file without any markup. The filename contains various metadata, e.g., a three-letter ISO code to represent the language of the text. We extract every text in Norwegian Bokmål, a total of 13,771 texts, and concatenate them for further pre-processing and use. Total counts of sentences, tokens and types are presented in Table 5.1.

Optical character recognition An important detail is that optical character recognition (OCR) has been applied to the texts. OCR is the automatic conversion of scanned images of different types of documents, such as handwritten or printed text, into machine-readable text. Consequently, the texts may be of varying quality and we want to quantify the impact of OCR errors on word embedding models.

In addition to a three-letter ISO code, each filename contains a decimal number representing the OCR software's confidence in the quality of the text. For example, the filename of one of Henrik Ibsen's texts contains an OCR confidence value of 875, to be read as 0.875. The intuition is, the higher the OCR confidence value, the better the quality of the text. We will examine the effects of OCR errors by conducting experiments on two different cut-offs, e.g., on texts with an OCR confidence value equal or greater than 0.85, a total of 9,513 texts, and on texts with an OCR confidence value equal or greater than 0.95, a total of 4,708 texts. Corpus statistics for the various OCR cut-offs are given in Table 5.2.

NBDigital includes texts from a wide period of time. From the initially extracted texts, the oldest text was published in 1736. Using an OCR confidence threshold of 0.85 and 0.95, the oldest texts were published in 1758 and 1820, respectively. The total distributions over the number of texts and year of publication for each OCR cut-off are presented in Figure 5.1. Seemingly, there is a correlation between the quality of a

OCR	# Texts	# Sentences	# Tokens	# Types, full-forms	# Types, lemmas
None	13,771	45,714,400	813,922,111	20,756,201	19,855,097
0.85	9,513	34,375,597	608,006,893	13,246,824	12,557,130
0.95	4,708	17,524,524	303,244,944	6,443,812	6,036,654

Table 5.2: Counts of texts, sentences, tokens and types for the various NBDigital OCR cut-offs.

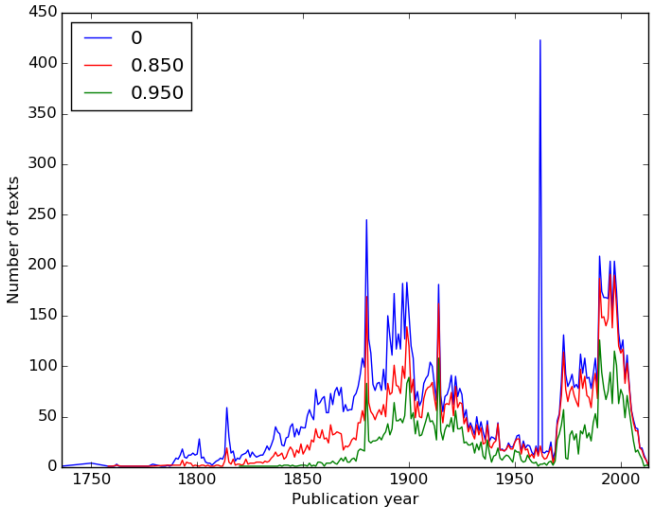

Figure 5.1: Distributions of year of publication and number of texts for the various NBDigital OCR cut-offs.

text and the year of publication. Recognizing text within an image of old handwritten material may be a challenging task due to variations in handwriting or poor inking. Hence, older texts may suggest a higher level of OCR errors. The experiments performed to quantify the effects of OCR errors on word embedding models will be further described in Section 6.3.1.

5.1.4 Combining the corpora

Training word embedding models from a larger corpus may in some cases yield better results. For this reason, we utilize two different combinations of the aforementioned corpora. Initially, we proposed to train word embedding models on the concatenation of all corpora only, e.g., NNC, NoWaC and NBDigital (with no OCR cut-off). However, the OCR errors in NBDigital may affect the quality of the combination of all corpora, as well as making it considerably more heterogeneous, as reflected in the type–token ratio in Table 5.1. Hence, we also train word embedding models on the concatenation of NNC and NoWaC only, limiting the possible effects of OCR errors. Total counts of

Corpus	# Sentences	# Tokens	# Types, full-forms	# Types, lemmas
NNC+NoWaC	121,017,219	2,214,623,842	12,757,597	11,345,192
All	166,731,619	3,028,545,953	31,631,169	29,511,825

Table 5.3: Counts of sentences, tokens and types for the different corpus concatenations.

sentences, tokens and types for each concatenation are presented in Table 5.3.

5.2 Pre-processing

In order to train word embedding models on the corpora described in the previous section, the texts need to be further pre-processed. Ideally, we would perform exactly the same pre-processing steps on each of the corpora. However, this is unattainable as pre-tokenized versions only are available for (parts of) NNC and NoWaC. Still, we strive to streamline the pre-processing as far as possible, i.e., we apply the same lemmatization.

We propose to evaluate the effects of text pre-processing on word embedding models using the created evaluation resources described in Chapter 3 and Chapter 4. For the task of synonym extraction, it is most meaningful to evaluate models trained on lemmas, as the synonym dictionary entry words are lemmas. Furthermore, for the task of analogical reasoning, it is only meaningful to evaluate full-forms on the syntactic analogies, as such analogies include inflections. Hence, we need two versions of each corpus – one of which is tokenized and one of which is lemmatized. The tools used for pre-processing are described in Section 5.2.1 and Section 5.2.2.

5.2.1 UDPipe

To perform tokenization and lemmatization we apply UDPipe (Straka, Hajič and Straková, 2016) v.1.2 with a pre-trained model for Norwegian Bokmål, both developed for the CoNLL 2017 shared task (Zeman et al., 2017). The pre-trained model was trained on the Universal Dependencies (UD) 2.0 versions of the Norwegian UD treebanks (Øvrelid and Hohle, 2016; Velldal, Øvrelid and Hohle, 2017), for participation in the CoNLL 2017 shared task (Straka and Straková, 2017).

UDPipe is a trainable pipeline for tokenization, morphological analysis, part-of-speech tagging, lemmatization and dependency parsing of CoNLL-U files, following the format defined in Universal Dependencies

ID	FORM	LEMMA	UPOSTAG	FEATS
1	Dagens	Dagens	PROPN	_
2	Næringsliv	Næringsliv	PROPN	_
3	tapte	tape	VERB	Mood=Ind | Tense=Past | VerbForm=Fin
4	en	en	DET	Gender=Masc | Number=Sing | PronType=Art
5	drøy	drøy	ADJ	Definite=Ind | Degree=Pos | Number=Sing
6	million	million	NOUN	Definite=Ind | Gender=Masc | Number=Sing
7	per	per	ADP	_
8	dag	dag	NOUN	Definite=Ind | Gender=Masc | Number=Sing
9	på	på	ADP	_
10	konflikten	konflikt	NOUN	Definite=Def | Gender=Masc | Number=Sing
11	.	$.	PUNCT	_

Table 5.4: Example of a sentence in CoNLL-U format. Fifth and last four columns are omitted.

version 2.[3] The software can be obtained from the UDPipe website[4], and is available as a binary for Linux/Windows/OS X, as a library for C++, Python, Perl, Java, C#, and as a web service.

The pipeline offers several running options for specifying the pre-processing of the input corpus. The input corpus may already be sentence segmented and tokenized, as is the case for NNC and NoWaC, or it can be a plain text, like NBDigital. We supply running options specifying to perform tokenization and lemmatization. Actually, the latter specifies to perform POS-tagging, but lemmatization is included in this option. Furthermore, we specify the output format and that the output is saved to a given file name. The CoNLL-U output files include three types of lines, e.g., word lines containing the annotation of a word in 10 tab-separated fields, blank lines separating sentences and comment lines prefixed with a hashtag. An example sentence is shown in Table 5.4. Descriptions of the relevant fields are given in Table 5.5.

To reduce the time of the pre-processing, we split each corpus into three equal parts and apply UDPipe on each part in parallel. For each corpus, and its associated CoNLL-U output files, furthermore for each sentence, we extract the tokens and concatenate them, separated by whitespaces, into a line. Similarly, we extract the lemmas. As a result of the pre-processing, we obtain two versions of each corpus, e.g., one of which is tokenized and one of which is lemmatized, e.g., a total set of 14 training corpora, including the NBDigital cut-offs.

5.2.2 The Abel computer cluster

Given the size of our data sets, the tasks of tokenization and lemmatization are computationally demanding, both in terms of time and memory usage. Initially, we configured the UDPipe pipeline

[3] http://universaldependencies.org/format.html
[4] http://ufal.mff.cuni.cz/udpipe

Field	Description
ID	Word index.
FORM	Form of word or punctuation mark.
LEMMA	Lemma or stem of word form.
UPOSTAG	Universal part-of-speech tag.
FEATS	List of morphological features.

Table 5.5: Description of relevant CoNLL-U format fields.

on a server hosted by the Language Technology Group (LTG) at the University of Oslo (UiO). We soon experienced that the experiments were too computationally expensive for this server and we needed to resolve to Abel[5], which is a high performance computing facility at UiO.

The Abel computing cluster is owned by UiO and the Norwegian Metacenter for High Performance Computing (NOTUR), and hosted by the Research Infrastructure Services Group at the University Center for Information Technology (USIT). Abel consists of more than 650 computers and 10,000 cores. The compute nodes have 64GiB memory and are connected to a common scratch disk space. We needed to apply for access to Abel, and by default home directories include 500GiB of disk space, which is sufficient for our work. Furthermore, a queue system ensures effective usage of the cluster.

A substantial amount of time and effort was devoted to setting up the Abel environment as well as learning the queue system. However, given the processing capacity of the cluster it will benefit the work on the subsequent tasks of this project, e.g., training and evaluating word embedding models.

5.2.3 Coverage

We compute coverage in terms of vocabulary overlap between the evaluation resources and the various corpora. The vocabulary overlaps between the analogy and synonym resources and each corpus are given in Table 5.6. The overlap is computed simply as the total number of types present in both the resource and corpus, over the total number of types in the resource.

We find that the vocabulary overlap is greater between the full-form corpora and the analogy resource, compared to the lemma corpora, as full-forms cover semantic as well as syntactic analogies. NNC, NNC combined with NoWaC, and the combination of all corpora equally obtain the highest coverage of 99.9% among the full-form corpora. In contrast, NBDigital has the slightest worst coverage of 97.4%. Moreover, the lemma corpora have greater vocabulary overlap with the

[5]http://www.uio.no/english/services/it/research/hpc/abel/

	Corpus	Resource	
		Analogy	Synonymy
Forms	NNC	99.9%	85.7%
	NBDigital	97.4%	90.2%
	NoWaC	99.7%	86.2%
	NNC+NoWaC	99.9%	90.1%
	All	99.9%	95.4%
Lemmas	NNC	98.7%	87.2%
	NBDigital	96.6%	90.9%
	NoWaC	98.6%	87.7%
	NNC+NoWaC	99.3%	91.2%
	All	99.7%	95.7%

Table 5.6: Vocabulary overlap between the evaluation data sets and the various corpora.

	OCR	Resource	
		Analogy	Synonymy
Forms	None	97.4%	90.2%
	0.85	97.3%	89.1%
	0.95	96.5%	85.7%
Lemmas	None	96.6%	90.9%
	0.85	96.2%	90.1%
	0.95	94.4%	87.1%

Table 5.7: Vocabulary overlap between the evaluation data sets and the various NBDigital OCR cut-offs.

synonym resource compared to the full-form corpora, as the dictionary entries of this resource are lemmas. The combination of all lemma corpora obtain the highest coverage of 95.7%. The coverage between corpora and the synonym resource spans a wider range than for the analogy resource. For instance, NNC has the worst coverage across the lemma corpora with a 87.2% vocabulary overlap, 8.5 percentage points worse than the best. We also report vocabulary overlap between the full-form corpora and the synonym resource, although these corpora are not evaluated using this resource.

Similarly, coverage of the various NBDigital cut-offs are given in Table 5.7. We find that increasing the OCR threshold does not seem to substantially decrease coverage. Coverage between NBDigital full-forms and the analogy resource decreases with only 0.9 percentage points when increasing the OCR threshold to 0.95. A slightly greater decrease in coverage is reported for NBDigital lemmas of the synonym resource.

5.3 Word embedding frameworks

There exist various frameworks for learning word representations. In particular, we will explore the word2vec (Mikolov, Chen et al., 2013), fastText (Bojanowski et al., 2016) and GloVe (Pennington, Socher and Manning, 2014) frameworks. The model architectures underlying these frameworks are further described in Section 2.2.4. Our intention is not to discover the perfect word embedding model, and hyperparameter optimization is thus out-of-scope of this project. Default values for the most relevant hyperparameter of each framework will be outlined in the following subsections.

5.3.1 Word2vec

A popular toolkit for computing word vectors is gensim (Řehůřek and Sojka, 2010), which is freely available from the gensim website.[6] Gensim is a Python reimplementation of the original word2vec toolkit, which provides implementations of the CBOW and Skip-gram model architectures.

Gensim's word2vec takes as its input a large corpus of text, actually a list of sentences, and accepts several hyperparameters, such as training algorithm, e.g., CBOW or Skip-gram, size of the context window and number of times to iterate over the corpus. We train both CBOW and Skip-gram models with default values for all hyperparameters. The most relevant hyperparameters are described in further detail below:

- The model is initialized and trained from an iterable object or list of `sentences`, where each sentence is a list of words.
- `sg` defines the model architecture, either 0 for CBOW or 1 for Skip-gram.
- `size` is the size of the feature vectors, e.g., the number of dimensions. Default value is 100.
- `window` is the size of the context window, e.g., the maximum distance between the target word and the predicted word. Default value is 5.
- Words with total frequency lower than `min_count` will be ignored. Default value is 5.
- `workers` is the number of working threads, which facilitates faster training with multicore machines. Default value is 3.
- `iter` is the number of iterations over the corpus. Default value is 5.

[6] https://radimrehurek.com/gensim

- By default, `sorted_vocab` defines to sort the vocabulary by descending frequency.

5.3.2 FastText

A second framework for computing word vectors is fastText, which is an open-source library released by Facebook AI Research.[7] FastText is essentially an extension of word2vec, and also provides the CBOW and Skip-gram model architectures. Additionally, fastText takes into account the internal structure of words.

FastText word embedding models can be trained by calling the fastText executable and specifying the training algorithm, either `cbow` or `skipgram`, and other hyperparameters. FastText takes as input a corpus of text, assumed to be an utf-8 encoded text file, and accepts hyperparameters such as size of the feature vectors, maximum length of character n-grams and number of iterations over the corpus. We train both fastText CBOW and Skip-gram models with default values for all hyperparameters. The most relevant ones are described in the following list:

- The model is initialized and trained from an `input` corpus text file.
- Words with total frequency lower than `minCount` will be ignored. Default value is 5.
- `wordNgrams` is the maximum length of word n-grams. Default value is 1.
- `minn` is the minimum length of character n-grams. Default value is 3.
- `maxn` is the maximum length of character n-grams. Default value is 6.
- `dim` is the size of the feature vectors, e.g., number of dimensions. Default value is 100.
- `ws` is the size of the context window, e.g., the maximum distance between the target word and the predicted word. Default value is 5.
- `epoch` is the number of iterations over the corpus. Default value is 5.
- `thread` is the number of worker threads. Default value is 12.

[7] https://github.com/facebookresearch/fastText

5.3.3 GloVe

A third framework for computing word vectors is GloVe.[8] This framework includes four main tools. `vocab_count` constructs a list of tokens and associated counts from a corpus. The corpus is assumed to be a text file consisting of whitespace-separated tokens. `cooccur` constructs word–word co-occurrence statistics from the corpus. Furthermore, `shuffle` shuffles these co-occurrence statistics and `glove` trains the GloVe model on the co-occurrence data with specified hyperparameters.

As default dimensionalty of GloVe vectors is 50, as opposed to 100 for word2vec and fastText, we also train word vectors of dimensionality 100. In this way, embeddings across the various frameworks are comparable. Apart from that, we employ default values for all hyperparameters. The most relevant hyperparameters are described in the list below:

- Words with total frequency lower than `min-count` will be ignored. Default value is 5.
- When `symmetric` is 1, left and right contexts are used. When 0, only the left context is used. Default value is 1.
- `window-size` is the size of the context window. Default value is 15.
- `vector-size` is the size of the feature vectors, e.g., number of dimensions. Default value is 50.
- `threads` is the number of worker threads. Default value is 8.
- `iter` is the number of iterations over the corpus. Default value is 25.

[8] https://github.com/stanfordnlp/GloVe

Chapter 6

Evaluation experiments

In this chapter, we will evaluate the performance of various word embedding models using the resources created in Chapter 3 and Chapter 4. We will attempt to isolate the effects of corpus, text preprocessing, word embedding framework and certain hyperparameters, but as our intention is not to discover the perfect word embedding model, we will not focus on hyperparameter optimization. Rather, we will demonstrate the usefulness of the evaluation resources for ranking the relative performance of different models.

The experiments and results for the tasks of analogical reasoning and synonym extraction will be presented in Section 6.1 and Section 6.2, respectively. In Section 6.3, we will present a few supplementary evaluation experiments.

6.1 Analogical reasoning

As discussed in Section 2.3.1, one intrinsic method for evaluating the quality of different word embedding models is to test their prediction of semantic and syntactic word similarities. We will utilize the Norwegian Analogy Test Set created in Chapter 3, and compute accuracy on the analogy questions following the scheme in Section 3.4. We will report evaluation results for the semantic and syntactic sections separately, in addition to the total. We will refer to accuracy on the semantic section as semantic accuracy and accuracy on the syntactic section as syntactic accuracy. Accuracy on all relation types are referred to as total accuracy. Moreover, we will report evaluation results for vocabularies restricted to the 30K and the 1M most frequent words. This restriction involves ignoring analogy questions including a word not present in the 30K or 1M most frequent words.

In Section 6.1.1, we will perform initial evaluation experiments on word2vec CBOW and Skip-gram embedding models. In Section 6.1.2,

we will compare word2vec to other word embedding frameworks, such as fastText and GloVe and in Section 6.1.3, we will discuss methodological concerns regarding the evaluation.

6.1.1 Word2vec embeddings and choice of corpora

The initial experiments will focus on the effects of training word embedding models on various types of corpora, in addition to the effects of different types of pre-processing of text. As we recall from Section 5.2, we hold one tokenized and one lemmatized version of each corpora, e.g., NNC, NBDigital and NoWaC. Similarly, we hold two versions of the combination of all corpora and of NNC combined with NoWaC. The tokenized corpora include words in their full forms, whereas the lemmatized corpora include dictionary forms of words, known as lemmas.

We limit the initial experiments to word2vec CBOW and Skip-gram embedding models. The methods underlying these models are described in more detail in Section 2.2.4. We train the models on all corpora, both full-forms and lemmas, and employ default values for all hyperparameters, as outlined in Section 5.3.1. The experimental setup across model architectures, corpora, pre-processing and vocabulary sizes results in a total of 40 different model configurations. We will individually discuss the experiments and results for word2vec full-form and lemma embeddings. First, we will discuss the restriction of vocabulary size.

Restricting the vocabulary size

Following the methodology of Mikolov, Chen et al. (2013), we evaluate word2vec CBOW and Skip-gram embeddings with vocabularies restricted to the 30K and 1M most frequent words. We find that restricting the vocabulary size affects both semantic, syntactic and total accuracy because it controls the number of analogy questions considered. That is, questions including a word not present in the 30K or 1M most frequent words are ignored.

Table 6.1 presents the number of considered questions for word2vec Skip-gram full-form embeddings trained on NNC combined with NoWaC with the vocabulary restricted to the 30K and 1M most frequent words. The results are broken down for the individual relation types within the semantic and syntactic sections in the analogy test set. The total number of questions of each relation type are given in the rightmost column. Similarly, Table 6.2 presents the number of considered questions for word2vec Skip-gram lemma embeddings trained on NNC. As syntactic analogies include inflections it is not meaningful to evaluate lemma embeddings on such analogies, and the syntactic section is omitted from this table.

	Relation type	30K vocab. # Considered	1M vocab. # Considered	# Total
Semantic	Common capital city	380	506	506
	All capital cities	903	4,446	4,524
	Currency	40	866	866
	City-in-county	2,310	2,542	2,542
	Man–woman	240	506	506
	Semantic total	3,873	8,866	8,944
Syntactic	Adjective-to-adverb	812	992	992
	Opposite	210	600	600
	Comparative	1,056	1,190	1,190
	Superlative	132	930	930
	Nationality adjective	631	1,597	1,599
	Past tense	992	1,482	1,560
	Plural nouns	462	1,056	1,122
	Present tense	552	870	870
	Syntactic total	4,847	8,717	8,863
	Total	8,720	17,583	17,807

Table 6.1: Number of considered questions and total number questions within each semantic and syntactic relation type in the Norwegian Analogy Test Set for word2vec Skip-gram full-form embeddings trained on NNC+NoWaC with the vocabulary restricted to the 30K and 1M most frequent words.

	Relation type	30K vocab. # Considered	1M vocab. # Considered	# Total
Semantic	Common capital city	420	506	506
	All capital cities	1,370	4,524	4,524
	Currency	40	808	866
	City-in-county	2,445	2,542	2,542
	Man–woman	306	506	506
	Total	4,581	8,886	8,944

Table 6.2: Number of considered questions and total number of questions within each semantic relation type in the Norwegian Analogy Test Set for word2vec Skip-gram lemma embeddings trained on NNC with the vocabulary restricted to the 30K and 1M most frequent words.

		30K vocab.			1M vocab.		
		% Acc.					
	Corpus	Sem.	Syn.	Total	Sem.	Syn.	Total
CBOW	NBDigital	14.8	49.3	34.9	8.6	32.1	22.0
	NNC	38.5	56.6	48.5	30.4	44.2	37.3
	NoWaC	31.2	57.9	46.7	20.8	43.2	32.1
	NNC+NoWaC	41.2	59.5	51.4	32.1	47.6	39.8
	All	37.1	59.2	49.3	30.2	47.6	39.0
SG	NBDigital	26.4	53.6	42.3	13.4	36.3	26.5
	NNC	**50.3**	61.6	56.5	**49.9**	52.2	**51.1**
	NoWaC	39.9	59.6	51.3	27.1	44.2	35.7
	NNC+NoWaC	49.7	**63.6**	**57.4**	43.0	**53.4**	48.2
	All	29.2	58.6	45.5	28.7	47.6	38.2

Table 6.3: Accuracy on the semantic and syntactic sections and total accuracy on all relation types in the Norwegian Analogy Test Set for word2vec CBOW and Skip-gram full-form embeddings with the vocabulary restricted to the 30K and 1M most frequent words.

We observe that increasing the vocabulary size substantially expands the number of analogy questions considered. For example, for the full-form embedding model, the number of questions considered in the *Currency* relation type increases from 40 to 866. Similarly, for the lemma embedding model, the total number of semantic questions considered increases from 4,581 to 8,886, almost covering all 8,944 semantic questions. Furthermore, word vectors tend to get less meaningful with decreasing frequency of the target word, and more questions can be assumed to be incorrectly answered.

However, it is not meaningful to compare word embedding models across vocabulary sizes, because restricting the vocabulary to the 1M most frequent words will – by definition – yield lower accuracy compared to the 30K most frequent words. Therefore, in the following, we will in parallel discuss the evaluation of word embedding models with different vocabulary sizes.

Word2vec full-form embeddings

This subsection presents evaluation results for word2vec CBOW and Skip-gram full-form embeddings. Full-form embeddings are trained on tokenized corpora containing words in their full forms as opposed to their dictionary forms or lemmas. Semantic, syntactic and total accuracy on the analogy test set for word2vec CBOW and Skip-gram full-form embeddings with the vocabulary restricted to the 30K and 1M most frequent words is presented in Table 6.3.

In general, Skip-gram full-form embeddings seem to recognize analogies

better compared to CBOW full-form embeddings. In fact, Skip-gram gives the highest semantic, syntactic and total accuracy across corpora and vocabulary sizes. When restricting the vocabulary to the 30K most frequent words, Skip-gram trained on NNC combined with NoWaC yields the best total accuracy of 57.4%. When expanding the vocabulary to the 1M most frequent words, Skip-gram trained on NNC yields the best total accuracy of 51.1%. Moreover, Skip-gram trained on NNC yields the best semantic accuracy for both vocabulary sizes, e.g., 50.3% with the 30K word vocabulary and 49.9% with the 1M word vocabulary. Similarly, Skip-gram trained on NNC combined with NoWaC yields the best syntactic accuracy of 63.6% and 53.4% with the 30K and 1M word vocabulary, respectively.

Furthermore, Skip-gram full-form embeddings consistently give higher accuracy across corpora and vocabulary sizes compared to CBOW. However, this is not the case when trained on the combination of all corpora, in which CBOW yields slightly better semantic, syntactic and total accuracy. Seemingly, CBOW might benefit more from additional data than Skip-gram does. Still, CBOW predicts semantic and syntactic analogies worse than Skip-gram when trained on the combination of NNC and NoWaC.

Generally, full-form embeddings trained on NNC combined with NoWaC seem to perform substantially better than those trained on NBDigital and slightly better than those trained on NNC. The reason for this might be that NNC is the largest individual corpus, assumingly of high quality since the texts are gathered from modern newspaper web editions, as opposed to the older texts in NBDigital which contain OCR errors. Moreover, combining NNC with NoWaC, which represents additional domains, increases the amount of training examples. As a result, the embeddings might learn better word representations. However, embeddings trained on NoWaC alone never yield the best accuracy, possibly because of the noisy nature of web documents.

It is clear that embeddings trained on NBDigital perform the worst, especially in terms of semantic accuracy. Actually, embeddings trained on NBDigital yield the worst semantic, syntactic and total accuracy across model architectures and vocabulary sizes. Adding NBDigital to the combination of NNC and NoWaC decreases accuracy, although combining NNC with only NoWaC increases it. As discussed in Section 5.1.3, the low accuracy might be due to OCR errors. Therefore, we want to quantify the impact of OCR errors on word embeddings and will present additional evaluation experiments in Section 6.3.1.

As we recall from Table 6.3, the best full-form embedding model is Skip-gram trained on NNC combined with NoWaC, in terms of total accuracy and with the vocabulary restricted to the 30K most frequent words. The evaluation results for this model are presented in further detail in Table 6.4. This table shows the number of correct questions, i.e., the number of questions in the analogy test set, each on the form A is

	Relation type	# Correct	# Considered	# Total	% Acc.
Semantic	Common capital city	249	380	506	65.5
	All capital cities	495	903	4,524	54.8
	Currency	10	40	866	25.0
	City-in-county	973	2,310	2,542	42.1
	Man–woman	196	240	506	81.7
	Semantic total	1,923	3,873	8,944	49.7
Syntactic	Adjective-to-adverb	271	812	992	33.4
	Opposite	26	210	600	12.4
	Comparative	773	1,056	1,190	73.2
	Superlative	77	132	930	58.3
	Nationality adjective	502	631	1,599	79.6
	Past tense	642	992	1,560	64.7
	Plural nouns	325	462	1,122	70.3
	Present tense	468	552	870	84.8
	Syntactic total	3,084	4,847	8,863	63.6
	Total	5,007	8,720	17,807	57.4

Table 6.4: Number of correct and considered questions, total number of questions and accuracy for each semantic and syntactic relation type in the Norwegian Analogy Test Set for the best full-form embedding model in terms of total accuracy, e.g., word2vec Skip-gram trained on NNC+NoWaC with the vocabulary restricted to the 30K most frequent words.

to *A** as *B* is to *B**, of which the model predicts the correct word *B**. Furthermore, the table shows the number of considered questions, i.e., the number of questions in the analogy test set of which all words are present in the restricted vocabulary. For comparison, the total number of questions in the analogy test set is also given. Finally, the table gives accuracy, defined as the number of correctly predicted questions over the total number of considered questions. The results are broken down for the individual relation types within the semantic and syntactic sections.

We observe from Table 6.4 that the model does considerably well in predicting semantic word similarities such as the gender relation, e.g., between *gutt* 'boy' and *jente* 'girl' or *bror* 'brother' and *søster* 'sister', yielding an 81.7% accuracy for the *Man–woman* relation type. In contrast, the model seems to struggle with syntactic word similarities such as the opposite relation, e.g., between *effektiv* 'effective' and *ineffektiv* 'ineffective' or *etisk* 'ethical' and *uetisk* 'unethical', yielding an accuracy of only 12.4% for the *Opposite* relation type.

Word2vec lemma embeddings

This subsection presents evaluation results for word2vec CBOW and Skip-gram lemma embeddings. Lemma embedding models are trained

	Corpus	30K vocab. % Sem. acc.	1M vocab. % Sem. acc.
CBOW	NBDigital	18.2	9.6
	NNC	46.0	39.1
	NoWaC	33.8	22.3
	NNC+NoWaC	49.4	41.0
	All	44.0	37.9
SG	NBDigital	29.2	14.7
	NNC	**56.3**	**53.2**
	NoWaC	46.2	30.6
	NNC+NoWaC	54.8	48.6
	All	33.1	31.0

Table 6.5: Accuracy on the semantic sections in the Norwegian Analogy Test Set for word2vec CBOW and Skip-gram lemma embeddings with the vocabulary restricted to the 30K and 1M most frequent words.

on lemmatized corpora containing dictionary forms of words, or lemmas. Semantic accuracy on the analogy test set for word2vec CBOW and Skip-gram lemma embeddings with the vocabulary restricted to the 30K and 1M most frequent words is presented in Table 6.5. As we recall, lemma embeddings are not evaluated on syntactic analogies, and the syntactic analogies are omitted from the table.

Similarly as observed for Skip-gram full-form embeddings, we find that Skip-gram lemma embeddings seem to recognize analogies better compared to CBOW lemma embeddings. More specifically, Skip-gram gives the highest semantic accuracy across corpora and vocabulary sizes. Skip-gram trained on NNC yields the best semantic accuracy of 56.3% when restricting the vocabulary to the 30K most frequent words, as well as the best semantic accuracy of 53.2% when expanding the vocabulary to the 1M most frequent words. This trend was also observed for the full-form embeddings in Table 6.3. However, in that case the semantic accuracy was slightly lower, 50.3% and 49.9%, respectively. Actually, lemma embeddings consistently give higher semantic accuracy compared to the corresponding full-form embeddings across all model configurations.

Moreover, Skip-gram lemma embeddings consistently give higher semantic accuracy across corpora compared to CBOW. But again, as found for the full-form embeddings, the exception is embeddings trained on the combination of all corpora, in which CBOW yields substantially better semantic accuracy. CBOW lemma embeddings trained on all corpora with the vocabulary restricted to the 30K most frequent words yield a semantic accuracy of 44.0%, which is over 10 percentage points higher than the semantic accuracy given by the corresponding Skip-gram embeddings. Still, Skip-gram yields the best results overall.

As for the full-form embeddings, the worst performing lemma embed-

	Relation type	# Correct	# Considered	# Total	% Acc.
Semantic	Common capital city	342	420	506	81.4
	All capital cities	1,043	1,370	4,524	76.1
	Currency	12	40	866	30.0
	City-in-county	953	2,445	2,542	39.0
	Man–woman	227	306	506	74.2
	Total	2,577	4,581	8,944	56.3

Table 6.6: Number of correct and considered questions, total number of questions and accuracy for each semantic relation type in the Norwegian Analogy Test Set for the best lemma embedding model in terms of total accuracy, e.g., word2vec Skip-gram trained on NNC with the vocabulary restricted to the 30K most frequent words.

dings, regardless of model architecture and vocabulary size, are trained on NBDigital. The lowest semantic accuracy of 9.9% is reported for CBOW trained on NBDigital with the vocabulary restricted to the 1M most frequent words. Furthermore, adding NBDigital to the combination of NNC and NoWaC decreases accuracy. For instance, Skip-gram trained on all corpora with the 30K word vocabulary yields a semantic accuracy of 33.1%, which is over 20 percentage points lower than the accuracy given by Skip-gram trained on the combination of NNC and NoWaC.

As we noticed from Table 6.5, the best lemma embedding model is Skip-gram trained on NNC, in terms of semantic accuracy and with the vocabulary restricted to the 30K most frequent words. The evaluation results for this model are presented in further detail in Table 6.6. This table shows the number of correct and considered and total number of questions within each semantic relation type as well as the accuracy. From these results, we find that the embeddings are considerably good at predicting semantic word similarities within the *Common capital city* relation type, e.g., between *Athen* 'Athens' and *Hellas* 'Greece' or between *Bagdad* 'Baghdad' and *Irak* 'Iraq', yielding a semantic accuracy of 81.4%. In contrast, the embeddings yield a semantic accuracy of only 30.0% for the *Currency* relation type, including analogies such as *Algerie* 'Algeria' – *dinar* 'dinar' and *Angola* 'Angola' – *kwanza* 'kwanza'.

6.1.2 Comparing word embedding frameworks

While the initial experiments in Section 6.1.1 focused on the effects of corpora and pre-processing of text, this subsection will focus on comparing various word embedding frameworks. We found in the previous subsections that the best full-form embedding model, in terms of total accuracy, was trained on NNC combined with NoWaC. Similarly, the best lemma embedding model, in terms of semantic accuracy, was trained on NNC. Word embeddings trained on NoWaC alone never

yielded best accuracy. Based on these results, we will conduct the subsequent evaluation experiments on word embeddings trained on NNC alone and on NNC combined with NoWaC. Furthermore, we want to keep parameters constant when comparing word embedding frameworks and we restrict the vocabularies to the 30K most similar words.

In addition to word2vec Skip-gram and CBOW full-form and lemma embeddings, we produce fastText and GloVe full-form and lemma word embeddings, each of which are trained on NNC and on NNC combined with NoWaC. The methods underlying these frameworks are described in Section 2.2.4. Again, our intention is not to discover the perfect word embedding model, and we employ default values for all hyperparameters, as described in Section 5.3.2 and Section 5.3.3. However, the default dimensionality of GloVe embeddings is 50, as opposed to 100 for word2vec and fastText. We therefore produce GloVe embeddings with vector sizes of 100 as well as 50 for comparison. Apart from that, we employ default values for all hyperparameters. In this way, we are able to compare word embeddings across frameworks. It is quite likely that individual parameter tuning will produce better performance, but such a search is out-of-scope of the current work.

In the following, we will first be discussing word2vec, FastText and GloVe full-form embeddings. Subsequently, we will be discussing the corresponding lemma embeddings.

Word2vec, FastText and GloVe full-form embeddings

Semantic, syntactic and total accuracy on the analogy test set for word2vec, fastText and GloVe full-form embeddings is presented in Table 6.7. We observe that fastText embeddings generally recognize analogies better compared to both word2vec and GloVe. More precisely, fastText Skip-gram trained on the combination of NNC and NoWaC yields the best total accuracy of 65.8%. Furthermore, fastText Skip-gram also gives the best semantic accuracy of 63.5% when trained on NNC. In the case of syntactic analogies, FastText CBOW trained on NNC yields the best accuracy of 70.4%. Also, fastText is superior to the other word embedding models in terms of training time. For example, the training time of fastText CBOW trained on NNC is about 1.5 hours, as opposed to over the double of word2vec CBOW trained on NNC. For both word2vec and fastText, CBOW embeddings are substantially more effective in training time compared to Skip-gram.

As discussed in Section 2.2.4, fastText embeddings take into account the internal structure of words and represent each word by the sum of its character n-gram embeddings. This might be particularly beneficial for embeddings trained for Norwegian, due to the compound nature of the language. For instance, the nominal phrase 'table tennis' is written

		% Acc.			
	Framework	Sem.	Syn.	Tot.	Training time
NNC	Word2Vec CBOW	38.5	56.6	48.5	3h09m
	Word2Vec SG	50.3	61.6	56.5	5h36m
	FastText CBOW	44.1	**70.4**	58.5	**1h32m**
	FastText SG	**63.5**	67.5	65.7	2h11m
	GloVe	50.9	57.9	54.7	3h42m
NNC+NoW.	Word2Vec CBOW	41.2	59.5	51.4	5h10m
	Word2Vec SG	49.7	63.6	57.4	9h50m
	FastText CBOW	42.1	67.1	56.0	2h15m
	FastText SG	63.0	68.1	**65.8**	3h41m
	GloVe	52.4	60.3	56.8	7h56m

Table 6.7: Accuracy on the semantic and syntactic sections and total accuracy on all relation types in the Norwegian Analogy Test Set for the various full-form embeddings trained on NNC and on NNC+NoWaC.

as a single word *bordtennis* in Norwegian. In general, it is expected that subword information might benefit fastText models in predicting syntactic analogies, as such analogies are related to the morphology of the words. For example, fastText Skip-gram trained on NNC and on the combination of NNC and NoWaC yields a syntactic accuracy of 67.5% and 68.1%, respectively. These accuracies are considerably better than those of word2vec Skip-gram and GloVe trained on the same corpora. The syntactic accuracy of fastText CBOW is at a comparable level.

Also, the subword information seem to benefit fastText Skip-gram embeddings in predicting semantic analogies. FastText Skip-gram trained on NNC and on the combination of NNC and NoWaC yields a semantic accuracy of 63.5% and 63.0%, respectively. However, this is not the case for fastText CBOW, of which the semantic accuracy is considerably lower than the syntactic. In fact, the difference between the semantic and syntactic accuracy of fastText CBOW trained on NNC is as great as 26.3 percentage points.

Interestingly, in the case of word2vec and fastText trained on NNC alone and on the combination of NNC and NoWaC, the semantic, syntactic and total accuracy is consistently higher for Skip-gram compared to CBOW. The only exception is fastText CBOW trained on NNC, which give a higher syntactic accuracy compared to fastText Skip-gram. Furthermore, as opposed to word2vec CBOW and GloVe, which consistently seem to benefit from more training data, fastText CBOW consistently yields lower accuracy, both semantic, syntactic and total, with increased size of the corpus.

As initially discussed, we produce GloVe embeddings with vector sizes of 100 as well as the default 50. Only GloVe embeddings with 100 dimensions are reported in Table 6.7 and Table 6.8. We find that semantic accuracy seem to be more sensitive to vector dimensionality

	Framework	% Sem. acc.	Training time
NNC	Word2Vec CBOW	46.0	3h32m
	Word2Vec SG	56.3	5h19m
	FastText CBOW	56.1	**1h19m**
	FastText SG	**68.1**	1h57m
	GloVe	59.7	2h53m
NNC+NoW.	Word2Vec CBOW	49.4	4h59m
	Word2Vec SG	54.8	8h31m
	FastText CBOW	55.3	1h59m
	FastText SG	65.7	2h47m
	GloVe	61.5	4h04m

Table 6.8: Accuracy on the semantic sections in the Norwegian Analogy Test Set for the various lemma embeddings trained on NNC and on NNC+NoWaC.

compared to syntactic accuracy. The preliminary GloVe embeddings trained on the combination of NNC and NoWaC with 50 dimensions yield a semantic accuracy of 28.3%. When trained with 100 dimensions, this accuracy increases with 24.1 percentage points to 52.4%. In contrast, the syntactic accuracy increases only with 13.3 percentage points for the same model. In any case, it is clear that vector dimensionality has a substantial effect on model performance and supplementary experiments on dimensionality will be discussed in Section 6.3.2.

Word2vec, FastText and GloVe lemma embeddings

Semantic accuracy on the analogy test set for word2vec, fastText and GloVe lemma embeddings is presented in Table 6.8. Again, we only report semantic accuracy for the lemma embeddings. Similarly as observed for fastText Skip-gram full-form embeddings, we find that fastText Skip-gram lemma embeddings trained on NNC predict semantic analogies with best accuracy. This model gives a semantic accuracy of 68.1%, which is 11.8 and 8.4 percentage points higher compared to the semantic accuracy of the corresponding word2vec Skip-gram and GloVe embeddings, respectively.

As noticed from Table 6.7, word2vec CBOW and GloVe full-form embeddings seem to consistently learn word representations better from more training data. This is also the case for the lemma embeddings, with an increased semantic accuracy of almost 4 percentage points for word2vec CBOW trained on NNC alone compared to the combination of NNC and NoWaC. In contrast, both word2vec Skip-gram and fastText Skip-gram embeddings yield slightly better semantic accuracy when trained on NNC alone.

Overall, we find that lemma embeddings predict semantic word

similarities better compared to full-form embeddings. Actually, all embedding models perform better on the semantic analogy questions when trained on lemmas compared to full-forms. A reason for this might be that the word similarity relations between the semantic analogy questions are not as morphologically based as the relations between the syntactic ones. Hence, lemmas, which do not include inflections, might predict them better. Also, each embedding gets more data as there are fewer unique words compared to full-form embeddings.

6.1.3 Methodological concerns

Regarding the evaluation of word embedding models using the analogy test set, there are some methodological issues that should be taken into account. First, the analogy test set is not balanced, e.g., the different relation types contain an unbalanced number of analogy questions. For instance, the *All capital cities* relation type contains a total of 4,524 questions, accounting for a little over 50% of the semantic questions. Moreover, the number of semantic and syntactic relation types differ, although the two sections include almost the same number of questions.

Second, we observe that the syntactic relation type *Nationality adjective* might as well be categorized as semantic, also pointed out by Fares et al. (2017). In the context of our evaluation results, this intuition is supported by Table 6.9, which shows the accuracy of word2vec Skip-gram lemma embeddings trained on NNC with the vocabulary restricted to the 30K most frequent words, broken down for the individual relation types within the semantic and syntactic sections. As previously stated, it is not meaningful to evaluate lemma embeddings on syntactic analogies, which can be reflected by the low accuracy for the syntactic relation types. However, the model yields a syntactic accuracy of 97.2% for the *Nationality adjective* relation type, which indicates that the questions within this relation type might be considered semantically related rather than syntactically related. Nevertheless, we follow the initial category sections as proposed by Mikolov, Chen et al. (2013).

Third, according to the Association for Computational Linguistics' website[1] reporting state-of-the-art results for the original Google Analogies Dataset, we find that researchers do not always report the size of the vocabulary used when evaluating word embeddings on the analogy task. As we recall from Table 6.1 and Table 6.2, restricting the vocabulary to the 30K or 1M most frequent words, substantially affects the number of questions considered, which in turn affects accuracy. As a consequence, evaluation scores for vocabularies restricted to different values, are not actually comparable.

[1] https://aclweb.org/aclwiki/Google_analogy_test_set_(State_of_the_art)

	Categories	# Correct	# Considered	# Total	% Acc.
Semantic	Common capital city	342	420	506	81.4
	All capital cities	1,043	1,370	4,524	76.1
	Currency	12	40	866	30.0
	City-in-county	953	2,445	2,542	39.0
	Man–woman	227	306	506	74.2
	Semantic total	2,577	4,581	8,944	56.3
Syntactic	Adjective-to-adverb	8	72	992	11.1
	Opposite	50	380	600	13.2
	Comparative	5	21	1,190	23.8
	Superlative	0	20	930	0.0
	Nationality adjective	1,331	1,369	1,599	97.2
	Past tense	0	2	1,560	0.0
	Plural nouns	0	5	1,122	0.0
	Present tense	3	30	870	10.0
	Syntactic total	1,397	1,899	8,863	73.6
	Total	3,974	6,480	17,807	61.3

Table 6.9: Number of correct and considered questions, total number of questions and accuracy for each semantic and syntactic relation type in the Norwegian Analogy Test Set for word2vec Skip-gram lemma embeddings trained on NNC with the vocabulary restricted to the 30K most frequent words.

6.2 Synonym extraction

We now turn to the second intrinsic method for vector evaluation that we have focused on, namely synonymy detection. We will utilize the synonym dictionary resource created in Chapter 4 and evaluate how well the word embedding models perform on the task of synonym extraction in terms of precision and recall as described in Section 4.3. As we recall, it is most meaningful to evaluate lemma embedding models, as the synonym dictionary entry words are lemmas. However, we will report results for all word embedding frameworks, i.e., word2vec, fastText and GloVe.

In Section 6.2.1, we will describe a final adjustment of the synonym dictionary with respect to word frequencies. In Section 6.2.2 we will present and discuss the results of the synonym extraction task. In section 6.2.3, we will detail a manual error analysis of some of the synonyms that were incorrectly found by the best model, and in Section 6.2.4 we will evaluate the models while restricting the vocabulary size.

Cut-off	# Headwords	# Tokens	# Types	# Avg. syn./head.
None	27,601	111,111	32,303	4.03
1	26,191	109,123	31,527	4.17
5	24,649	106,749	30,756	4.33
10	23,625	105,163	30,275	4.45

Table 6.10: Number of headwords, tokens, types and average number of synonyms per headword in the Norwegian Synonymy Test Set with various frequency cut-offs.

6.2.1 Synonym dictionary frequency cut-off

Preliminary to the evaluation experiments, we will make a final adaption of the synonym dictionary in terms of frequency. Each of the word embedding frameworks produce word vectors for words which occur a minimum of 5 times in the training corpus. Similarly, we modify the initial synonym dictionary so that each headword, and at least one of its synonyms, must occur 5 or more times in the concatenation of all lemmatized corpora, e.g., NBDigital, NNC and NoWaC. As long as one synonym occurs 5 or more times, all synonyms of the given headword are retained regardless of their frequencies.

For comparison, Table 6.10 shows the number of headwords, tokens, types and average number of synonyms per headword, in the initial synonym dictionary as well as the synonym dictionary with different cut-offs, e.g., 1, 5 and 10. Surprisingly, limiting the frequency of headwords and synonyms does not substantially decrease coverage. Possibly, some frequency threshold for headwords might have been considered in the creation of the synonym dictionary. In the following subsections, we will report results using the synonym dictionary with a frequency cut-off of 5.

6.2.2 Discussion of results

In this subsection, we will present and discuss the evaluation results for word2vec, fastText and GloVe lemma embeddings on the task of synonym extraction.

Evaluation is performed in two steps. Firstly, for each configuration of word embedding frameworks and corpora, a total of 25 different configurations, we find the 10 most similar words of each headword in the synonym dictionary for which we have an embedding in addition to having embeddings for at least one of its synonyms. Similar words are found by computing the cosine similarity between the vector of the given headword and, by default, the vector of every other word in the vocabulary, using the `similar_by_word()` method available in gensim. The 10 most similar words are those associated with the 10

highest cosine similarity scores.

Secondly, we evaluate performance in terms of precision and recall. As previously described in Section 4.3, we have for this task defined the standard evaluation metrics so that they can ultimately be viewed as two versions of accuracy. More precisely, we have defined precision to be the number of headwords for which we have found a correct synonym among its k most similar words, over the total number of headwords for which we have found synonyms. Furthermore, recall is defined to be the number of headwords for which we have found a correct synonym among its k most similar words, over the total number of headwords in the synonym dictionary.

We compute precision and recall scores for the 1, 5, and 10 most similar words found by word2vec, fastText and GloVe lemma embeddings trained on the various corpora. The results are given in Table 6.11. We have multiplied each score by 100 for better readability and the 1, 5 and 10 most similar words are reported as values of k. We observe that word2vec CBOW trained on NNC yield the overall best precision for both the 1, 5 and 10 most similar words. As previously stated, a prediction is considered to be correct if there is at least one correct synonym among the k most similar words. Hence, precision is expected to improve with increasing k. The model gives a best precision of 9.9% for k=1, 21.4% for k=5 and 26.5% for k=10. Furthermore, we find that word2vec CBOW embeddings consistently yield highest precision scores across corpora as well as values of k. These results contrast those of the analogy task which showed that word2vec Skip-gram embeddings, both full-forms and lemmas, consistently performed better compared to word2vec CBOW and that fastText embeddings performed the overall best.

In the case of recall, we also observe that word2vec CBOW embeddings yield the highest scores for all values of k. This model yields a best recall score of 8.8% for k=1, 19.7% for k=5 and 24.8% for k=10 when trained on the concatenation of all corpora. Note that recall scores for models trained on the combination of all corpora are equal to the corresponding precision scores. The reason for this is that the synonym dictionary has been modified with respect to the combination of all lemmatized corpora, as described in Section 6.2.1. Thus, all headwords will have embeddings and the denominator of precision and recall will be the same. Furthermore, we find that word2vec CBOW embeddings consistently yield the highest recall scores across corpora as well as values of k.

We find that the worst performing word embedding model, in terms of both precision and recall, is fastText CBOW. As we can see from Table 6.11, the fastText CBOW embeddings yield substantially lower precision and recall scores across all corpora and values of k compared to the other models, with fastText Skip-gram performing slightly better. The overall lowest precision and recall scores are reported for fastText

	Model	k=1 % P	k=1 % R	k=5 % P	k=5 % R	k=10 % P	k=10 % R
NBDigital	Word2Vec CBOW	5.5	5.0	14.8	13.3	19.7	17.7
	Word2Vec SG	3.5	3.2	9.8	8.8	13.0	11.7
	FastText CBOW	1.3	1.2	4.0	3.6	6.2	5.6
	FastText SG	1.5	1.4	4.9	4.4	8.0	7.2
	GloVe	4.0	3.6	11.7	10.5	15.9	14.3
NNC	Word2Vec CBOW	**9.9**	8.5	**21.4**	18.3	**26.5**	22.7
	Word2Vec SG	8.2	7.0	18.6	15.9	23.3	19.9
	FastText CBOW	2.9	2.5	7.5	6.4	11.0	9.4
	FastText SG	3.3	2.8	10.5	9.0	16.1	13.8
	GloVe	9.0	7.7	19.7	16.8	24.5	21.0
NoWaC	Word2Vec CBOW	8.6	7.4	18.4	15.8	23.5	20.2
	Word2Vec SG	6.2	5.3	13.6	11.7	17.1	14.7
	FastText CBOW	2.2	1.9	6.1	5.3	9.1	7.8
	FastText SG	2.2	1.9	7.4	6.4	11.3	9.7
	GloVe	5.4	4.6	13.5	11.6	18.0	15.4
NNC+NoW.	Word2Vec CBOW	9.4	8.7	20.5	18.9	25.7	23.7
	Word2Vec SG	7.2	6.6	16.7	15.4	20.9	19.3
	FastText CBOW	2.7	2.5	6.7	6.2	10.0	9.2
	FastText SG	2.9	2.7	9.4	8.7	14.2	13.1
	GloVe	8.0	7.4	17.7	16.3	22.7	20.9
All	Word2Vec CBOW	8.8	**8.8**	19.7	**19.7**	24.8	**24.8**
	Word2Vec SG	5.8	5.8	14.1	14.1	18.0	18.0
	FastText CBOW	1.4	1.4	4.9	4.9	7.4	7.4
	FastText SG	1.9	1.9	6.6	6.6	10.3	10.3
	GloVe	6.8	6.8	16.4	16.4	21.5	21.5

Table 6.11: Precision and recall scores for the 1, 5, and 10 most similar words found for the task of synonym extraction by word2vec CBOW and Skip-gram, fastText CBOW and Skip-gram and GloVe lemma embeddings trained on the various corpora.

CBOW trained on NBDigital, with 1.3% precision and 1.2% recall for k=1. In contrast, fastText CBOW embeddings were one of those yielding the best accuracy on the analogy test set. The subword information included in the fastText model, which seemingly benefited it in predicting analogies, does not seem to aid the embeddings as much in detecting synonyms.

In general, the relative differences in precision and recall scores between the various word embedding frameworks seem to be consistent, both across corpora and values of k. Word2vec CBOW embeddings always yield the best scores, both precision and recall, across corpora and for all values of k, followed by GloVe, word2vec Skip-gram, fastText Skip-gram and fastText CBOW. Only some minor exceptions from this overall trend are observed for models trained on NoWaC for k=1 and k=5.

Furthermore, the relative differences in precision scores between the various corpora seem to be consistent. Word embeddings trained on NNC consistently yield the best precision scores across embedding frameworks and values of `k`, followed by models trained on the combination of NNC and NoWaC. In the case of NoWaC alone and the combination of all corpora, we find a few inconsistencies. However, models trained on NBDigital always yield the lowest precision scores across frameworks and `k`. Also, recall scores generally decrease with the size of the corpora, with the combination of all corpora yielding the highest recall scores and NBDigital as the smallest corpus yielding lowest recall scores.

Regardless of relative differences, we find that both precision and recall scores across word embedding frameworks, corpora and values of `k` are quite low. Therefore, in the next subsection we will perform a manual error analysis of the most similar words found by the best performing model in terms of precision, namely word2vec CBOW trained on NNC.

6.2.3 Error analysis

As we can see from Table 6.11, the precision and recall scores are quite low for all model configurations. This was also the case for other studies on synonym extraction for English, for example the study performed by Leeuwenberg et al. (2016). However, only considering precision or recall might not reflect the actual performance of the word embedding models. In many cases, the found synonyms are actually good suggestions, but they are simply not covered in the synonym dictionary. In other cases, different types of relations rather than synonymy are captured, such as antonymy, hypernymy, hyponymy or topical similarity.

In this subsection, we will examine in more depth the most similar words of word2vec CBOW trained on NNC, which yielded the best

precision scores for both the 1, 5 and 10 most similar words, as shown in Table 6.11. We randomly choose a selection of 50 words for which none of the found synonyms were considered correct in the automatic evaluation. Based on the error categories proposed by Leeuwenberg et al. (2016), we manually categorize the 1st and 2nd most similar words found of each chosen word. The categories that were observed in our analysis are listed below.

- **Human-judged synonyms**: Synonyms judged by a native Norwegian speaker.
- **Spelling variants**: Spelling variants and misspellings.
- **Related**: The two words are semantically related or similar in domain.
- **Unrelated/unknown**: The two words are unrelated or the relation between them is unknown.
- **Names**: Proper nouns.
- **Co-hyponyms**: The two words share a hypernym.
- **Inflections**: Inflections of the given word.
- **Hyponyms**: The found word is of more specific meaning.
- **Contrastive**: The two words are contrastive in meaning.
- **Hypernym**: The found word is of less specific meaning.
- **Foreign**: A non-Norwegian word.

The results of the analysis are given in Table 6.12. Each category is shown in the first column. The second and third columns show the number of times the 1st or 2nd most similar word, respectively, was categorized in a given category. The last column shows two examples of each category, where the word in front of the dash is the given word and the word behind it is either its 1st or 2nd most similar word.

It is worth noting that some categories are vaguely defined and one word is found to fit into two categories. That is, *styrkeprøve* and *kraftprøve*, both meaning test of strength, are considered human-judged synonyms as well as co-hyponyms. Similarly as found by Leeuwenberg et al. (2016), we find that the two largest error categories are *Related* and *Unrelated/unknown*. Word embeddings generally capture both similarity in content, known as *sameness*, and similarity in domain, known as *relatedness*. For example, *bue* 'bow' and *trommeskinn* 'drum skin' are related in terms of both being some kind of musical instrument equipment, i.e., they are similar in domain. Although, *bue* and *trommeskinn* do not have the same meanings, and are thus not considered synonyms, the two words may well occur in similar contexts. Hence, the model finds *trommeskinn* as the 2nd most similar word to *bue*. Similarly, words contrastive in meaning, i.e., antonyms,

Category	Most similar 1st	2nd	Example
Human-judged synonyms	6	2	fjomp / dust, styrkeprøve / kraftprøve
Spelling variants	6	2	blackout / black-out, idérikdom / iderikdom
Related	11	17	innbilning / vrøvl, nervøsitet / pessimisme
Unrelated/unknown	11	15	ærend / skjulested, intendere / uhistorisk
Names	2	1	avtrede / Kanofarten, ponere / Zillertal
Co-hyponyms	2	4	kobra / tarantell, styrkeprøve / manndomsprøve
Inflections	4	3	bløt / bløte, samstemme / samstemt
Hyponyms	6	3	bue / fiolinbue, utslipp / CO2-utslipp
Contrastive	2	-	ekvivalent / motstykke, negativ / positiv
Hypernym	-	1	amfora / leirkrukke
Foreign	1	2	futhark / fæstkultur, sjakkel / wire

Table 6.12: Manual categorization of 50 randomly selected words for which none of the synonyms found for the task of synonym extraction by word2vec CBOW lemma embeddings trained on NNC were considered correct.

tend to appear in similar contexts, which is a well-known challenge for distributional models. For example, *positiv* 'positive' is found as the 1st most similar word to *negativ* 'negative'.

Actually, in a minimum of 12 out of 50 instances the model finds a correct most similar word, which is either a human-judged synonym or a misspelling of the given word. This indicates that the evaluation is rather strict, which might partially explain the low scores. Moreover, the inflected words are near-synonyms but their parts-of-speech differ. Leeuwenberg et al. (2016) demonstrated that the incorporation of POS tags can help avoid such errors. In the next section, we will explore to what extent restricting the size of the vocabulary affects the scores.

6.2.4 Restricting the vocabulary size

The precision and recall scores reported in Section 6.2.2, were computed without any restriction of the vocabulary size. However, similarly as we limited the size of the vocabulary when computing accuracy on the analogy test set in Section 6.1.1, we can limit the vocabulary size when finding the k most similar words of a given word. More precisely, we restrict the vocabulary size for the embeddings after training. In this subsection, we will present evaluation results when limiting the vocabulary size.

As discussed in Section 6.1.1, limiting the vocabulary size affects accuracy on the analogy test set because it controls the number of analogy questions that are considered. Restricting the vocabulary size when finding the k most similar words of a given word works similarly. The gensim method for finding most similar words takes an optional parameter `restrict_vocab`, which limits the number of

		30K vocab.						1M vocab.					
		k=1		k=5		k=10		k=1		k=5		k=10	
	Model	% P	% R	% P	% R	% P	% R	% P	% R	% P	% R	% P	% R
NBDigital	Word2Vec CBOW	6.7	6.0	16.1	14.5	21.0	18.8	5.5	5.0	14.8	13.3	19.7	17.7
	Word2Vec SG	5.7	5.1	14.3	12.9	19.0	17.0	3.5	3.2	9.9	8.9	13.2	11.8
	FastText CBOW	3.9	3.5	11.5	10.3	15.8	14.2	1.7	1.5	5.1	4.6	8.2	7.4
	FastText SG	4.1	3.6	11.7	10.5	16.0	14.4	1.9	1.7	6.3	5.7	10.1	9.1
	GloVe	4.9	4.4	12.4	11.1	16.5	14.8	4.2	3.8	12.1	10.9	16.2	14.6
NNC	Word2Vec CBOW	**10.3**	8.8	**21.3**	18.2	**26.5**	22.7	**9.9**	8.5	**21.4**	18.3	**26.5**	22.7
	Word2Vec SG	10.1	8.6	20.8	17.8	25.9	22.2	8.2	7.0	18.6	15.9	23.4	20.0
	FastText CBOW	6.9	5.9	16.9	14.5	22.1	18.9	3.1	2.6	8.3	7.1	12.5	10.7
	FastText SG	8.6	7.4	19.9	17.0	25.1	21.5	3.6	3.0	11.8	10.1	17.7	15.2
	GloVe	8.4	7.2	18.8	16.1	23.7	20.3	9.1	7.8	19.7	16.9	24.6	21.1
NoWaC	Word2Vec CBOW	9.3	8.0	19.4	16.7	24.5	21.0	8.6	7.4	18.4	15.8	23.5	20.2
	Word2Vec SG	8.6	7.4	18.9	16.3	24.0	20.1	6.2	5.3	13.6	11.7	17.1	14.7
	FastText CBOW	5.4	4.7	14.6	12.5	19.2	16.5	2.2	1.9	6.4	5.5	9.6	8.3
	FastText SG	6.3	5.4	16.5	14.1	21.2	18.2	2.3	2.0	7.9	6.8	12.0	10.3
	GloVe	6.4	5.5	14.7	12.6	19.0	16.4	5.5	4.8	13.7	11.8	18.1	15.6
NNC+NoW.	Word2Vec CBOW	10.2	9.4	21.2	19.6	26.3	24.2	9.4	8.7	20.6	19.0	25.7	23.7
	Word2Vec SG	9.4	8.7	19.8	18.2	24.9	23.0	7.3	6.7	17.0	15.6	21.4	19.7
	FastText CBOW	6.5	6.0	15.9	14.7	20.7	19.1	3.1	2.9	8.5	7.8	12.5	11.6
	FastText SG	8.0	7.4	18.6	17.2	23.7	21.9	3.5	3.2	11.5	10.6	17.0	15.7
	GloVe	8.1	7.5	17.4	16.1	22.0	20.3	8.5	7.8	18.3	16.8	23.1	21.3
All	Word2Vec CBOW	9.5	**9.5**	20.4	**20.4**	25.5	**25.5**	8.8	**8.8**	19.7	**19.7**	24.9	**24.9**
	Word2Vec SG	7.7	7.7	17.2	17.2	21.9	21.9	5.9	5.9	14.6	14.6	19.0	19.0
	FastText CBOW	5.4	5.4	13.8	13.8	18.3	18.3	2.2	2.2	7.6	7.6	11.2	11.2
	FastText SG	6.0	6.0	15.5	15.5	20.5	20.5	2.6	2.6	9.4	9.4	14.8	14.8
	GloVe	7.0	7.0	16.0	16.0	20.6	20.6	7.5	7.5	17.2	17.2	22.3	22.3

Table 6.13: Precision and recall scores for the 1, 5, and 10 most similar found for the task of synonym extraction by word2vec CBOW and Skip-gram, fastText CBOW and Skip-gram and GloVe lemma embeddings trained on the various corpora with the vocabulary restricted to the 30K and 1M most frequent words.

vectors that are considered for computing cosine similarities with a given headword, as opposed to using vectors of every other word in the model. Here, we explicitly limit the vocabulary size to 30K and 1M, which indicates that the vectors of the 30K and 1M most frequent words will be considered.

Table 6.13 presents the precision and recall scores for the 1, 5 and 10 most similar words found by word2vec, fastText and GloVe lemma embeddings trained on the various corpora with the vocabulary restricted to the 30K and 1M most frequent words. Generally, the results follow previously observed trends for models without restricting the vocabulary size. For instance, word2vec CBOW trained on NNC yield the overall best precision across all values of k as well as for all vocabulary sizes. Similarly, word2vec CBOW trained on the combination of all corpora consistently yield the best recall scores.

Moreover, fastText CBOW is still generally the worst performing model in terms of both precision and recall scores across corpora, values of k and vocabulary sizes. However, fastText CBOW embeddings yield substantially better precision and recall scores with the vocabulary restricted to the 30K most frequent words compared to 1M. For example,

fastText CBOW trained on NNC improves precision from 3.1% to 6.9% and recall from 2.6% to 5.9% for k=1. For comparison, word2vec CBOW trained on NNC improves precision from 9.9% to 10.3% and recall from 8.5% to 8.8% for k=1. Interestingly, while word2vec and fastText CBOW and Skip-gram embeddings consistently improve scores when limiting the vocabulary to a smaller size, this is not the case for GloVe embeddings.

Generally, the relative differences in precision and recall scores between the various word embedding frameworks seem to be consistent across corpora and values of k with the vocabulary restricted to the 1M most frequent words. Here, the relative differences are similar to those without restricting the vocabulary. In contrast, the relative differences are not as consistent with the vocabulary restricted to the 30K most frequent words. Although, word2vec CBOW embeddings consistently yield the best precision and recall scores and fastText CBOW embeddings consistently yield the worst, we observe variations in the relative difference between the other models. These variations seem to be associated with increasing value of k as well as increasing size of the corpus.

As we have observed, limiting the vocabulary to a smaller size generally improves precision and recall across embedding framework, corpora and values of k. These findings are possibly due to some frequency bias in which a restriction of the vocabulary size might help avoid. For instance, if a headword in the synonym dictionary is very infrequent in the training corpus, the embeddings might associate it the most with another very infrequent word, even though the correct synonym is perhaps found within the 30K most frequent words. Restricting the vocabulary is thus a way of forcing the embeddings to find most similar words within a set of more frequent words.

6.3 Supplementary experiments

As a supplement to the analogical reasoning and synonym extraction experiments in Section 6.1 and Section 6.2, respectively, we will in this section perform a few additional experiments focusing on certain parameters. First, we will investigate to what extent OCR errors affect the quality of word embeddings. Second, we will explore the effects of word vector dimensionality.

6.3.1 OCR cut-offs

As found in the previous sections, all word embedding models yielded the worst evaluation results when trained on NBDigital, for both the task of analogical reasoning and the task of synonym extraction.

Moreover, as stated in Section 5.1.3, NBDigital contains OCR errors, i.e., errors caused by the process of automatically converting scanned documents into machine-readable text. For this reason, we want to quantify the effects of OCR errors on the quality of word embeddings by experimenting with different NBDigital cut-offs. These cut-offs are based on the OCR software's confidence in the quality of the text, i.e., the OCR confidence values. We will limit the subsequent experiments to word2vec CBOW and Skip-gram full-form and lemma embeddings.

Semantic, syntactic and total accuracy on the analogy test set for word2vec CBOW and Skip-gram full-form embeddings trained on the various NBDigital cut-offs with vocabularies restricted to the 30K and 1M most frequent words are presented in Table 6.14. As previously observed, word2vec Skip-gram embeddings perform better on both semantic and syntactic analogies and yield the best total accuracy across vocabulary sizes compared to word2vec CBOW embeddings. This is also the case for word2vec Skip-gram embeddings trained on the various cut-offs.

In general, it is clear that the results are influenced by the variation of OCR cut-off, but the tendencies are not completely clear-cut. We observe that word2vec Skip-gram embeddings trained on NBDigital with a 0.85 OCR cut-off yield a best semantic accuracy of 27.0%, in addition to a best total accuracy of 43.2% with the vocabulary restricted to the 30K most frequent words. Furthermore, word2vec Skip-gram embeddings trained on NBDigital with a 0.95 OCR cut-off yield the best syntactic accuracy of 55.0% with the 30K word vocabulary. In contrast, with the 1M word vocabulary, word2vec Skip-gram embeddings trained on NBDigital without any cut-off yield the best syntactic and total accuracy. Nevertheless, none of these configurations yield better accuracy than the best results of word2vec Skip-gram embeddings presented in Table 6.3, and are still far below the results obtained for other training corpora.

Semantic accuracy on the analogy test set for word2vec CBOW and Skip-gram lemma embeddings trained on the various NBDigital cut-offs with vocabularies restricted to the 30K and 1M most frequent words are presented in Table 6.15. Again, word2vec Skip-gram embeddings yield better results compared to word2vec CBOW. For both models, a 0.85 OCR cut-off gives better semantic accuracy compared to no cut-off or a 0.95 cut-off. Moreover, no OCR cut-off gives better accuracy compared to a 0.95 cut-off. Possibly, this is due to the corpus decreasing too much in size. But again, none of the accuracies are better than the best accuracy reported for the embeddings trained on other corpora in Table 6.5.

Table 6.16 presents the precision and recall scores for the 1st most similar words found by word2vec CBOW and Skip-gram lemma embeddings trained on the various NBDigital cut-offs with the vocabulary restricted to the 30K and 1M most frequent words. We choose to report results

		30K vocab.			1M vocab.		
		% Acc.					
	OCR	Sem.	Syn.	Tot.	Sem.	Syn.	Tot.
CBOW	None	14.8	49.3	34.9	8.6	32.1	22.0
	0.85	16.3	50.9	36.9	9.0	31.5	21.8
	0.95	14.9	50.4	36.0	8.0	29.7	20.4
SG	None	26.4	53.6	42.3	13.4	**36.3**	**26.5**
	0.85	**27.0**	54.3	**43.2**	**13.8**	34.9	25.9
	0.95	22.1	**55.0**	41.7	11.6	32.5	23.5

Table 6.14: Accuracy on the semantic and syntactic sections and total accuracy on all relation types in the Norwegian Analogy Test Set for word2vec CBOW and Skip-gram full-form embeddings trained on the various NBDigital OCR cut-offs with the vocabulary restricted to the 30K and 1M most frequent words.

	OCR	30K vocab. % Sem. acc.	1M vocab. % Sem. acc.
CBOW	None	18.2	9.6
	0.85	20.1	9.9
	0.95	15.8	8.2
SG	None	29.2	14.7
	0.85	**32.0**	**15.4**
	0.95	23.7	12.0

Table 6.15: Accuracy on the semantic sections in the Norwegian Analogy Test Set for word2vec CBOW and Skip-gram lemma embeddings trained on the various NBDigital OCR cut-offs with the vocabulary restricted to the 30K and 1M most frequent words.

		30K vocab.		1M vocab.	
		k=1		k=1	
	OCR	% P	% R	% P	% R
CBOW	None	6.7	6.0	5.5	5.0
	0.85	**7.0**	**6.2**	6.0	**5.3**
	0.95	**7.0**	5.8	**6.1**	5.1
SG	None	5.7	5.1	3.5	3.2
	0.85	6.2	5.5	3.8	3.4
	0.95	6.5	5.4	4.0	3.3

Table 6.16: Precision and recall scores for the 1st most similar words found for the task of synonym extraction by word2vec CBOW and Skip-gram lemma embeddings trained on the various NBDigital OCR cut-offs with the vocabulary restricted to the 30K and 1M most frequent words.

for only k=1, as the relative differences between word2vec CBOW and Skip-gram embeddings across values of k were generally found to be consistent in Table 6.13.

In contrast to performance on the analogy test set, reported in Table 6.15, for the task of synonym extraction we find that word2vec CBOW lemma embeddings yield better scores compared to word2vec Skip-gram. Word2vec CBOW embeddings yield a shared best precision of 7.0% when trained on NBDigital with a 0.85 and 0.95 OCR cut-off and with the vocabulary restricted to the 30K most frequent words. For both vocabulary sizes, a 0.85 OCR cut-off yields the best recall. Word2vec Skip-gram embeddings trained on the original NBDigital corpus actually yield the worst precision and recall scores for both vocabulary sizes. Overall, OCR cut-offs marginally improve results but results obtained for embeddings trained on NBDigital are still far worse than the results for other training corpora.

6.3.2 Vector dimensionality

In previous sections, when we evaluated GloVe embeddings with 50 and 100 dimensions on the analogy test set, we observed that vector dimensionality has a substantial effect on model performance. In this subsection, we will investigate in further detail the effects of vector dimensionality.

As we can see from Table 6.7 in the section discussing full-form word embedding frameworks, fastText Skip-gram trained on the combination of NNC and NoWaC with the vocabulary restricted to the 30K most frequent words give the highest total accuracy on the analogy test set. Therefore, we will perform the first supplementary vector dimensionality experiments on this model configuration. Semantic, syntactic and total accuracy of embeddings with vector sizes of 50, 300

		% Acc.			
	Dim.	Sem.	Syn.	Tot.	Training time
FT SG	50	43.3	57.1	51.0	**2h21m**
	100	63.0	68.1	65.8	3h41m
	300	80.9	**70.2**	**75.0**	7h05m
	600	**82.2**	67.9	74.3	14h36m

Table 6.17: Accuracy on the semantic and syntactic sections and total accuracy on all relation types in the Norwegian Analogy Test Set for FastText Skip-gram full-form embeddings trained on NNC+NoWaC with different dimensions and the vocabulary restricted to the 30K and 1M most frequent words.

and 600 is given in Table 6.17.

It is clear that vector dimensionality has a great impact on how well the embeddings recognize analogies. FastText Skip-gram full-form embeddings with 600 dimensions yield the highest semantic accuracy of 82.2%, almost 20 percentage points better than those with default 100 dimensions and nearly 40 percentage points better than those with 50 dimensions. However, such large vectors require extensive training time. Actually, vectors of 600 dimensions require 14.5 hours of training, whereas vectors of 50 dimensions only require a little under 2.5 hours. Embeddings with 300 dimensions yield the best syntactic and total accuracy of 70.2% and 75.0%, respectively, 13.1 and 24 percentage points better than the lowest syntactic and total accuracy given by the embeddings with 50 dimensions. Seemingly, semantic analogies are even more sensitive to vector dimensionality compared to the syntactic.

Furthermore, we observe from Table 6.8, in the section discussing lemma embedding frameworks, that fastText Skip-gram lemma embeddings trained on NNC with the vocabulary restricted to the 30K most frequent words yield the best semantic accuracy. Hence, we report semantic accuracy of this model with vector sizes of 50, 100, 300 and 600 in Table 6.18. We find that fastText Skip-gram lemma embeddings with 300 dimensions yield the highest semantic accuracy of 80.9%, slightly lower than the best semantic accuracy of the full-form embeddings in Table 6.17. Thus, with more dimensions, the previously observed result that lemma embeddings always recognize semantic analogies better compared to full-form embeddings, does no longer hold. However, we find that fastText Skip-gram lemma embeddings generally require substantially less training time than full-form embeddings. For instance, lemma embeddings with 600 dimensions require less than half the amount of training time compared to fastText Skip-gram full-form embeddings with 600 dimensions.

We observed from Table 6.13, in the section discussing results of syn-

	Dim.	% Sem. acc.	Training time
FT SG	50	50.5	**1h20m**
	100	68.1	1h57m
	300	**80.9**	4h39m
	600	79.8	7h16m

Table 6.18: Accuracy on the semantic sections in the Norwegian Analogy Test Set for FastText Skip-gram lemma embeddings trained on NNC with different dimensions and the vocabulary restricted to the 30K and 1M most frequent words.

		30K vocab.		1M vocab.	
		k=1		k=1	
	Dim.	% P	% R	% P	% R
CBOW	50	8.2	7.0	8.2	7.0
	100	10.3	8.8	9.9	8.5
	300	12.5	10.7	11.5	9.8
	600	**13.3**	**11.4**	**12.2**	**10.4**

Table 6.19: Precision and recall scores for the 1st most similar words found for the task of synonym extraction by word2vec CBOW lemma embeddings trained on NNC with different dimensions and the vocabulary restricted to the 30K and 1M most frequent words.

onymy detection, that word2vec CBOW lemma embeddings trained on NNC yielded the best precision scores and we perform the dimensionality experiments on this model configuration. Table 6.19 presents precision and recall scores for the 1st most similar words found by word2vec CBOW lemma embeddings trained on NNC with different dimensions and with the vocabulary restricted to the 30K and 1M most frequent words. Again, we only report results for k=1. It is clear that increasing the size of the vectors improves synonymy detection. We find that embeddings with 600 dimensions consistently yield the best precision and recall scores with the vocabulary restricted to the 30K as well as the 1M most frequent words. Moreover, precision and recall scores consistently increase with increasing vector size and the relative difference between the various vector sizes is the same across vocabulary size.

6.4 Summary

In this chapter, we have presented a range of evaluation experiments and results for different distributional semantic models on the tasks of analogical reasoning and synonym extraction. Moreover, we have attempted to isolate the effects of various types of corpora, preprocessing of text, frameworks and certain hyperparameters such as vector dimensionality. We will summarize the main findings across

evaluation experiments in the conclusion of Chapter 7.

Chapter 7

Conclusion

In this work, we have created two benchmark data sets that enable intrinsic evaluation of distributional semantic models for Norwegian. While such resources are available for English, they did not exist for Norwegian prior to this work. Furthermore, we have produced large-coverage semantic vectors trained on a selection of corpora, applying different levels of text pre-processing, using several word embedding frameworks. Finally, we have demonstrated the usefulness of the evaluation resources created in the context of this thesis for ranking the relative performance of different word embedding models.

As detailed in Chapter 3, we have created the Norwegian Analogy Test Set for the purpose of evaluating model performance in recognizing so-called analogies, like *granddaughter* is to *grandson* as *sister* is to *brother*. The resource has been created by semi-automatically adapting the original Google Analogies Dataset proposed by Mikolov, Chen et al. (2013). Following automatic translation, we have put considerable effort into post-processing the test set, both in terms of correcting translational errors, and modifying several of the translated analogies due to language-specific differences between Norwegian and English as well as cultural variation. Moreover, the Norwegian Analogy Test Set and an associated evaluation script have been made available online in a separate repository[1] under the official Language Technology Group (LTG) organization on GitHub.

We have also created the Norwegian Synonymy Test Set which can be used to evaluate model performance in the task of extracting synonyms, as described in Chapter 4. The resource is based on *Norske synonymer blå ordbok*, which is a Norwegian synonym dictionary created by Dag Gundersen and published by Kunnskapsforlaget. In the initial stages of this project, we formed a collaboration with Kunnskapsforlaget and they provided a digital version of the synonym dictionary. Moreover, they permitted us to mine information from it

[1] https://github.com/ltgoslo/norwegian-analogies

in order to semi-automatically compile a benchmark data set suitable for testing distributional models. There were several advantages of this collaboration. First, we did not need to use annotators to create a resource from scratch. Second, the dictionary is created by professional lexicographers and is therefore reliable and of high quality. Third, it has large coverage with over 28,000 entry words. However, we have performed extensive work on extracting entry words and associated synonyms, especially regarding spelling variants which are both explicitly and implicitly encoded. The Norwegian Synonymy Test Set and scripts for evaluation have also been made available in a separate repository[2] under the LTG organization on GitHub and with the CC BY-NC-SA 4.0 licence.[3] Additionally, we have provided a version of the test set in which synonyms are grouped by word sense. Such a resource can be used for evaluating model performance on the task of word sense induction.

In order to assess the usefulness of these resources, we have trained large-coverage semantic vectors for Norwegian on various corpora, such as the Norwegian Newspaper Corpus, the Norwegian Web as Corpus (Guevara, 2010) and the NBDigital Corpus, using popular word embedding frameworks like word2vec (Mikolov, Chen et al., 2013), fastText (Bojanowski et al., 2016) and GloVe (Pennington, Socher and Manning, 2014). Given the size of these corpora, the tasks of tokenization, lemmatization and training embeddings are computationally demanding in terms of time as well as memory usage. Therefore, we have performed these tasks on a high performance computing facility – Abel – owned by the University of Oslo (UiO) and the Norwegian Metacenter for High Performance Computing (NOTUR). A substantial amount of time and effort was devoted to setting up the cluster environment and learning the queue system. However, the processing capacity of the cluster clearly benefited our work. Due to the importance of replicability, and also to facilitate re-use, the word embeddings trained in the context of this project will be made available online in the model repository[4] maintained by LTG and The Nordic Language Processing Laboratory (NLPL). Furthermore, the various pre-processed corpora will be made available to the extent permitted by their licences.

As demonstrated in Chapter 6, we have used the created evaluation resources to rank the relative performance of different models. Generally, we observed that word embeddings trained on NNC alone or on the combination of NNC and NoWaC gave best results for both tasks. Thus, more training data does not necessarily yield better results. Also, the effects of different types of corpora were found to be consistent across tasks. For instance, embeddings trained on NBDigital always gave low-

[2] https://github.com/ltgoslo/norwegian-synonyms
[3] https://creativecommons.org/licenses/by-nc-sa/4.0/
[4] http://vectors.nlpl.eu/repository/

est scores. Furthermore, there was no single model that performed best on both tasks, but the results within the individual tasks were very consistent. We found that fastText Skip-gram embeddings performed superior to the other models in recognizing analogies. Trained with default hyperparameters, these embeddings gave a best total accuracy of 65.8% for full-forms and a best semantic accuracy of 68.1% for lemmas. Similarly, word2vec CBOW embeddings consistently yielded best precision and recall scores for the task of synonym extraction. For instance, these embeddings gave a best precision of 21.4% for the 5 most similar words without restricting the vocabulary. Moreover, the large spread in scores across model configurations, for both tasks, indicate that the resources are well suited to describe the strengths and differences of the various models.

7.1 Future work

Model parameter tuning and optimization was out-of-scope of this project. However, we have shown that the evaluation resources that have been created in the context of this thesis are applicable for defining a relative ranking of different model configurations. Our supplementary evaluation experiments showed that parameters such as vector dimensionality substantially affect model performance and it is likely to assume that other parameters will as well. Hence, it would be interesting to examine in further detail the effects of various hyperparameters. Moreover, it would be interesting to evaluate the performance of traditional count-based models for Norwegian and compare the results with those that have been obtained for the prediction-based models in this work.

Intrinsic evaluation of distributional semantic models has been a subject of criticism (Faruqui et al., 2016; Chiu, Korhonen and Pyysalo, 2016). The quality of word representations is commonly assessed in terms of correlation with human judgments, but word representations that perform well on such benchmark data sets do not necessarily improve performance when they are added as features into downstream tasks, such as sentiment analysis or text classification. Therefore, a suggestion for future work is to study the relation between intrinsic and extrinsic evaluation of model performance for Norwegian.

For the task of synonym extraction, one can examine the extent to which the *relative cosine similarity*, proposed by Leeuwenberg et al. (2016) and described in Section 2.3.1, might improve precision. Also, the effects of incorporating part-of-speech tags can be further explored.

Furthermore, as we additionally have provided a version of the Norwegian Synonymy Test Set in which the synonyms are grouped by word sense, another suggestion for future work is to evaluate model performance on a word sense induction task.

Bibliography

Agirre, E. et al. (2009). A study on similarity and relatedness using distributional and WordNet-based approaches. In: *Proceedings of the 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Boulder, CO, USA, pp. 19–27.

Baroni, M., G. Dinu and G. Kruszewski (2014). Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Baltimore, MD, USA, pp. 238–247.

Batchkarov, M. et al. (2016). A critique of word similarity as a method for evaluating distributional semantic models. In: *Proceedings of the 1st Workshop on Evaluating Vector Space Representations for NLP*. Berlin, Germany, pp. 7–12.

Bojanowski, P. et al. (2016). Enriching Word Vectors with Subword Information. In: *arXiv preprint arXiv:1607.04606*.

Chiu, B., A. Korhonen and S. Pyysalo (2016). Intrinsic Evaluation of Word Vectors Fails to Predict Extrinsic Performance. In: *Proceedings of the 1st Workshop on Evaluating Vector Space Representations for NLP*. Berlin, Germany, pp. 1–6.

Fares, M. et al. (2017). Word vectors, reuse, and replicability: Towards a community repository of large-text resources. In: *Proceedings of the 21st Nordic Conference of Computational Linguistics*. Gothenburg, Sweden, pp. 271–276.

Faruqui, M. et al. (2016). Problems With Evaluation of Word Embeddings Using Word Similarity Tasks. In: *Proceedings of the 1st Workshop on Evaluating Vector Space Representations for NLP*. Berlin, Germany, pp. 30–35.

Ferraresi, A. et al. (2008). Introducing and evaluating ukWaC, a very large web-derived corpus of English. In: *Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can we beat Google?* Marrakech, Morocco, pp. 47–54.

Finkelstein, L. et al. (2002). Placing Search in Context: The Concept Revisited. In: *ACM Transactions on Information Systems* 20.1, pp. 116–131.

Guevara, E. (2010). NoWaC: a large web-based corpus for Norwegian. In: *Proceedings of The 11th Annual Conference of the North Amer-*

ican Chapter of the Association for Computational Linguistics: Human Language Technologies 2010, Sixth Web as Corpus Workshop. Los Angeles, CA, USA, pp. 1–7.

Harris, Z. S. (1954). Distributional Structure. In: *Word* 10, pp. 146–162.

Hill, F., R. Reichart and A. Korhonen (2015). SimLex-999: Evaluating Semantic Models with (Genuine) Similarity Estimation. In: *Computational Linguistics* 41.4, pp. 665–695.

Johannessen, J. B. et al. (2012). OBT+stat: A combined rule-based and statistical tagger. In: *Exploring Newspaper Language: Corpus compilation and research based on the Norwegian Newspaper Corpus*. Ed. by A. Gisle. John Benjamins Publishing Company, pp. 51–65.

Jurafsky, D. and J. H. Martin (2009). *Speech and Language Processing*. 2nd ed. Pearson Education: Prentice Hall.

— (2017). *Speech and Language Processing*. 3rd ed. draft. URL: https://web.stanford.edu/~jurafsky/slp3/.

Jurgens, D. A. et al. (2012). Semeval-2012 task 2: Measuring degrees of relational similarity. In: *Proceedings of the 1st Joint Conference on Lexical and Computational Semantics*. Montreal, Canada, pp. 356–364.

Landauer, T. K. and S. T. Dumais (1997). A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge. In: *Psychological Review* 104.2, pp. 211–240.

Leeuwenberg, A. et al. (2016). A Minimally Supervised Approach for Synonym Extraction with Word Embeddings. In: *The Prague Bulletin of Mathematical Linguistics* 105, pp. 111–142.

Levy, O. and Y. Goldberg (2014). Linguistic Regularities in Sparse and Explicit Word Representations. In: *Proceedings of the 18th Conference on Computational Natural Language Learning*. Baltimore, MD, USA, pp. 171–180.

Levy, O., Y. Goldberg and I. Dagan (2015). Improving Distributional Similarity with Lessons Learned from Word Embeddings. In: *Transactions of the Association for Computational Linguistics* 3, pp. 211–225.

Løvold, H. H. (2017). Tuning Word Embeddings for Neural Dependency Parsing of Norwegian. MA thesis. University of Oslo.

Mikolov, T., K. Chen et al. (2013). Efficient Estimation of Word Representations in Vector Space. In: *arXiv preprint arXiv:1301.3781*.

Mikolov, T., W.-t. Yih and G. Zweig (2013). Linguistic Regularities in Continuous Space Word Representations. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, GA, USA, pp. 746–751.

Miller, G. A. (1995). WordNet: A Lexical Database for English. In: *Communications of the Association for Computing Machinery* 38.11, pp. 39–41.

Øvrelid, L. and P. Hohle (2016). Norwegian Universal Dependencies. In: *Proceedings of the 10th International Conference on Language Resources and Evaluation*. Portorož, Slovenia, pp. 1579–1585.

Pennington, J., R. Socher and C. D. Manning (2014). GloVe: Global Vectors for Word Representation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Doha, Qatar, pp. 1532–1543.

Řehůřek, R. and P. Sojka (2010). Software Framework for Topic Modelling with Large Corpora. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta, pp. 45–50.

Schütze, H. and J. Pedersen (1993). A Vector Model for Syntagmatic and Paradigmatic Relatedness. In: *Proceedings of the 9th Annual Conference of the UW Centre for the New OED and Text Research*. Oxford, England, pp. 104–113.

Straka, M., J. Hajič and J. Straková (2016). UDPipe: Trainable Pipeline for Processing CoNLL-U Files Performing Tokenization, Morphological Analysis, POS Tagging and Parsing. In: *Proceedings of the 10th International Conference on Language Resources and Evaluation*. Portorož, Slovenia.

Straka, M. and J. Straková (2017). Tokenizing, POS Tagging, Lemmatizing and Parsing UD 2.0 with UDPipe. In: *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Vancouver, Canada, pp. 88–99.

Velldal, E., L. Øvrelid and P. Hohle (2017). Joint UD Parsing of Norwegian Bokmål and Nynorsk. In: *Proceedings of the 11th Nordic Conference of Computational Linguistics*. Gothenburg, Sweden, pp. 1–10.

Wu, H. and M. Zhou (2003). Optimizing Synonym Extraction Using Monolingual and Bilingual Resources. In: *Proceedings of the 2nd International Workshop on Paraphrasing*. Stroudsburg, PA, USA, pp. 72–79.

Zeman, D. et al. (2017). 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*. Vancouver, Canada, pp. 1–19.