NTNU
Norwegian University of Science and Technology

Cross-lingual information retrieval using compound word splitting

Martin Nordal

Master of Science in Computer Science
Submission date: July 2016
Supervisor: Björn Gambäck, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Martin Nordal

Cross-lingual information retrieval using compound word splitting

Master of Science in Computer Science, Spring 2016

Abstract

Finding web pages written in a foreign language may be a difficult process when using online search providers. This is because the information needed normally has to be formulated in the same language. The field of cross-lingual information retrieval seeks to ease this challenge by handling the gap between information in one language and information demand in another. Literature were found to introduce means to deal with different aspects of this language gap, and one particular framework was found to combine some of these. This thesis focuses on building a framework to solve other aspects that are demanding.

The framework implemented comprises query translation and document retrieval. Particularly the handling of compound words is analysed to improve query translation. The approach to compound word splitting attempts at being language independent and combines the word length feature with usage of a training corpus. Experiments were conducted to evaluate the splitting of compound words both separately in Norwegian and in context of cross-lingual document retrieval with translations to English, Spanish and German.

Experiments found that taking the word length feature into account improved an otherwise purely statistical approach to compound word splitting. It was also found that compound word splitting could be avoided in some cases when used in cross-lingual document retrieval. Various improvements are possible, such as better tuning of the compound word splitter, detection of rare cases in which compounds are formed, or to deal with the problems that occur in document retrieval when splitting up a compound word.

Sammendrag

Å finne nettsider skrevet på fremmedspråk kan være en vanskelig prosess når man bruker søketilbydere på nett. Dette er fordi den etterspurte informasjonen normalt må formuleres på det samme språket. Flerspråklig informasjonsgjenfinning er et forskningsfelt som går ut på å lette denne utfordringen ved å håndtere skillet mellom informasjon på et språk og informasjonsetterspørsel på et annet. Litteratur viste seg å introdusere midler for å ta hånd om forskjellige aspekter ved dette språkskillet, og spesielt ett rammeverk hadde kombinert noen av disse. Denne oppgaven fokuserer på å bygge et rammeverk for å løse andre aspekter som er krevende.

Det implementerte rammeverket omfatter spørringsoversetting og dokumentgjenfinning. Spesielt ble håndtering av sammensatte ord analysert for å forbedre spørringsoversetting. Framgangsmåten for å dele opp sammensatte ord er ment å være språkuavhengig og kombinerer ordlengde som egenskap med bruken av et treningskorpus. Eksperimenter ble gjennomført for å evaluere oppdeling av sammensatte ord både separat på norsk og i sammenheng med flerspråklig dokumentgjenfinning med oversetting til engelsk, spansk og tysk.

Eksperimenter viste at å ta ordlengde som egenskap i betraktning forbedret en ellers rent statistisk basert tilnærming til oppdeling av sammensatte ord. Det viste seg dessuten at oppdeling av sammensatte ord kunne vært unngått i noen tilfeller hvor det ble brukt i flerspråklig dokumentgjenfinning. Forskjellige forbedringer er mulige, som for eksempel å bedre finjustere oppdelingen av sammensatte ord, gjenkjenning av sjeldne tilfeller for hvordan ord er satt sammen, eller å ta hånd om de problemer som følger i dokumentgjenfinning når sammensatte ord har blitt oppdelt.

Preface

This Master's thesis was written by Martin Nordal during the spring of 2016 and is the final delivery for the Master of Science (MSc) degree in Computer Science at the Department of Computer and Information Science at the Norwegian University of Science and Technology (NTNU). Some of the background theory is based on a previous project that was conducted during the autumn of 2015 [Martin Nordal, 2015]. The idea of this assignment is based on a previous thesis written by Neergaard [2012].

Martin Nordal
Trondheim, 12th July 2016

Acknowledgements

First of all I would like to thank my supervisor Björn Gambäck for all the help and guidance he has given me, and Lars Bungum for providing resources. I would also like to thank my wonderful girlfriend Vilde Haug Almestrand for helping me with the language of this paper, and keeping me motivated.

Contents

List of Figures

List of Tables

1. Introduction

The Internet has become an important resource for information for people around the world. Today more and more people gain easier access to the Internet, and the amount of contents is increasing as well. Websites appear in a variety of languages, about half of them are currently presented in English [Q-Success]. Whilst roughly one quarter of the Internet users have English as mother tongue [Miniwatts Marketing Group]. With the diversity of contents and languages, it can be problematic for a single person to find all available information for a given subject.

The activity of information retrieval (IR) is to obtain specific information from a set of data. Online search engines do this when they respond to users' information needs, and to them the Internet is the data set. In a typical IR setting one has to formulate a search query consisting of some central keywords. However, this can be difficult to accomplish in a foreign language. In such cases it could be advantageous to the user if the query was automatically translated from the user's mother tongue to the language in which the desired documents were written. This challenge is one of the foundations to the field of cross-lingual information retrieval (CLIR).

1.1. Goals and Research Questions

Goal *This thesis aims at developing and testing a scalable framework for cross-lingual information retrieval, which supports multiple languages.*

Neergaard [2012] built such a framework that could translate a query, given its language, into one or several other languages. The translated queries would undergo normal information retrieval together with the original query to indicate the translation performance. Neergaard [2012] proved some of the challenges that arise with query translation, including inequalities in word inflections, consideration of synonyms and the handling of multi word expressions, compound words and named entities. The purpose of this thesis is to build a similar framework for performing CLIR, and to concentrate on some improvements that were proposed by Neergaard [2012].

The first improvement is to add flexibility to the framework by using language independent approaches, described further in Research Question 1 below. The second challenge is to improve translation capabilities by handling compound words, which is discussed in Research Question 2 below. Further improvements could be accomplished with named entity recognition, as stated in Research Question 3 below.

Research Question 1 *How to perform CLIR and be as little dependent on language specific resources as possible?*

The prosperity of a tool relies on its flexibility. In association with a CLIR framework, it would be ideal to handle any language. In this thesis, four languages will be used as examples, where the resources already exists. A sub goal is to implement methods that could work on any language.

Research Question 2 *What is the best method to split compound words?*

Compound words normally occur in corpora, but can not be expected to occur in dictionaries. Thus compound words may lead dictionary based translations to fail. The most common solution to this problem is to split these compound words, as will be a sub goal in this thesis. The solution will as well focus on a language independent approach according to Research Question 1.

Research Question 3 *What are named entities, and why do they have to be identified?*

Named entities bring complexity to the task of translation. Some names should pass unaltered, and others would have to be translated. To detect and group named entities is indeed challenging, as they occur rarely in dictionaries. A sub goal to answer this question is to look at previous work on the field of named entity recognition and to prove the importance of this field.

1.2. Research Method

A combination of methodologies will be used to achieve the goals of this thesis. Literature review will be essential to all of the research questions. Looking at previous studies is useful in order to know what has proven to work and not, and to give an idea of what remains to be tested. To answer Research Question 1 and 2 further, a framework will be designed, implemented and evaluated based on assumptions that are drawn from the literature review. The implementation concerning Research Question 2 will be subject to both qualitative and quantitative evaluation in order to give a brief idea of its performance. The overall framework will be subject to quantitative measurements. It was decided not to include any means to answer Research Question 3 in the architecture, as that was considered a too extensive topic for this thesis. Experiments will instead briefly identify why this issue is important.

1.3. Contributions

The experiments conducted in this thesis have shown the viability of applying compound word splitting in CLIR.

1. *Compound word splitting was based on a combination of the word length feature and statistical information from a training corpus.*
2. *Taking the word length feature into account performed better than the entirely statistical approach to compound word splitting.*
3. *Compound word splitting was applied in query translations for use in document retrieval.*
4. *In some cases, compound word splitting could have been avoided when performing query translations in document retrieval.*

1.4. Thesis Structure

Chapter 1 gives an introduction to the topic of this thesis and states the most important problems that are involved.

Chapter 2 will explain the vector space model in information retrieval and how it extends to a cross-lingual setting, as well as several concepts in natural language processing that can be used to support the vector space model.

Chapter 3 will shed light on more recent refinements within the same concepts as in Chapter 2.

Chapter 4 will combine some some of the findings from Chapter 3 to propose an architecture for performing query translations, mainly with focus on the handling of compound words.

Chapter 5 provides the plan to test the architecture from Chapter 4 as well as parameters used. It also includes results from these tests.

Chapter 6 gives an evaluation the results from Chapter 5 to summarise limitations, contributions and future work, as well as an overall conclusion.

2. Background Theory

The task of information retrieval (IR) is when a user queries some information. In most applications, this would involve some data to be indexed in advance so that certain information of interest is used to respond the query. A well known paradigm in the field of IR, is to represent the information algebraically in forms of vectors for further computation. This is called the *vector space model*, which we will look further into in Section 2.1. How to measure performance will be explained in Section 2.2. Various techniques for optimisation will be accounted for in Section 2.3. Section 2.4 will explain additional challenges that occur in a cross-lingual setting.

2.1. Vector space model

The purpose of the vector space model is to give text documents an algebraic representation. This model brings the opportunity to assign heuristic measurements to documents. More specifically, the goal is to represent documents and queries as vectors, to enable weighting and comparing. The first part of this transformation is tokenisation, which is to split a document into a list of its tokens, or terms if punctuations are discarded. This split is normally done at whitespaces, as terms are assumed to consist of single words, but Chapter 3 will elucidate why that method is inadequate. In the list that is obtained, some terms appear more often than others. The next part is to create a set containing these terms labelled with their frequency from the list. The set is unordered, and this document representation is referred to as the *bag of words* model.

Given bags of words from each document, it is possible to combine these into a common vocabulary. Let $\vec{T} = [t_1, t_2, ..., t_n]$ denote the vocabulary as a vector, that contains the terms used in all collected documents. Its number of dimensions is equal to the number of terms. Then for a document collection D using this vocabulary, a document can be denoted $\vec{d_j} = [w_{1,j}, w_{2,j}, ..., w_{n,j}]$.

2.1.1. Term weighting and vector normalisation

The value of $t_{i,j}$ in $\vec{d_j}$ is some weight that has been assigned to the term (by i) in the particular document (by j). The reason to weight a term is to indicate the importance of that term in a given context. A term weight can be any heuristic measure. For long documents, these weights would add up so that the vector increases in length. A document vector can, however, be normalised after weighting, such that $|\vec{d_j}| = 1$ There are several measures of term weighting. Probably the most common ones are *term frequency* and *inverse document frequency*, as we will look at individually and in combination.

2.1.2. Term frequency (TF)

The term frequency allows us to quantify the occurrences of a term in a document. Hence, it indicates how important a term is to a document. For a term t_i in document $\vec{d_j}$, the term frequency is denoted $tf_{i,j} = tf(t_i, \vec{d_j})$. The weight of this measure depends on the choice of function $tf(t_i, \vec{d_j})$. It can be binary, i.e., 1 if the term is present, or 0 otherwise. It can also be the number of occurrences, that may be given a logarithmic scale.

2.1.3. Inverse document frequency (IDF)

Let us first introduce the concept of document frequency, as it is basis for the inverted document frequency. The document frequency df_i of a term t_i in document collection D can be defined as the number of documents in which t_i occurs. It can be written as:

$$df_i = df(t_i, D) = |\{\vec{d_j} \in D : t_i \in \vec{d_j}\}| \tag{2.1}$$

The motivation for measuring document frequency is that terms occurring in many documents provide little information in a search context. Hence, the document frequency indicates how little important the term is. By inverting the document frequency, it will instead indicate how much important a term is. Zipf's law states that in natural language, a word occurs approximately twice as often as the next most frequent word [Baeza-Yates and Ribeiro-Neto, 2011, pp. 70-72]. So, if words are ranked by frequency, each word's rank is inversely proportional to its frequency. Thus the inverse document frequency will have a logarithmic scale:

$$idf_i = \log \frac{|D|}{df_i} = \log \frac{|D|}{|\{\vec{d_j} \in D : t_i \in \vec{d_j}\}|} \tag{2.2}$$

2.1.4. Term frequency and inverse document frequency (TF-IDF)

When combining term frequency with inverted document frequency, we can indicate how important a term is to a particular document compared to the overall collection of documents. This measure, shortened TF-IDF, will essentially indicate terms that occur often in few document [Spärck Jones, 1972]. TF-IDF is defined as:

$$tf - idf_{i,j} = tf - idf(t_i, \vec{d_j}) = tf(t_i, \vec{d_j}) \cdot idf(\vec{d_j}) \tag{2.3}$$

2.1.5. Cosine similarity

When a document, perhaps a query, is represented as a vector, that vector will have a length and direction based on its composition of terms. The vector appears to be multidimensional, expanding mostly in dimensions corresponding to terms that occur often in the original document. To measure the similarity of two documents, we want to measure the similarity between their vectors. Vectors with similar term frequency distributions will have similar directions, with a small angle between. The angle between two document vectors can thus be used to measure these documents' similarity. This is called the cosine similarity:

$$sim(\vec{d_j}, \vec{q}) = cos\theta = \frac{\vec{d_j} \cdot \vec{q}}{|\vec{d_j}||\vec{q}|} \tag{2.4}$$

If $\vec{d_j}$ and $\vec{q}$ are normalised, taking these two vectors' scalar product will be sufficient:

$$sim(\vec{d_j}, \vec{q}) = cos\theta = \vec{d_j} \cdot \vec{q} \tag{2.5}$$

Completely different documents with no common terms are orthogonal, since their vector components will point in different dimensions. Their cosine similarity will be 0.

2.2. Measuring performance

Measuring performance in context of information retrieval will require a test set for which each document is known to be relevant or nonrelevant [Manning et al., 2008, p. 140]. Relevance is dependent on a user's need, thus relevance is based on subjective evaluations. Documents may be retrieved or not regardless of their relevance. This results in four different outcomes for each document, as seen in Table 2.1. When counting the occurrences of each outcome, two measures can be obtained. Precision (P) is the number of relevant documents retrieved divided by the total number of retrieved documents. Recall (R) is the number of relevant documents retrieved divided by the total number of relevant documents.

$$P = \frac{tp}{tp + fp} \qquad R = \frac{tp}{tp + fn} \tag{2.6}$$

	relevant	nonrelevant
retrieved	true positives (tp)	false positives (fp)
not retrieved	false negatives (fn)	true negatives (tn)

Table 2.1.: Four different outcomes of document retrieval that can be used to measure performance.

To attain high precision, few non-relevant documents will have to be retrieved. At the same time, this increases the risk of having less relevant documents retrieved, affecting the recall. Precision and recall are generally incompatible measures because it is difficult to optimise for both of them at the same time. An alternative to precision and recall will be to look at the fraction of decisions made right during a document retrieval. The accuracy (A) is defined as:

$$A = \frac{tp + tn}{tp + fp + fn + tn} \tag{2.7}$$

In some situations it might be that either precision or recall is considered more important than the other. The F-measure is the weighted harmonic mean of precision and recall. It has the ability to take both into account, but weight one over another. F_β-measure and its balanced F_1-score are defined as [Manning et al., 2008, p. 144]:

$$F_\beta = (1 + \beta^2) \cdot \frac{P * R}{\beta^2 \cdot P + R} \qquad F_1 = 2 \cdot \frac{P * R}{P + R} \tag{2.8}$$

2.3. Natural language processing

The vector space model requires some steps of text preprocessing. The model has some limitations, or challenges, as some of them can be attenuated. The field of natural language processing has some methods that will help to ease the problems that occur when applying the vector space model.

2.3.1. Stemming and lemmatisation

Words appear with different grammatical properties, like singular and plural, or based on gender or grammatical cases. In the vector space model, terms are compared literally, not matching words in different forms. Such variations will appear as noise to this model.

A concept that is commonly used here is stemming. The rules by which stemming is performed depend on the language it is performed at. This is because each language has different grammatical rules and properties. Stemming is the removal of all affixes, which leaves us with the stem of a word. An example is the English word `nastiness`. When the suffix (a type of affix) `-iness` is removed, the word that remains is `nast`. We now have the stem of the word, which utters a meaning, that can be slightly different from the original word with the affix.

A slightly different alternative to the process of stemming is lemmatisation. That is to find a general and dictionary friendly form of a given word, something called a lemma. In the `nastiness` example, the lemma becomes `nasty`. Loponen and Järvelin [2010] built a dictionary and corpus independent lemmatiser, StaLe, that was intended for languages for which few resources existed. StaLe would learn its own lemmatisation rules from pairs of conjugated and non-conjugated words. Lemmatisation can be preferable to stemming in the context of information retrieval, as completely different words can have the same stem. Stemmers are, on the other hand, easier to make and should in theory be faster.

2.3.2. Stop word removal

Stop words can also be a problem when retrieving information. When comparing two documents one does not wish to stumble upon language specific words. The concept of stop words includes words that often occur in a language [Leskovec et al., 2014, pp. 7-9]. However, there are no automatic way to distinguish stop words. Yet, measures like IDF and TF-IDF would take stop words into account; stop words will generally add unnecessary complexity in the vector space model. Stop words can be removed categorically, but that can also be problematic, as stop words might be used to form named entities.

2.3.3. Part-of-speech tagging

When the bag-of-words model is applied, information concerning each term's context disappears, which results in loss of information. The idea behind part-of-speech (POS) tagging is to categorise terms, since equally written words can have different parts of speech. This enables labelling of terms before applying the bag-of-words model, in order to preserve some of the information regarding the terms' meaning. A part-of-speech tagger can be rule based [Brill, 1992] or based on a treebank [Brants, 2000].

2.3.4. Compound words

Compound words appear in several languages when words that can be found in dictionaries are combined into expressions that may not necessarily be found in dictionaries. Examples of English compound words are `printer cartridge`, `football` and `dry-cleaning`. These appear in the forms of open, closed, and hyphenated, respectively. For German and Norwegian, among other languages, open compounds are not formally allowed and are instead closed.

Morphological operations define how words are compounded. For example, the Norwegian word `tungvektsløfter` (heavyweight lifter) uses the epenthetic -s- between `tungvekt` and `løfter`, but not between `tung` and `vekt` [Johannessen and Hauglin, 1996]. Despite the formal incorrectness of using open compounds in Norwegian, they often appear from users not knowing this rule.

The fact that languages set different rules to how compound words are formed, combined with deviations from these rules, is particularly challenging in cross-lingual applications such as that of information retrieval. A universal rule is needed, and the perhaps simplest, to use open compounds only, is the most suitable to the vector space model. Thus the field of compound word splitting is to detect closed compounds and to split them.

2.3.5. Query expansion

A drawback that occurs with the vector space model is that queries have to contain the exact terms that one would expect to be finding in the retrieved documents. Synonyms or similar terms will not be considered, which lowers the recall. The method of query expansion is useful in such situations. When an initial query has been placed and some top-k scored results have been retrieved, that query can be expanded or altered to increase the recall in a second retrieval. Query expansions can be formed by explicit feedback from user or from implicit feedback through other analysi [Baeza-Yates and Ribeiro-Neto, 2011, pp. 177-200]. Expansion terms can also come from a thesaurus.

2.4. Cross-lingual information retrieval

The concept of cross-lingual information retrieval (CLIR) appears when a language barrier is added to normal information retrieval. Hence, it means that a query of one language is used to find documents in another. This will involve some sort of translation between the two languages.

2.4.1. Parallel corpora

A collection of documents that is represented in more than one language is referred to as a parallel corpus. If a set of languages $(L_1, L_2, ..., L_p)$ is added to the vector space model, we get a set of vocabularies $(\vec{W}_1, \vec{W}_2, ..., \vec{W}_k)$ as well. These vocabularies will not need to be equally large due to inequalities of languages. In a parallel corpus, each document vector $\vec{d_j,}k$ will exist for all languages involved. If the document vectors $\vec{d_1,}a$, $\vec{d_2,}a$ and $\vec{d_3,}a$ exist for language L_a, as an example, the corresponding document vectors $\vec{d_1,}b$, $\vec{d_2,}b$ and $\vec{d_3,}b$ would also exist for language L_b, as seen in Figure 2.1.

$$\begin{array}{llll} L_a: & \vec{d}_{1,a} & \vec{d}_{2,a} & \vec{d}_{3,a} \\ L_b: & \vec{d}_{1,b} & \vec{d}_{2,b} & \vec{d}_{3,b} \end{array}$$

Figure 2.1.: Document vectors $\vec{d}$ that appear in parallel across languages L.

Two main approaches to cross-lingual information retrieval would be to either translate every document in advance to build a parallel corpus or to apply query translation. McCarley [1999] found that both methods are equally precise, but that document translation is more costly, and thus query translation is more preferable. However, document translation can also perform slightly better than query translation, and a combination can result in even better performanc [Chen and Gey, 2003].

2.4.2. Translating queries

Translation is not trivial, as dictionaries may have several translations per term. If all available translations are used in a query translation, it results in an expanded query, and hence possibly increased recall with decreased precision. These are referred to as unbalanced queries, as weighting will favour terms with most translations. Then in order to balance a query, the weight of each term will have to be distributed between its translation [Levow and Oard, 2002].

An alternative to query balancing is pruning. Aljlayl et al. [2002] introduced reverse dictionary pruning which aims at keeping translations that translate back to the original language. Federico and Bertoldi [2002] introduced the N-reduction pruner that uses term frequencies in documents to estimate the N most likely translations.

2.4.3. Named entity recognition

When names occur in corpora, they should in most cases not be subject to translation. For example, if somebody named "June" is mentioned in a document, that name could potentially, but erroneously, be interpreted as a month, and then be translated. The field of named entity recognition seeks to identify such named entities. In order to do this, the challenge has for a long time been to overcome the need of domain knowledge. An approach has been to consider different word features in rule based classification [Nadeau and Sekine, 2007].

3. Related Work

3.1. Existing frameworks

The EXtensible Cross-Linguistic Automatic Information Machine (EXCLAIM) is a tool that uses Wikipedia as parallel corpus [Kirchner et al., 2009]. It is extensible to other types of corpora as well. There exist, however, just brief information about its behaviour, and no information regarding its performance was found.

CLIRch by Neergaard [2012] is an open source framework of great inspiration to this thesis. It aims at using state of the art refinements in natural language processing in order to translate a user query. Despite having all programs that CLIRch depends on, there is no manual on how to use the source code. Some important details on how to run the code were left to imagination, and the attempt to use and extend CLIRch never succeeded. Another drawback is that CLIRch uses some tools intended for specific languages. This thesis will instead concentrate on a framework that is independent from language specific resources.

3.2. Compound word splitting

Much of the research on compound word splitting has been aimed at the German language, as it does not have open compounds. With a rough ten percent occurrence of compound words [Johannessen and Hauglin, 1996], Norwegian is also demanding. Ranang [2010] built a compound word splitter for Norwegian that followed some morphological operations that occur when compounding Norwegian words. Koehn and Knight [2003] used term frequencies from a training corpus, combined with German morphological operations, to find the most suitable candidate split or to keep the original word where appropriate. Macherey et al. [2011] was able to learn morphological operations on a variety of languages by using a training corpus and by introducing split penalty in order to prevent eager splitting.

3.3. Part-of-speech tagging

Differences in grammatical properties that are available in each language means that languages will have different part-of-speech tagsets. This can be problematic in a cross-lingual environment. For example, existential there (as in *there is*) can have its own tag [Santorini, 1990]. Norwegian nynorsk would have the equivalent *der (er)*, but Norwegian bokmål would use the pronoun *det (er)*. When creating a universal tagset, distinctivenesses in each language are simplified tags that can relate to other languages [Petrov et al., 2011].

3.4. Named entity recognition

The term *named entity* first appeared at the MUC-6 conference which focused at information extraction from unstructured text [Grishman and Sundheim, 1996]. The first obvious challenge within named entity recognition (NER) was to give proper categories to names. In the following years experiments have shown that to have many categories is difficult to maintain [Sekine, 2004]. Yet, it is hard to omit domain knowledge at all. Some named entities are written in the same form, and to avoid sense disambiguation, domain knowledge is useful in combination with non-local features from external corpora [Ratinov and Roth, 2009]. This is one of the reasons why the emergence of Wikipedia has contributed greatly to the field of NER. Studies have shown that the semantics of Wikipedia, such as categorisations and linkage features across articles, are useful in NER applications [Cucerzan, 2007, Nothman et al., 2013].

Cucerzan [2007] used the term *surface form* as how an entity appears in written form and focused on solving the problem when having ambiguous surface forms. For example, the name `Texas` can refer to one of the states in the USA, but also other places, a TV series, films, a novel etc [Wikipedia, 2016]. Wikipedia will in these cases dedicate a page to the ambiguous surface form, in order to list the entities comprised by that surface form. Cucerzan [2007] stored information from Wikipedia into two databases: one lists the different surface forms with all associated entities, and the other lists all entities with tags associated. The procedure to disambiguate a named entity was first to see if Wikipedia contained an article with that exact name. Otherwise, the corresponding disambiguation article would have to be looked up, and the most commonly used entity was regarded as most likely.

4. Architecture

The objective for this thesis was to make a complete framework for conducting query translation based CLIR. This included document indexation, query translation and document retrieval for evaluation, as illustrated in Figure 4.1. In order to build a proper document index, a statistical compound word splitter had to be trained in advance. This resulted in a need of two corpora, but as we will see from the experiments in Chapter 5, one corpus was split in half.

The vector space model from Section 2.1 was essential to document indexation and retrieval, as will be explained in Section 4.3 and 4.5, respectively. Query translation was dictionary based and used various preprocessing in advance: such as lemmatisation, stop word removal and compound word splitting, as described in Section 4.2. The compound word splitting was attempted both by a statistical approach, in Section 4.1.2, and a splitter built specifically for Norwegian, in Section 4.1.1.

4.1. Compound word splitter

To give an answer to Research Question 2, two different compound word splitters were implemented and tested. A Norwegian splitter was inspired by Ranang [2010] and was used as a reference. In order to satisfy Research Question 1, the approach to implement a language independent compound word splitter was statistical. It was inspired by that of Macherey et al. [2011] by using term frequencies and split penalty to score candidate splits. It also used word length as a feature to both remove noise and as an additional ingredient in candidate split scoring. Macherey et al. [2011] denoted morphological operations as v/w where v in a closed compound is replaced by w in the corresponding open compound. An empty value for v or w can be denoted as ε. Then during training of the statistical splitter morphological operations were identified and counted. When using the splitter these operations were applied.

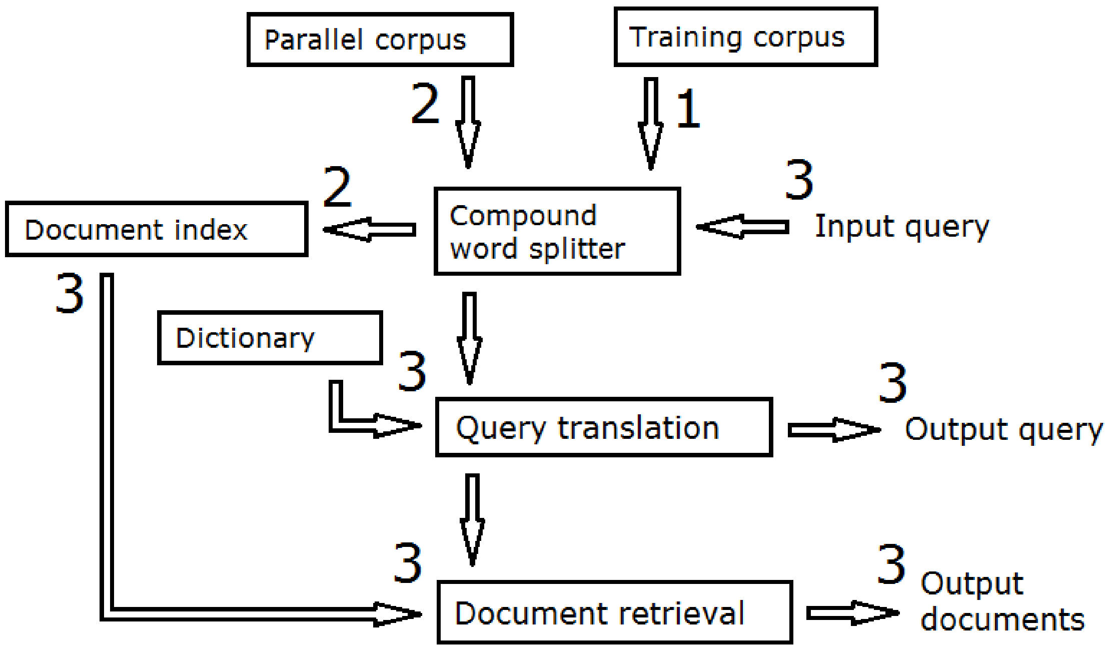

Figure 4.1.: The compound word splitter needs to (1) process the training corpus before (2) the parallel corpus can be indexed or (3) any queries can be translated in order to retrieve documents.

4.1.1. Norwegian splitter

The Norwegian splitter was an implementation of algorithms proposed by [Ranang, 2010, Chapter 3]. Taking a word as input, the splitter began to split this word in every possible way. This process also took into consideration that no consonant can repeat contiguously more than twice when compounding words, and that a possibly third occurrence would have to be added when decompounding. For example, the word `musikkorps` would then be split into the candidates (`musik`,`korps`), (`musikk`,`orps`) and (`musikk`,`korps`) etc. Instead of keeping and sorting all split candidates, some with certain characteristics were not used. First, split candidates of more than two parts where the parts also have an average length of less than three characters were assumed noise and were removed. Then, the splitter would remove all candidates that could be considered as complete words. Ranang [2010] used a slightly modified Early chart parser [Earley, 1970] together with a filter that "checks the root node to see whether the tree represents something that can be considered part of a word (the alternative would be a complete word)" [Ranang, 2010, p. 49]. The implementation in this framework would simply look for an epenthetic -s- or -e- in a part of word. A series of evaluations were then performed to score each split candidate in order to find the most likely.

4.1.2. Statistical splitter

The basis to make a statistical compound word splitter was first to build an index of term frequencies from a corpus. Term frequency in this context will be used as the number of occurrences in the entire corpus and not single documents. During experiments however, it turned out that looking up term frequencies was time consuming, specially for a large training corpus. Noise removal based on the word length feature was applied prior to using term frequencies to save time, and as we will see in Section 5.2.1, noise removal also helped to find better split candidates. As mentioned in Section 4.1.1, Ranang [2010] defined noise as when the average word length in a candidate split was less than three characters. This number was believed to prove successful mainly for use in Norwegian. In the statistical splitter, the definition of noise was implemented as when the average word length in a candidate split was less than the square root of the average word length of the entire training corpus.

Scoring of candidate splits

The statistical splitter began likewise as the Norwegian splitter, to split a word in every possible way, and then to discard noise. Term frequencies were then used in order to score each candidate split. It was also assumed that the length of a word should not differ significantly from the average word length. A high term frequency would be desirable, but an abnormal word length could allude either a compound word or eager splitting. For each input word, itself and all of its split candidates were given the heuristic score:

$$S(u) = -kn + \sum_{i=1}^{n+1} (\log(c + \mathit{freq}(u_i)) - \log(e + r(\mathit{length}(u_i) - \mathit{avg\_length}))) \tag{4.1}$$

Here n is the number of split points, $n+1$ is the number of parts in the candidate split, k is the split penalty constant and $freq(u_i)$ is the term frequency of the part u_i in the candidate split u. The logarithm gives a penalty for each part of zero term frequency in a split. However, a small constant $c << 1$ was added, so that the penalty was not infinitely. We see from this equation that no penalty would be given to an unbroken term, unless that term itself has zero term frequency or is of abnormal length. The candidate split with highest score was considered most likely. The constant r penalises word compounds that differ much from the average word length. It was set to 0.25 without any formal investigations.

Obtaining morphological operations

When the compound word splitter had come to a decision on how to split a word, each part of that split was looked up in a dictionary. Parts that were not found in the dictionary were immediately considered as linking morphemes. A word cannot start or end with a linking morpheme, and in such cases the split would be discarded. Then, all possible morphological operations would be identified by using superposition.

Linking morphemes were removed from the split, one at a time. The remaining parts were merged back together and compared to the original compound word, which indicates a morphological operation. Hence, noise was expected to occur, and some heuristic was needed in order to score each finding. By using the split candidate score from Equation 4.1, the utility from removing each linking morpheme was measured:

$$\delta S(m) = S(b) - S(a) \tag{4.2}$$

Here m would be the morphological operation, a is the complete split, and b is without the actual linking morpheme. This measure would penalise the use of b if the suspected linking morpheme turned out to be a highly frequent term. Finally, each utility $\delta S(m)$ was added to a global score $\sum \delta S(m)$ of its corresponding morphological operation. The resulting list of morphological operations would thus include all findings, and the scoring is intended to distinguish between frequent linking morphemes and those who appear to be noise.

Making use of linking morphemes

A morphological operation that appears to occur often does not guarantee that it is valid, it might as well come from a common mistake. Frequent terms or named entities that are not found in dictionaries will be mistreated as linking morphemes. This is when human intervention is desirable in order to remove what seems as obvious errors that might affect further performance significantly. The guideline was to remove morphological operations that involved common terms as linking morphemes.

When the statistical compound word splitter was used, all candidates to split a word were once again obtained. For each candidate split, all of the obtained morphological operations were applied if possible ordered by their global scores. This would remove or replace a linking morpheme where appropriate. All candidate splits were scored as in Equation 4.1, and this score would favourise candidate splits where linking morphemes had been removed. The candidate split with highest score was considered most likely and should not contain any linking morpheme.

4.2. Preprocessing

The preprocessor as described in this section was used both when indexing documents as well as prior to query translations. Queries were tokenised simply by splitting at white spaces, and the actual corpora was found to be tokenised already. Non-letter or non-number characters were removed, accents on letters were kept. Hence stop word removal was done by using lists from the Snowball stemme [Porter, 2001]. Then each token went through the compound word splitter.

The next step of preprocessing was lemmatisation and part-of-speech tagging performed by the TreeTagger [Schmid, 1994]. However, the TreeTagger has not yet been trained for the Norwegian language, and in that particular case the Oslo-Bergen Tagger [Johannessen et al., 2011] was used. The resulting tagsets varies depending on the languages, and these would have to be replaced into a common tagset. Table A.1, A.2, A.3 and A.4 in Appendix A show how these tagsets was united following the work of Petrov et al. [2011]. The result is a list of terms labelled with their part-of-speech.

4.3. Document indexation

The framework was designed to read any corpus taken from the Opus parallel corpora project [Tiedemann, 2012]. The terms from each document had to be put into a database following the vector space model as described in Chapter 2. This model was slightly extended, as terms were represented by their lemma and POS. Thus only terms of the same POS would match, but grammatical inflections would not infer. The title and language of all documents were stored in a separate table in the database. Another table represented the grammar, which in this case was all unique combinations of word lemmas, POS and language. One last table linked the other two in a many-to-many relation and kept track of term frequencies between documents and terms.

4.4. Query translation

In order to translate queries, machine readable dictionaries were used, and it was attempted to make use of part-of-speech (POS) tagging from the dictionaries. When looking up a word, the list of results would always begin with translations of identical POS. Candidates of other POS would be appended to the results. Both reverse pruning and query balancing were implemented for comparison.

Reverse pruning When translations of a word had been retrieved from a dictionary, each translation was only accepted if it translated back to the original language. Translations of equal POS as for the original word were preferred, and if not obtained, any POS would be accepted.

Query balancing The term frequency of a word would be divided equally among its translations. However, translations of any POS were only accepted if no translation of the same POS as the original had been obtained.

Regardless of reverse pruning or query balancing, a word would pass unaltered if it had no translations at all. This was an effort to maintain recall in case of an unidentified named entity.

4.5. Document retrieval

The process of document retrieval was based upon computing the cosine similarity from Equation 2.5, comparing each indexed document with a query. This was done simply by using the database table that linked documents and terms, as described in Section 4.3. Two documents would however not be comparable if they appeared in different languages. The equality of two terms depends on both their lemma and POS. Documents were returned if their cosine similarity to a query were above zero, and thus at least one term would have to match, although the score is not kept for any other purpose.

5. Experiments and Results

The purpose of this chapter is to provide details on how the framework presented in Chapter 4 was tested, as well as results from these tests. First, the statistical compound word splitter will need some corpus to train on. How large this corpus needs to be, and an optimal split penalty, are to be discussed in Section 5.1.3. It takes some knowledge about a language to evaluate whether the statistical splitter succeeds. This is why this splitter will only be evaluated in Norwegian. When the compound word splitter is configured, it must be applied to the parallel corpus. The Norwegian compound word splitter will be applied to the parallel corpus separately. Then a document retrieval, as described in Section 5.1.4, will be able to prove whether these approaches are useful by comparing their performance with that involving the original parallel corpus. The document retrieval will also have alternative configurations to uncover some other factors that take place in a query translation.

5.1. Experimental Setup

5.1.1. Dictionaries

All dictionaries were provided by the Norwegian Kunnskapsforlaget[1] via the Norwegian University of Science and Technology. These dictionaries translated from Norwegian to English, German and Spanish, and vice versa. Thus query translations were not performed between English, German and Spanish. Translations between English, German and Spanish would have to be performed through Norwegian, but it was not part of any experiments. As seen in Table 5.1, most dictionaries in these experiments have typically one and a half translations per word.

[1] http://www.kunnskapsforlaget.no/

From	To	No. of words	No. of translations	Translations/word
English	Norwegian	31 783	51 085	1.61
German	Norwegian	27 531	39 156	1.42
Spanish	Norwegian	23 935	55 463	2.32
Norwegian	English	27 632	40 237	1.45
Norwegian	German	26 460	35 469	1.34
Norwegian	Spanish	24 722	39 140	1.58

Table 5.1.: How many words the different dictionaries translated to.

5.1.2. Corpus

The framework was tested with the OpenSubtitles2016 collection [Lison and Tiedemann, 2016] that had been provided by Opus [Tiedemann, 2012]. The collection consists of subtitles for various films in a wide assortment of genres over a wide range of years. Each document contains subtitles for a particular film, and several, perhaps none, documents may represent a single combination of film and subtitle language. This corpus was found to contain many misspellings where two similarly looking letters had been mistaken. A common error was when I (uppercase i) occurred instead of l (lowercase L), for example as `paraIIeI` instead of `parallel`.

The OpenSubtitles2016 corpus was split in two: one small parallel corpus for use in information retrieval and one larger portion to train the statistical compound word splitter. It follows from the previous section that, due to the dictionaries provided, all translations would have to include Norwegian. Thus, any document that was not or could not translate to Norwegian could have been discarded as parallel corpus, but that would lead to an unnecessarily large parallel corpus and no training corpus for Norwegian. Instead, for all documents that were not present in all of Norwegian, English, German and Spanish, the decision was to discard these as parallel corpus. They would instead serve as training corpus to the statistical compound word splitter. This still did not lead to a desirable size of training corpus for the compound word splitter, and the parallel corpus was also capped to documents that represented 500 film titles per language, the rest were added to the training corpus. Yet, the number of documents per film varied from one language to another. Table 5.2 show this by the notable variation in the number of documents per language for the parallel corpus, albeit the word counts per document are somewhat comparable. The resulting training corpus for Norwegian consisted of 6 696 documents with 295 761 unique words and a total word count of 16 092 719.

A concern during these experiments was that the OpenSubtitles2016 collection alone would not serve as a sufficient training corpus to the statistical compound word splitter. To see if this was the case, a portion of the noTenTen corpus Jakubíček et al. [2013] was added. The noTenTen corpus is a collection of news articles extracted from online Norwegian newspapers. The first 1 536 000 documents of the noTenTen corpus were included in the training corpus, which in addition to the OpenSubtitles2016, raised the unique word count to 4 808 241 and the total word count to 352 904 803.

Language	Documents	Unique words	Word count	Word count/ unique word	Word count /document
Norwegian	628	66 415	1 762 383	26.54	2 806.34
English	5 593	72 115	23 422 938	325.08	4 187.90
German	938	94 104	2 941 287	31.26	3 135.70
Spanish	3 464	113 083	12 043 788	106.50	3 476.84

Table 5.2.: The size of parallel corpus divided by language.

5.1.3. Training the compound word splitter

Training of the compound word splitter was performed in two steps. First a suitable split penalty had to be obtained. Then, using that penalty as parameter, morphological operations were ranked globally based on a portion of the training corpus.

Obtaining an optimal split penalty

The split penalty constant k from Equation 4.1 reflects how likely it is for a word to be split, or how many times the word will be split. An optimal value of k is believed to depend on the language, but also on the size of training corpus that is used, as well as if word length penalty is included or not. Macherey et al. [2011] used a rather small split penalty for German compared to other languages, as the German corpus stood out as the smallest.

In order to find an optimal split penalty k, the accuracy of the statistical compound word splitter was considered. A set of random words were taken from the training corpus, of which each compound word was labelled with a correct split candidate. To not split was often the correct case as well. Words that were misspelled, foreign or named entities were however discarded from this set. The compound word splitter would revise each word as described in Section 4.1.2, and the resulting accuracy would be the fraction of correct outcomes and the total input words. The challenge would be to find a value of k that maximised the accuracy. It could have been ideal to use simulated annealing, if the compound word splitter had been less costly to run. Instead, an initial split penalty was set, and the finite difference method with decreasing steps $h_{n+1} = 0.75 \cdot n$ was used to iteratively find a final split penalty. This was under the assumption that no local maxima existed.

Using morphological operations

To obtain morphological operations that form compound words, a random selection of words from the training corpus were revised as described in Section 4.1.2. From each word that was found to be compounded, possible morphological operations were extracted and globally scored. The list of globally ranked morphological operations was then revised to have noise removed.

5.1.4. Document retrieval

The architectural description from Chapter 4 includes several ingredients to perform query translation. To see how useful different components were, they had to be tested individually. The framework was tested according to Table 5.3, which indicates that the usage of part-of-speech tagging, different compound splitting and different pruning techniques were varied to see their individual contributions to the framework's performance. Each configuration were given a name based on these parameters, as seen in the first column of Table 5.3.

Document retrieval was conducted using the queries listed in Table 5.4. These queries were used in pairs, such that each query had an equivalent in another language. The document retrieval would try to determine if the query translation is more successful from one language to another. The first configuration in Table 5.3 is the only not including any translation, it will instead show the performance when using a query directly in its language. This configuration only included lemmatisation and stop word removal, as this had also been applied to the parallel corpus.

A common feature of Q_1, Q_2 and Q_3 was that their Norwegian representations were all compound words, unlike all of their non-Norwegian representations. These query pairs were also formed under the constraint that dictionaries were able to translate between the languages involved. This was intended as a pitfall when compound word splitting in Norwegian, to see if correct translations would be missed and replaced by words of other meanings. Q_4, Q_5 and Q_6 all contained compound words in both Norwegian and English, in the forms of closed and open, respectively. These query pairs were made to prove the necessity of compound word splitting in Norwegian when translating to English, and to see how these queries translated from English to Norwegian. In addition Q_5 and Q_6 contained named entities, one of which was not subject to translation, and another that was not present in the actual dictionaries.

It turned out that to determine relevant documents for a query was not trivial. As mentioned in 5.1.2, the part of OpenSubtitles2016 that was used as parallel corpus had several hundreds or thousands of documents, depending on the language. Documents were already titled with a unique number, but did not contain any descriptive title. The best solution to make queries with corresponding relevant documents was to start by looking up phrases in Norwegian. Apart from being the smallest language in the parallel corpus in terms of document count, all translations also had to include Norwegian. Given documents that were found in Norwegian, their titles (unique numbers) were ordered so that corresponding documents could be found in other languages. These were assumed to be equally relevant.

Configuration	Lemmatiser/POS tagger	POS used	Compound splitter	Pruning
reference	TreeTagger/OBT	n/a	n/a	n/a
rev	TreeTagger/OBT	no	none	reverse
bal	TreeTagger/OBT	no	none	balancing
pos-rev	TreeTagger/OBT	yes	none	reverse
pos-bal	TreeTagger/OBT	yes	none	balancing
cws-s-rev	TreeTagger/OBT	no	Statistical	reverse
cws-s-bal	TreeTagger/OBT	no	Statistical	balancing
cws-s-pos-rev	TreeTagger/OBT	yes	Statistical	reverse
cws-s-pos-bal	TreeTagger/OBT	yes	Statistical	balancing
cws-n-rev	TreeTagger/OBT	no	Norwegian	reverse
cws-n-bal	TreeTagger/OBT	no	Norwegian	balancing
cws-n-pos-rev	TreeTagger/OBT	yes	Norwegian	reverse
cws-n-pos-bal	TreeTagger/OBT	yes	Norwegian	balancing

Table 5.3.: Different configurations of document retrieval that were tested. The reference configuration was applied without query translation.

Name	Query a	Language a	Query b	Language b
Q_1	*skrunøkkel*	Norwegian	*spanner*	English
Q_2	*kattunge*	Norwegian	*Kätzchen*	German
Q_3	*stormannsgal*	Norwegian	*megalómano*	Spanish
Q_4	*lånehai*	Norwegian	*loan shark*	English
Q_5	*Poirot samfunnsparasitt*	Norwegian	*Poirot society parasite*	English
Q_6	*Themsen klokkefabrikk*	Norwegian	*Thames clock factory*	English

Table 5.4.: Different pairs of queries that were subject to translation and applied to the document retrieval.

5.2. Experimental Results

5.2.1. Compound word splitting

Optimisation

In an extraction of 200 words from the training corpus, as many as 153 appeared to be compound words[2]. A compound word splitter should then score an accuracy above the baseline of 0.235 to indicate some level of usefulness. As a reference, the Norwegian splitter scored an accuracy of 0.210 on the same test. The size of training corpus was expected to affect both the accuracy and optimal split penalty of the statistical splitter. Table 5.5 shows how the accuracy as well as optimal split penalty increased when the noTenTen corpus was added. The results in Table 5.5 also states that noise removal and word length penalty improved the accuracy, both individually, but most in combination. Another advantage of noise removal was that the experiments involved appeared to be much faster than the others.

When applying both noise removal and word length penalty, optimisation involving both OpenSubtitles2016 and noTenTen was found to last about 30 times longer than with OpenSubtitles2016 only. Table 5.5 clearly indicate that using both corpora is a little advantageous with respect to accuracy. From the baseline accuracy of 0.235, the best accuracy comprising both OpenSubtitles2016 and noTenTen was 9.72% higher than the best accuracy using OpenSubtitles2016 only. However, taking time into account, using only OpenSubtitles2016 as training corpus was considered the most valuable. The best optimisation result based on OpenSubtitles2016 exclusively with an accuracy of 0.595 can be seen in Figure 5.1.

Morphological operations

Morphological operations for word decompounding were found in a random selection of 100 000 words from the training corpus. Table 5.6 shows the top ten most highly ranked operations after they were sorted by their respective global utility $\sum \delta S(m)$ from Equation 4.2. The operations s/ε and e/ε were prominent, which is suitable for Norwegian. However, it was not trivial to distinguish correct and incorrect findings from only the numbers in Table 5.6. The training corpus was inspected to see that the single letters `s`, `l`, `r`, `t`, and `e` occurred 6 252, 6 111, 1 834, 3 383 and 124 times, respectively. Why this is challenging is to be discussed in Section 6.1.1. The common words `te` (tea) and `le` (to laugh) were were found in dictionaries and are clearly not linking morphemes in Norwegian. Only the morphological operations s/ε and e/ε were used when applying the statistical compound word splitter on the parallel corpus.

[2]This was after misspelled, foreign and words of named entities were removed.

Word count	From corpus	Noise removal	Word length penalty	Split penalty	Accuracy
16 092 719	OS16	no	no	11.335	0.565
16 092 719	OS16	yes	no	11.335	0.575
16 092 719	OS16	no	yes	9.688	0.580
16 092 719	OS16	yes	yes	9.688	0.595
352 904 803	OS16+noTT	no	no	14.665	0.610
352 904 803	OS16+noTT	yes	no	14.489	0.615
352 904 803	OS16+noTT	no	yes	12.970	0.620
352 904 803	OS16+noTT	yes	yes	12.947	0.630

Table 5.5.: The number of words occurring in a training corpus affects the optimal split penalty and its corresponding accuracy. The small corpus consists of OpenSubtitles2016 (OS16) alone, and the large corpus is an extension with the noTenTen (noTT) corpus. Noise removal and taking the word length feature into account also affects the accuracy.

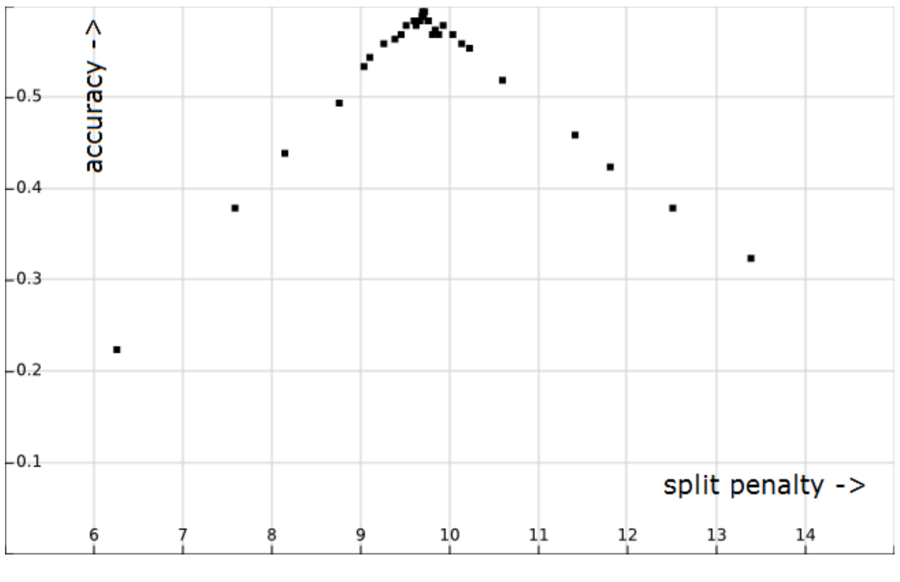

Figure 5.1.: The relation between split penalty and accuracy of the statistical splitter.

Operation (m)	Occurrences	$\sum \delta S(m)$	Average $\delta S(m)$
s/ε	3682	9253.6	2.5132
l/ε	725	1838.6	2.5360
r/ε	274	1024.6	3.7396
t/ε	238	747.48	3.1407
e/ε	114	733.42	6.4335
te/ε	133	463.87	3.4877
k/ε	78	317.93	4.0760
an/ε	113	311.96	2.7607
le/ε	88	240.12	2.7286
b/ε	67	237.00	3.5374

Table 5.6.: Based on a random selection of words from the training corpus, these morphological operations were found to be the top 10 most likely involved in Norwegian word decompounding.

Applying the compound word splitters

As mentioned in Section 5.2.1, the statistical and the Norwegian approach to compound word splitting scored differently in terms of accuracy. First of all, the Norwegian splitter would in most cases leave a compound word unaltered, but in some cases words were split erroneously. In the statistical approach, these two errors appeared to occur equally often. As an example, the word `forbrenningsmotor` (combustion engine) was not split because there was not a statistical basis apparent enough to conclude that `forbrenning` and `motor` makes a valid split. On the other hand, the word `fatle` (sling), was split into `fat` (barrel) and `le` (to laugh) because these words appeared more often in the training corpus.

When the statistical compound word splitter were applied on the parallel corpus, it resulted in an increase of word count to 4 077 948 and a decrease of unique words to 37 143. The Norwegian splitter resulted in 67 532 unique words and a word count of 1 913 251.

5.2.2. Document retrieval

Results from applying the queries Q_1 to Q_6 can be seen in Appendix B as Table B.1 to Table B.6, respectively. These tables includes results of all configurations from Table 5.3 and includes total retrieved documents, precision, recall and F_1-score. As this evaluation included more than just one pair of languages, it was not suitable to summarise these results in one table. Instead, the findings of interest will be commented briefly in this section.

Query pair Q_1

The Norwegian `skrunøkkel` had several translations to English, and `spanner` of same part-of-speech was desired. This is why reverse pruning in this case outperformed query balancing in terms of precision, without loss of recall, as displayed in Table B.1. Considering part-of-speech is slightly advantageous. Only the statistical splitter was able to split `skrunøkkel` into `skru` and `nøkkel`. Translated to `screw` and `key` (English), this lead to a considerable amount of false positives and no recall. Translating from English to Norwegian had no problems, apart from when retrieving in the statistically splitted corpus.

Query pair Q_2

According to both taggers and dictionaries that were involved, the Norwegian *kattunge* (kitten) was found to be a noun, whereas the German `Kätzchen` appeared to be a pronoun. When considering this difference in part-of-speech, no translation was found, as seen in Table B.2. Only the statistical splitter was able to split `kattunge`, which resulted in `katt` and `ung`, after lemmatisation. Translated to `jung` and `katze` (German), this lead to a considerable amount of false positives. When translating from German to Norwegian, `kattunge` was not found in the statistically splitted corpus.

Query pair Q_3

In this case the Spanish `megalómano` (noun) translates to `stormannsgal` (adjective) in Norwegian. The opposite dictionary translation yields `person som lider av stormannsgalskap` (person suffering from megalomania). As seen in Table B.3, this generally results in false positives when translating from Norwegian to Spanish. Only the statistical splitter was able to split `stormannsgalskap`, which resulted in `stor` (big), `mann` (man), `gal` (crazy) and `skap` (cupboard), each of which has numerous translations to Spanish, and only one of them relates to the word `megalómano`. Thus, the perfect recall is considered luck and is reflected in the low precision.

Query pair Q_4

None of the two phrases `lånehai` (Norwegian) and `loan shark` (English) were found in the dictionaries. Only the statistical splitter was able to split `lånehai`, which resulted in `lån` and `hai`, after lemmatisation. We see in Table B.4 that translation from Norwegian to English only succeeds inn terms of recall when applying the statistical compound word splitter, albeit it includes a considerable amount of false positives. Translation from English to Norwegian yielded impeccable recall, without severe aggravation of precision, when statistical compound word splitting was involved. Matching part-of-speech generally improved the precision.

Query pair Q_5

Both queries in this Norwegian-English pair contained the named entity `Poirot`, a person's name that was not to be translated. This term passed unaltered through query translation in both directions because it was not present in the dictionary. We see in Table B.5 that this resulted in good recall. The exception was when considering part-of-speech in the Norwegian query, the only case in which the Oslo-Bergen Tagger failed to recognise `Poirot` as a noun. The terms `society` and `parasite` yielded several translations to Norwegian, which contributed to false positives. The term `samfunnsparasitt` had no translation to English, but was splitted into `samfunn` and `parasitt` by the statistical splitter. These parts had numerous translations to English which improved recall, but with many false positives.

Query pair Q_6

This example includes the named entity `Thames`, which is `Themsen` in Norwegian. These terms were not found in the dictionaries, hence they were not translated, and thus they did not have any influence on the document retrieval. Neither were the two phrases `klokkefabrikk` (Norwegian) and `clock factory` (English) found in the dictionaries. Without compound word splitting, no translation was made from Norwegian to English. Only the statistical splitter was able to split `klokkefabrikk` into `klokke` and `fabrikk`. We see in Table B.6 that this generally results in a good recall, albeit having many false positives. The English words `clock` and `factory` yielded several translations to Norwegian, of which none relevant seemed to translate back. However, involving the statistically splitted corpus, translation from English to Norwegian lead to impeccable recall with poor precision. Other translations present in the balanced queries appeared to match the relevant documents.

6. Discussion and Conclusion

After seeing some results from Chapter 5, it should be possible to answer Research Questions 1 and 2 from Chapter 1. It is also desirable to mention if these results infer with the findings regarding Research Question 3.

6.1. Discussion

The results from Chapter 5, their implications and limitations will be discussed in this section. First the performance of compound word splitting alone will be discussed, and then the overall document retrieval.

6.1.1. Compound word splitting

The statistical splitter

As seen in Table 5.5, optimisation of the statistical splitter resulted in accuracies of around 0.6 [1] depending on the different parameters. Noise were suspected to affect the statistical compound word splitter, and particularly two types were found when using the OpenSubtitles2016 corpus:

1. As mentioned in Section 5.1.2, misspellings often occurred by mistaking two similarly looking letters. The extent of these misspellings was not clear, albeit these misspellings were not assumed to provide much useful information to the statistical splitter.

2. There was found a large amount of certain single letters, as stated in Section 5.2.1. It was unclear if this originated from the preprocessing which removed non-letters and non-numbers. The large count of these single letters were problematic to the statistical splitter because they compromised the use of term frequency. These frequency of letters could dictate the scoring and choice of candidate splits and hence quantify erroneous linking morphemes.

We see from Table 5.5 that a larger training corpus provided a better basis to the statistical compound word splitter measured by accuracy. Taking the word length feature into account also contributed notably. The best accuracy comprising the large training corpus was 9.72% higher than the best accuracy of the small training corpus. Despite the large training corpus was 21.9 times larger than the small corpus in terms of word

[1]Accuracies above a baseline of 0.235 were considered somewhat useful.

count. The vast distinction in runtime during optimisation was however the primary reason to why the small corpus was used in further experiments.

Both the large and the small training corpus from Section 5.1.2 comprised a portion of OpenSubtitles2016 that was found to contain noise. Being extracted directly from the Internet, the noTenTen corpus did not appear to contain the misspellings in form of mistaken letters. A rather small portion of the noTenTen corpus should have been used as training corpus to the statistical compound word splitter, to see if this alone serves a better basis to the splitter.

In Table 5.6 no morphological operation was found including the hyphen character. This comes from the preprocessing where all non-letters and non-digits were removed, thus all hyphenated compounds became closed instead. Hyphens could have been passed through the preprocessing to see if it improved the compound word splitting performance.

Section 5.2.1 briefly states that words in some cases were erroneously split or kept as a compound, based on the global term frequencies. The statistical approach had no local knowledge as basis to whether a word was a compound. The word length feature also contributed, but as a global measure too, as it used the average word length to compare with.

In Section 5.2.1 the most usual linking morphemes in Norwegian compounds were obtained. However, some words appear to include rare morphological operations. For example, the word `møkkaverden` (dung world) can be split into `møkk` (dung) and `verden` (world), which seems to use the letter `a` as linking morpheme. Suppletitive stems can also form compound words [Johannessen and Hauglin, 1996]. For example, the word `bilde` (picture) is found in `billedkunstner` (picture artist) and `billedspråk` (picture language), using the morphological operation (*led/de*). The word `klesskap` (closet) is composed by `klær` (clothes) and `skap` (cupboard). This appears to use the morphological operation (*ær/es*). The approach to compound word splitting in this thesis was frequency based, and thus rare morphological operations like these were not obtained.

The Norwegian splitter

As Section 5.2.1 states, the Norwegian splitter did not achieve a desirable performance. We see this further in Section 5.2.2, where words were left unaltered instead of being split. The problem when making a compound word splitter like this is that it builds on knowledge about a specific language. When implementing this splitter, the algorithms were seen as intricate and were never fully understood. To let assumptions interfere in the details of these algorithms is a likely reason to why this approach failed.

6.1.2. Document retrieval

Translating the query pairs Q_1, Q_2 and Q_3 from Norwegian proved the point that compound word splitting is not always desirable. As summarised in Section 5.2.2, words were split unnecessarily, leading to completely different translations. These words should first have been looked up in the dictionaries to affirm that compound word splitting was unnecessary.

In the cases of Q_4, Q_5 and Q_6, compound word splitting was necessary when translating from Norwegian.The results in Section 5.2.2 also points out how translation from English to Norwegian were successful only when document retrieval targeted the corpus that was subject to compound word splitting. This is because all compound words involved were in the open form. On the other hand, this also lead to a disadvantage of the bag of words model, where the relation between words in an open compound is lost. This is indicated by the generally high amount of false positives involving these queries.

From the results in Section 5.2.2 alone, it is not possible to determine what is generally the most favourable of query balancing or reverse pruning. Reverse pruning was in some cases able to reduce false positives, but in exchange of reduced recall. If not leading to completely loss of recall, this generally improved the precision. In other cases query balancing outperformed reverse pruning in terms of recall, but with loss of precision.

The results in Section 5.2.2 did not prove the use of part-of-speech to be useful, as it generally lead to reduction of recall and little improvement of precision. Albeit consistency was generally found from different part-of-speech taggers. However, it was not always the case between the dictionaries involved in the experiments.

It was found in the particular case of Q_5 in Section 5.2.2 that the Oslo-Bergen Tagger could not recognise the named entity `Poirot` as a noun, whereas the TreeTagger assumed the word to be a noun. The queries did not comprise any sentences, making it hard to actually determine this part-of-speech. As two different decisions appeared, the part-of-speech as a constraint to the document retrieval did not lead to any results. The aid of named entity recognition would be valuable in cases like this. The query pair Q_6 involved a named entity that varies depending on the language. Thus, a more sophisticated approach of named entity recognition would be needed to handle this.

6.2. Contributions

In the work of this thesis, compound word splitting was applied in cross-lingual information retrieval. Hence the contribution from this thesis is two-folded.

The compound word splitting combined statistical information from a training corpus with the word length feature. Term frequency in the training corpus was used in order to score and find the most likely split of a word. The word length feature served both as an inexpensive means to filter noise and as an approximation to decide whether a word is a compound. Using the word length feature resulted in improved compound word splitting as opposed to the entirely statistical approach.

It was found that even if compound word splitting was possible, it was not always appropriate. Splitting a word from a closed to an open compound added complexity to the document retrieval. If a dictionary can provide a translation of a word, then compound word splitting should be avoided.

6.3. Conclusion

The motivation for this thesis was to make a framework for cross-lingual information retrieval with focus on some few aspects that were reflected in the research questions.

1. *How to perform CLIR and be as little dependent on language specific resources as possible?*

The final implementation mainly consisted of components that were intended to work in any language. The exception was the Oslo-Bergen Tagger which acted as a replacement to the TreeTagger for Norwegian. However, some inconsistency in part-of-speech occurred with the dictionaries provided. This lead to the assumption that matching part-of-speech in document retrieval was not useful. The statistical approach to compound word splitting proved successful in the aspect of being language independent. It did not rely on language specific resources itself, and it improved document retrieval in certain cases.

2. *What is the best method to split compound words?*

The statistical approach to compound word splitting was able to operate on a relatively small training corpus, and given the input data, experiments resulted in usable accuracies. Experimenting with a remarkably larger corpus yielded some better results, but was also found to be time consuming. To amend usage of the word length feature into the statistical splitter was found to be more useful, taking the time consumption into account, albeit it was not originally the plan to consider. The increase in performance was slightly less than with changing to a large corpus.

3. *What are named entities, and why do they have to be identified?*

A brief literature review was conducted to explore different approaches to treat named entities. The experiments in this thesis also proved how unhandled named entities can affect document retrieval in a cross-lingual setting.

6.4. Future Work

The limitations from Section 6.1 leads to some possible improvements, as will be proposed here. Neergaard [2012] already proved numerous challenges when implementing a framework to perform CLIR. In this section new or specifically relevant topics will be addressed.

6.4.1. Statistical compound word splitting

The word length feature

The exact nature of noise removal and word length penalty were not examined like with the optimisation of split penalty. These amendments were tuned briefly for use in Norwegian language only. It might be that their potential were not met during these experiments, and they would possibly need to be altered to work in other languages as well.

Rare morphological operations

As stated in Section 6.1.1, more morphological operations exist in Norwegian than what the experimental results indicate. It is also assumed that such rare morphological operations occur only with specific words. In order to choose correct morphological operations to decompose a word, it may be beneficial to keep track of how often they occur together.

Compound word splitting and the bag of words

In Section 5.2.2 it appeared that splitting compound words in the bag of words model involved large amounts of false positives when applying document retrieval. The vector space model could be extended by using a composite pattern, such that terms can be a group of terms in order to conserve their relation as a compound. Such a term group should thus be unbreakable, it should not be possible to consider just one of its terms. A term group will then be given an overall weighting to indicate the importance of its members together.

6.4.2. Named entity recognition

The results from Section 5.2.2 show that unhandled named entities can affect document retrieval with various outcomes. Named entities would have to be identified, but also classified, in order to determine whether they should be subject to translation. Although not proven in experiments of this thesis, classification of named entity would also be useful in order to obtain the correct translations.

6.4.3. Automatic detection of stop words

In this thesis stop words were removed categorically using stop word lists intended for specific languages. In order to satisfy the language independency more properly, perhaps a statistical approach could be made to determine stop words.

Bibliography

M Aljlayl, O Frieder, and D Grossman. On bidirectional English–Arabic search. *Journal of the American Society for Information Science and Technology*, 53(13):1139–1151, 2002. Wiley Online Library.

Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. Modern Information Retrieval - the concepts and technology behind search, Second edition. Addison-Wesley Professional, 2011.

Thorsten Brants. TnT: a statistical part-of-speech tagger. In *Proceedings of the sixth conference on Applied natural language processing*, pages 224–231. Association for Computational Linguistics, Saarland University, Germany, 2000.

Eric Brill. A simple rule-based part of speech tagger. In *Proceedings of the workshop on Speech and Natural Language*, pages 112–116. Association for Computational Linguistics, University of Pennsylvania, Pennsylvania, 1992.

Aitao Chen and Fredric C. Gey. Combining query translation and document translation in cross-language retrieval. In *Comparative evaluation of multilingual information access systems*, pages 108–121. Springer, 2003.

Silviu Cucerzan. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 708–716, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/D/D07/D07-1074`.

Jay Earley. An Efficient Context-free Parsing Algorithm. *Commun. ACM*, 13(2):94–102, February 1970. ISSN 0001-0782. doi: 10.1145/362007.362035. URL `http://doi.acm.org/10.1145/362007.362035`.

Marcello Federico and Nicola Bertoldi. Statistical cross-language information retrieval using n-best query translations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 167–174. ACM, Trento, Italy, 2002.

Ralph Grishman and Beth Sundheim. Message Understanding Conference- 6: A Brief History. In *16th International Conference on Computational Linguistics, Proceedings of the Conference, COLING 1996, Center for Sprogteknologi, Copenhagen, Denmark, August 5-9, 1996*, pages 466–471. New York University, New York; Naval Command, Control and Ocean Surveillance Center, 1996. URL `http://aclweb.org/anthology/C96-1079`.

Miloš Jakubíček, Adam Kilgarriff, Vojtěch Kovář, Pavel Rychlý, Vít Suchomel, et al. The tenten corpus family. In *7th International Corpus Linguistics Conference CL*, pages 125–127. Masaryk University, Czech Republic, 2013.

Janne Bondi Johannessen and Helge Hauglin. An automatic analysis of Norwegian compounds. In *16th Scandinavian Conference of Linguistics*. University of Oslo, Oslo/Norway, 1996.

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

Jesse Saba Kirchner, Justin Nuger, and Yi Zhang. An Extensible Crosslinguistic Readability Framework. In *Proceedings of the 2Nd Workshop on Building and Using Comparable Corpora: From Parallel to Non-parallel Corpora*, BUCC '09, pages 11–18, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-53-4. URL `http://dl.acm.org/citation.cfm?id=1690339.1690344`.

Philipp Koehn and Kevin Knight. Empirical methods for compound splitting. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 187–193. Association for Computational Linguistics, University of Southern California, California, 2003.

Jure Leskovec, Anand Rajaraman, and Jeffrey D. Ullman. *Mining of Massive Datasets, 2nd Ed.* Cambridge University Press, 2014. ISBN 978-1107077232.

Gina-Anne Levow and Douglas W. Oard. *Signal boosting for translingual topic tracking.* Springer, 2002.

Pierre Lison and Jörg Tiedemann. OpenSubtitles2016: Extracting Large Parallel Corpora from Movie and TV Subtitles. 2016. University of Oslo; University of Helsinki.

Aki Loponen and Kalervo Järvelin. *Multilingual and Multimodal Information Access Evaluation: International Conference of the Cross-Language Evaluation Forum, CLEF 2010, Padua, Italy, September 20-23, 2010. Proceedings*, chapter A Dictionary- and Corpus-Independent Statistical Lemmatizer for Information Retrieval in Low Resource Languages, pages 3–14. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. ISBN 978-3-642-15998-5. doi: 10.1007/978-3-642-15998-5_3. URL `http://dx.doi.org/10.1007/978-3-642-15998-5_3`.

Klaus Macherey, Andrew M. Dai, David Talbot, Ashok C. Popat, and Franz Och. Language-independent compound splitting with morphological operations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1395–1404. Association for Computational Linguistics, Google Inc., California; University of Edinburgh, Scotland, 2011.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval.* Cambridge University Press, New York, NY, USA, 2008. ISBN 0521865719, 9780521865715.

Martin Nordal. Film recommending based on Twitter data. Technical report, Norwegian University of Science and Technology, Trondheim, December 2015.

J. Scott McCarley. Should We Translate the Documents or the Queries in Cross-language Information Retrieval? In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*, ACL '99, pages 208–214, Stroudsburg, PA, USA, 1999. Association for Computational Linguistics. ISBN 1-55860-609-3. doi: 10.3115/1034678.1034716. URL `http://dx.doi.org/10.3115/1034678.1034716`.

Miniwatts Marketing Group. Internet world users by language. `http://internetworldstats.com/stats7.htm`. Last visited: 2016-03-07.

David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007. John Benjamins publishing company.

Morten Minde Neergaard. CLIRch, an extensible open source framework for query translation: evaluated for use on the Norwegian/Spanish language pair., 2012. MSc Thesis, Norwegian University of Science and Technology, Trondheim.

Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R Curran. Learning multilingual named entity recognition from Wikipedia. *Artificial Intelligence*, 194:151–175, 2013. Elsevier.

Slav Petrov, Dipanjan Das, and Ryan McDonald. A universal part-of-speech tagset. *arXiv preprint arXiv:1104.2086*, 2011. Google Research, New York; Carniege Mellon University, Pennsylvania.

Martin F. Porter. Snowball: A language for stemming algorithms. `http://snowball.tartarus.org/texts/introduction.html`, 2001. Last visited: 2016-06-04.

Q-Success. Usage of content languages for websites. `http://w3techs.com/technologies/overview/content_language/all`. Last visited: 2016-03-01.

Martin Thorsen Ranang. *Open-Domain Word-Level Interpretation of Norwegian: Towards a General Encyclopedic Question-Answering System for Norwegian.* PhD thesis, 2010. Norwegian University of Science and Technology, Trondheim.

Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. Association for Computational Linguistics, University of Illinois, USA, 2009.

Beatrice Santorini. Part-of-speech tagging guidelines for the Penn Treebank Project (3rd revision). 1990. University of Pennsylvania, Pennsylvania.

Anne Schiller, Simone Teufel, Christine Stöckert, and Christine Thielen. Vorläufige Guidelines für das Tagging deutscher Textcorpora mit STTS. *Draft, Universities of Stuttgart and Tübingen*, 1995.

Helmut Schmid. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the international conference on new methods in language processing*, volume 12, pages 44–49. UCL Press, 1994.

Satoshi Sekine. Named entity: History and future. `http://cs.nyu.edu/~sekine/papers/NEsurvey200402.pdf`, 2004. Last visited: 2016-06-23.

Karen Spärck Jones. *A statistical interpretation of term specificity and its application in retrieval*, volume 28. MCB UP Ltd, 1972.

Jörg Tiedemann. Parallel Data, Tools and Interfaces in OPUS. In *LREC*, pages 2214–2218. Uppsala University, Uppsala/Sweden, 2012.

Wikipedia. Texas (disambiguation) — wikipedia, the free encyclopedia, 2016. URL `\url{https://en.wikipedia.org/w/index.php?title=Texas_(disambiguation)&oldid=709264007}`. [Online; accessed 11-July-2016].

A. Part-of-speech tagset replacements

When using the TreeTagger and the Oslo-Bergen Tagger, several tagsets appeared. These were converted into a common tagset, as seen in the tables of this appendix.

[1]http://www.cis.uni-muenchen.de/ schmid/tools/TreeTagger/data/spanish-tagset.txt

Tag	Replacement
adj	adjective
adv	adverb
det	determiner
inf-merke	unknown
interj	interjection
konj	conjunction
prep	preposition
pron	pronoun
sbu	unknown
subst	noun
ukjent	unknown
verb	verb
clb	punctuation

Table A.1.: The tagset of the Oslo-Bergen Tagger is simply translated from Norwegian to English.

Tag	Replacement	Tag	Replacement
CC	conjunction	VBD	verb
CD	pronoun	VBG	verb
DT	determiner	VBN	verb
EX	pronoun	VBZ	verb
FW	unknown	VBP	verb
IN	preposition	VD	verb
IN/that	pronoun	VDD	verb
JJ	adjective	VDG	verb
JJR	adjective	VDN	verb
JJS	adjective	VDZ	verb
LS	punctuation	VDP	verb
MD	verb	VH	verb
NN	noun	VHD	verb
NNS	noun	VHG	verb
NP	noun	VHN	verb
NPS	noun	VHZ	verb
PDT	determiner	VHP	verb
POS	noun	VV	verb
PP	pronoun	VVD	verb
PP$	pronoun	VVG	verb
RB	adverb	VVN	verb
RBR	adverb	VVP	verb
RBS	adverb	VVZ	verb
RP	unknown	WDT	determiner
SENT	punctuation	WP	pronoun
SYM	punctuation	WP$	pronoun
TO	pronoun	WRB	adverb
UH	interjection	:	punctuation
VB	verb	$	punctuation

Table A.2.: The English tagset[Santorini, 1990] used by the TreeTagger includes some detailed grammatical properties that were discarded in order to fit a simpler tagset.

Tag	Replacement	Tag	Replacement
ACRNM	unknown	PPO	pronoun
ADJ	adjective	PPX	pronoun
ADV	adverb	PREP	preposition
ALFP	punctuation	PREP	preposition
ALFS	punctuation	PREP/DEL	preposition
ART	pronoun	QT	punctuation
BACKSLASH	punctuation	QU	adjective
CARD	unknown	REL	pronoun
CC	punctuation	RP	punctuation
CCAD	punctuation	SE	pronoun
CCNEG	punctuation	SEMICOLON	punctuation
CM	punctuation	SLASH	punctuation
CODE	unknown	SYM	punctuation
COLON	punctuation	UMMX	noun
CQUE	conjunction	VCLIger	verb
CSUBF	conjunction	VCLIinf	verb
CSUBI	conjunction	VCLIfin	verb
CSUBX	conjunction	VEadj	verb
DASH	punctuation	VEfin	verb
DM	pronoun	VEger	verb
DOTS	punctuation	VEinf	verb
FO	unknown	VHadj	verb
FS	punctuation	VHfin	verb
INT	pronoun	VHger	verb
ITJN	conjunction	VHinf	verb
LP	punctuation	VLadj	verb
NC	noun	VLfin	verb
NEG	unknown	VLger	verb
NMEA	noun	VLinf	verb
NMON	noun	VMadj	verb
NP	noun	VMfin	verb
ORD	unknown	VMger	verb
PAL	unknown	VMinf	verb
PDEL	unknown	VSadj	verb
PE	unknown	VSfin	verb
PERCT	punctuation	VSger	verb
PNC	punctuation	VSinf	verb
PPC	pronoun		

Table A.3.: The Spanish tagset[1]used by the TreeTagger includes some detailed grammatical properties that were discarded in order to fit a simpler tagset.

Tag	Replacement	Tag	Replacement
ACRNM	unknown	PPO	pronoun
ADJ	adjective	PPX	pronoun
ADV	adverb	PREP	preposition
ALFP	punctuation	PREP	preposition
ALFS	punctuation	PREP/DEL	preposition
ART	pronoun	QT	punctuation
BACKSLASH	punctuation	QU	adjective
CARD	unknown	REL	pronoun
CC	punctuation	RP	punctuation
CCAD	punctuation	SE	pronoun
CCNEG	punctuation	SEMICOLON	punctuation
CM	punctuation	SLASH	punctuation
CODE	unknown	SYM	punctuation
COLON	punctuation	UMMX	noun
CQUE	conjunction	VCLIger	verb
CSUBF	conjunction	VCLIinf	verb
CSUBI	conjunction	VCLIfin	verb
CSUBX	conjunction	VEadj	verb
DASH	punctuation	VEfin	verb
DM	pronoun	VEger	verb
DOTS	punctuation	VEinf	verb
FO	unknown	VHadj	verb
FS	punctuation	VHfin	verb
INT	pronoun	VHger	verb
ITJN	conjunction	VHinf	verb
LP	punctuation	VLadj	verb
NC	noun	VLfin	verb
NEG	unknown	VLger	verb
NMEA	noun	VLinf	verb
NMON	noun	VMadj	verb
NP	noun	VMfin	verb
ORD	unknown	VMger	verb
PAL	unknown	VMinf	verb
PDEL	unknown	VSadj	verb
PE	unknown	VSfin	verb
PERCT	punctuation	VSger	verb
PNC	punctuation	VSinf	verb
PPC	pronoun		

Table A.4.: The German tagset[Schiller et al., 1995] used by the TreeTagger includes some detailed grammatical properties that were discarded in order to fit a simpler tagset.

B. Results from document retrievals

This appendix contains results from the information retrieval experiments described in Chapter 5.1.4.

Configuration	Retrieved	Recall	Precision	F_1-score
reference	4	1	1	1
rev	4	1	1	1
bal	4	1	1	1
pos-rev	4	1	1	1
pos-bal	4	1	1	1
cws-s-rev	0	0	0	0
cws-s-bal	0	0	0	0
cws-s-pos-rev	0	0	0	0
cws-s-pos-bal	0	0	0	0
cws-n-rev	4	1	1	1
cws-n-bal	4	1	1	1
cws-n-pos-rev	4	1	1	1
cws-n-pos-bal	4	1	1	1

(a) Translation from English to Norwegian.

Configuration	Retrieved	Recall	Precision	F_1-score
reference	3	1	1	1
rev	3	1	1	1
bal	98	1	0.03061	0.05941
pos-rev	3	1	1	1
pos-bal	81	1	0.03704	0.07143
cws-s-rev	3052	0	0	0
cws-s-bal	3175	0	0	0
cws-s-pos-rev	1457	0	0	0
cws-s-pos-bal	2183	0	0	0
cws-n-rev	5589	1	0.0005368	0.001073
cws-n-bal	5589	1	0.0005368	0.001073
cws-n-pos-rev	0	0	0	0
cws-n-pos-bal	0	0	0	0

(b) Translation from Norwegian to English.

Table B.1.: Retrieval performance results when applying Q_1.

Configuration	Retrieved	Recall	Precision	F_1-score
reference	9	1	1	1
rev	9	1	1	1
bal	9	1	1	1
pos-rev	0	0	0	0
pos-bal	0	0	0	0
cws-s-rev	0	0	0	0
cws-s-bal	0	0	0	0
cws-s-pos-rev	0	0	0	0
cws-s-pos-bal	0	0	0	0
cws-n-rev	9	1	1	1
cws-n-bal	9	1	1	1
cws-n-pos-rev	0	0	0	0
cws-n-pos-bal	0	0	0	0

(a) Translation from German to Norwegian.

Configuration	Retrieved	Recall	Precision	F_1-score
reference	30	1	1	1
rev	30	1	1	1
bal	30	1	1	1
pos-rev	0	0	0	0
pos-bal	0	0	0	0
cws-s-rev	827	0.9667	0.03507	0.06768
cws-s-bal	827	0.9667	0.03507	0.06768
cws-s-pos-rev	0	0	0	0
cws-s-pos-bal	0	0	0	0
cws-n-rev	30	1	1	1
cws-n-bal	30	1	1	1
cws-n-pos-rev	0	0	0	0
cws-n-pos-bal	0	0	0	0

(b) Translation from Norwegian to German.

Table B.2.: Retrieval performance results when applying Q_2.

Configuration	Retrieved	Recall	Precision	F_1-score
reference	3	1	1	1
rev	0	0	0	0
bal	289	0.6667	0.006920	0.01370
pos-rev	0	0	0	0
pos-bal	289	0.6667	0.006920	0.01370
cws-s-rev	0	0	0	0
cws-s-bal	305	0.6667	0.006557	0.01299
cws-s-pos-rev	0	0	0	0
cws-s-pos-bal	305	0.6667	0.006557	0.01299
cws-n-rev	0	0	0	0
cws-n-bal	287	0.6667	0.006969	0.01379
cws-n-pos-rev	0	0	0	0
cws-n-pos-bal	287	0.6667	0.006969	0.01379

(a) Translation from Spanish to Norwegian.

Configuration	Retrieved	Recall	Precision	F_1-score
reference	3	1	1	1
rev	0	0	0	0
bal	3	1	1	1
pos-rev	0	0	0	0
pos-bal	2	0.6667	1	0.8000
cws-s-rev	3442	1	0.0008716	0.001742
cws-s-bal	3451	1	0.0008693	0.001737
cws-s-pos-rev	3428	1	0.0008751	0.001749
cws-s-pos-bal	3448	1	0.0008701	0.001739
cws-n-rev	0	0	0	0
cws-n-bal	3	1	1	1
cws-n-pos-rev	0	0	0	0
cws-n-pos-bal	2	0.6667	1	0.8000

(b) Translation from Norwegian to Spanish.

Table B.3.: Retrieval performance results when applying Q_3.

Configuration	Retrieved	Recall	Precision	F_1-score
reference	2	1	1	1
rev	220	0.5	0.004545	0.009009
bal	528	1	0.003788	0.007547
pos-rev	95	0.5	0.01053	0.02062
pos-bal	95	0.5	0.01053	0.02062
cws-s-rev	228	1	0.008772	0.01739
cws-s-bal	556	1	0.003597	0.007168
cws-s-pos-rev	133	1	0.01504	0.02963
cws-s-pos-bal	133	1	0.01504	0.02963
cws-n-rev	220	0.5	0.004545	0.009009
cws-n-bal	528	1	0.003788	0.007547
cws-n-pos-rev	94	0.5	0.01064	0.02083
cws-n-pos-bal	94	0.5	0.01064	0.02083

(a) Translation from English to Norwegian.

Configuration	Retrieved	Recall	Precision	F_1-score
reference	748	1	0.05481	0.1039
rev	0	0	0	0
bal	0	0	0	0
pos-rev	0	0	0	0
pos-bal	0	0	0	0
cws-s-rev	1790	1	0.02291	0.04478
cws-s-bal	1790	1	0.02291	0.04478
cws-s-pos-rev	1611	1	0.02545	0.04964
cws-s-pos-bal	1611	1	0.02545	0.04964
cws-n-rev	0	0	0	0
cws-n-bal	0	0	0	0
cws-n-pos-rev	0	0	0	0
cws-n-pos-bal	0	0	0	0

(b) Translation from Norwegian to English.

Table B.4.: Retrieval performance results when applying Q_4.

Configuration	Retrieved	Recall	Precision	F_1-score
reference	1	1	1	1
rev	288	1	0.003472	0.006920
bal	292	1	0.003425	0.006826
pos-rev	288	1	0.003472	0.006920
pos-bal	292	1	0.003425	0.006826
cws-s-rev	321	1	0.003115	0.006211
cws-s-bal	337	1	0.002967	0.005917
cws-s-pos-rev	321	1	0.003115	0.006211
cws-s-pos-bal	337	1	0.002967	0.005917
cws-n-rev	264	1	0.003788	0.007547
cws-n-bal	269	1	0.003717	0.007407
cws-n-pos-rev	264	1	0.003788	0.007547
cws-n-pos-bal	269	1	0.003717	0.007407

(a) Translation from English to Norwegian.

Configuration	Retrieved	Recall	Precision	F_1-score
reference	1108	1	0.005415	0.01077
rev	6	1	1	1
bal	6	1	1	1
pos-rev	0	0	0	0
pos-bal	6	1	1	1
cws-s-rev	1725	1	0.003478	0.006932
cws-s-bal	1725	1	0.003478	0.006932
cws-s-pos-rev	1724	0.8333	0.002900	0.005780
cws-s-pos-bal	1725	1	0.003478	0.006932
cws-n-rev	6	1	1	1
cws-n-bal	6	1	1	1
cws-n-pos-rev	0	0	0	0
cws-n-pos-bal	6	1	1	1

(b) Translation from Norwegian to English.

Table B.5.: Retrieval performance results when applying Q_5.

Configuration	Retrieved	Recall	Precision	F_1-score
reference	2	1	1	1
rev	52	0	0	0
bal	628	1	0.003185	0.006349
pos-rev	50	0	0	0
pos-bal	350	1	0.005714	0.01136
cws-s-rev	70	1	0.02857	0.05556
cws-s-bal	628	1	0.003185	0.006349
cws-s-pos-rev	68	1	0.02941	0.05714
cws-s-pos-bal	153	1	0.01307	0.02581
cws-n-rev	52	0	0	0
cws-n-bal	628	1	0.003185	0.006349
cws-n-pos-rev	50	0	0	0
cws-n-pos-bal	349	1	0.005731	0.01140

(a) Translation from English to Norwegian.

Configuration	Retrieved	Recall	Precision	F_1-score
reference	2095	1	0.009547	0.01891
rev	0	0	0	0
bal	0	0	0	0
pos-rev	0	0	0	0
pos-bal	0	0	0	0
cws-s-rev	5056	1	0.003956	0.007880
cws-s-bal	5056	1	0.003956	0.007880
cws-s-pos-rev	4469	0.9500	0.004252	0.008465
cws-s-pos-bal	4469	0.9500	0.004252	0.008465
cws-n-rev	0	0	0	0
cws-n-bal	0	0	0	0
cws-n-pos-rev	0	0	0	0
cws-n-pos-bal	0	0	0	0

(b) Translation from Norwegian to English.

Table B.6.: Retrieval performance results when applying Q_6.

C. How to setup the framework

The framework uses components that are distributed across several neighbouring directories: `clir-src`, `clir-corpora`, `clir-lexica` and `clir-tagger`. These names and locations can be altered.

C.1. clir-src

This directory will contain the source code. The framework uses Maven for automatic installation of the following dependencies:

org.jdom:jdom2:2.0.6

mysql:mysql-connector-java:5.1.39

org.webjars:jquery:2.1.4

org.apache.commons:commons-compress:1.10

org.annolab.tt4j:org.annolab.tt4j:1.2.1

com.github.rholder:snowball-stemmer:1.3.0.581.1

In addition the Oslo-Bergen Tagger (*nix OS-es only!) and the TreeTagger need to be installed manually. The file `clir-src/script/obtagger.sh` is used to run the Oslo-Bergen Tagger and must be made executable. The file `clir-src/environment.properties` must be revised to set certain settings such as database connection parameters.

C.2. clir-corpora

Corpora are grouped in directories with names corresponding to their source. However, only support for Opus parallel corpora has currently been implemented. The next directory is named after the collection. Then all documents should be grouped by language in a single archive, as provided by the Opus project, leaving file paths like this: `clir-corpora/Opus/OpenSubtitles2016/en.tar.gz` The framework will read through these archives at runtime.

C.3. clir-lexica

Dictionaries that were used in this thesis are not for redistribution, but the procedure to have them extracted remains in the framework just in case. It is likely that an alternative dictionary extractor will have to be made. When unpacked, each item in a dictionary should appear in the following single-line pattern:

```
word[pos]: translation_a/translation_b :: (other) translation ;; other group
of :: translations
```

The `[pos]` (part of speech) is optional. Words appearing in parenthesis means that they may be part of the translation and not. A slash indicates that both of the two surrounding words are translations, individually.

C.4. clir-tagger

This is the directory to place TreeTagger parameter files, uncompressed.