Universität des Saarlandes

Naturwissenschaftlich-Technische Fakultät I

Fachrichtung Informatik

Masterarbeit

Linking Multilingual Semi-Structured Documents

vorgelegt von

Florian Dietz

am 2015-03-24

Supervisor:

Prof. Dr. Dietrich Klakow

Advisor:

Dr. Sergey Sosnovsky

Begutachtet von:

Prof. Dr. Dietrich Klakow

Prof. Dr. Jörg Siekmann

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Statement in Lieu of an Oath

I hereby confirm that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis

Einverständniserklärung

Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlicht wird.

Declaration of Consent

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Saarbrücken,_____ _____
(Datum/Date) (Unterschrift/Signature)

Acknowledgements

I would like to thank my advisor, Dr. Sergey Sosnovsky, for his advice and support. I am especially thankful for the numerous discussions we had towards defining and realizing this work. I am grateful to my supervisor, Prof. Dietrich Klakow, who first introduced me to the field of natural language processing and kindled my interest in it.

I further want to thank my colleagues, Thorsten Hey, Özgün Erensoy, and Kyrill Pugatschewski. They were always there for me to talk to, and the project was only made possible by our collaboration.

UNIVERSITÄT DES SAARLANDES

Abstract

Naturwissenschaftlich-Technische Fakultät I

Master of Science

Linking Multilingual Semi-Structured Documents

by Florian Dietz

This thesis examines the possibility of creating a semantic linking between parts of textbooks in different languages. This is done as part of the Interlingua project, an international research effort with the goal of creating an online e-learning service to help students who study in a foreign language.

To perform this linking, a reference model is used to create a VSM. This is based on a curated multilingual glossary, the ISI glossary. Different ways of constructing this VSM are explored, based on different interpretations of the glossary. Several ways of improving performance are attempted: Dimensionality reduction in the form of LSA and PCA is used to compress the VSM. Most importantly, unlike previous, similar tasks, the documents used here are inherently structured, since they are parts of textbooks. Two different ways of utilizing this information were tried. Firstly, when linking a part of a textbook, its neighbouring sections are considered as well and influence the similarity score. Secondly, the similarities between the parent structures in the textbooks' hierarchical models are used as a baseline for the similarities of their children.

The quality of the linking was measured using NDCG. Results showed that the different ways of constructing the initial VSM result in different linking qualities and that best results can be achieved by combining them. Dimensionality reduction techniques did not improve performance. Taking the order and structure of textbooks into account did improve performance on both counts. In all cases, the gains were small but noticeable.

Abbreviations

NLP	**N**atural **L**anguage **P**rocessing
MLDC	**M**ulti**l**ingual **D**ocument **C**lustering
MT	**M**achine **T**ranslation
TF-IDF	**T**erm **F**requency - **I**nverse **D**ocument **F**requency
VSM	**V**ector **S**pace **M**odel
LSA/LSI	**L**atent **S**emantic **A**nalysis/**I**ndexing
LDA	**L**atent **D**irichlet **A**llocation
PCA	**P**rincipal **C**omponent **A**nalysis
ISI	**I**nternational **S**tatistical **I**nstitute
MAP	**M**ean **A**verage **P**recision
NDCG	**N**ormalized **D**iscounted **C**umulative **G**ain
NER	**N**amed **E**ntity **R**ecognition
POS-tagging	**P**art **O**f **S**peech tagging

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Today, there is a plethora of information online about virtually any topic. Search engines are a good way to get information in a general sense, but they are often not specific enough. Students in particular frequently find themselves reading textbooks that have a few sections that are hard to understand. It is useful in such cases to provide the student with a similar section in another textbook that may be easier to understand. This is a more specific type of retrieval task than search engines usually deal with.

This has already been looked at for English textbooks[1]. However, international students face the same problem, and the issue is more important for them: many international students speak a language well enough to understand and say most things, but they can be confused by technical terms or previously unencountered figures of speech. Often they are missing prerequisites in a different language that prevent a good understanding of the text. It is important that these students continue to read in the language foreign to them so that this information gap closes over time instead of widening further. However, it can occasionally be useful to give them access to text in their native language in order to bridge the gap and allow them to better understand the foreign text.

Most commonly used online sources of information, like Wikipedia, are often not suited for educational purposes. They tend to describe things, rather than explaining them.

Ideally, there should be a way to easily and swiftly provide these students with text passages in their native tongue that explain what they do not understand. This way, they can follow the official textbook and practice the language normally, but when something is hard, they can quickly clarify things by looking at a section in a textbook of their native language that describes the same thing.

1.2 Context: The Interlingua Project

The Interlingua project aims at solving this problem. It is an international cooperative effort between the German Research Center for Artificial Intelligence (DFKI, Germany), the Luxemburg Institute of Science and Technology (TUDOR, Luxembourg), the University of Lorraine (France), Liége University (Belgium) and the Hochschule für Technik und Wirtschaft des Saarlandes (HTW Saar, Germany). The project is part of the program Interreg IV-A.

The primary goal of the Interlingua project is providing an extensible solution to support students of the Greater Region at studying in a foreign language. This goal is achieved by developing e-learning systems to allow foreign students on-demand access to educational content in their mother tongue[2].

Among the problems addressed by the project are the following:

- Institutions are often unaware of the issues faced by foreign students.
- The course material is rarely adapted for the cultural, linguistic and background conditions.
- Existing e-learning systems do not take advantage of modern semantic and adaptive technologies.
- Standard teaching methodology lacks such effective educational techniques as formative assessment and individual scaffolding.
- Foreign students frequently have problems connecting new material to their background knowledge.

The Interlingua project develops an e-learning platform. It will include a web service through which students will be able to query a server for alternate textbooks in their native language. This will help students to make connections between their native language and the new, foreign study material.

In order to provide alternatives for requested sections of textbooks, the project is designed to feature a database of pre-computed optimal links between collections of textbooks in different languages. Ultimately, this should cover as many languages as possible. For now however, it will be limited to the three languages of the partners working on Interlingua: English, French and German.

1.3 Task

The topic of this thesis is the creation of the linking between textbooks of different languages. We assume here that we already have a collection of textbooks to link. This is not obvious, since textbooks are usually published in PDF format, which is difficult to parse. However, this problem is dealt with by other parts of the Interlingua project.

This preprocessing not only makes PDF files readable, but also splits them into separate documents along natural dividers of the text. Each chapter, sub-chapter, sub-sub-chapter and so on becomes its own document. The task is to find, for any given such part of a textbook, another part in another textbook in the target language. See Figure 1.1 for the preprocessing and Figure 1.2 for the linking.

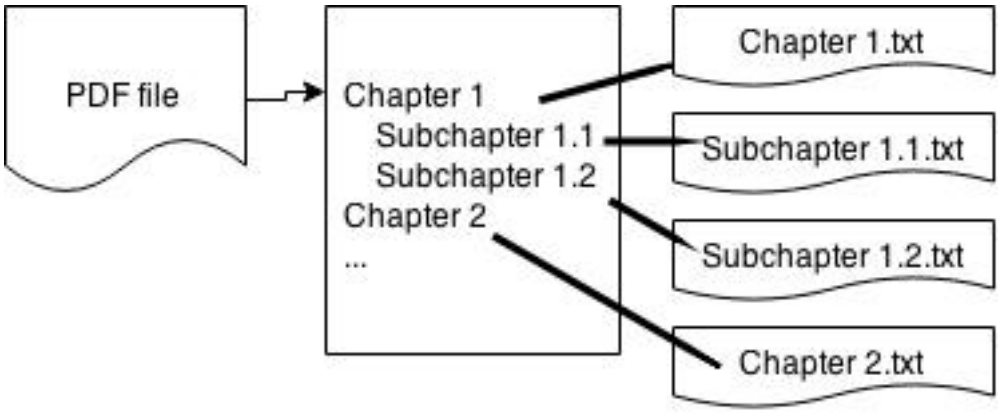

FIGURE 1.1: Each PDF file of a textbook is split into segments according to its content in a preprocessing step. Each Segment gets its own text file, which can be easily parsed and indexed, unlike the PDF files themselves.

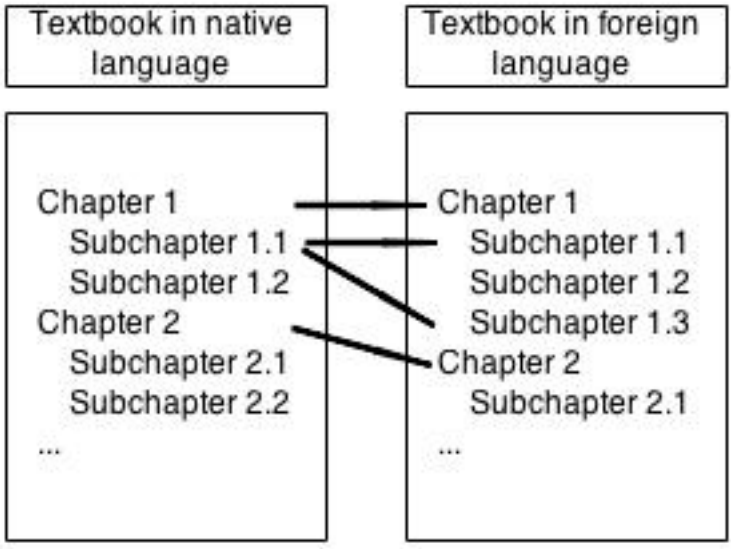

FIGURE 1.2: In this example, there are two textbooks of different languages. Different parts of these books are linked to each other by similarity. Note that some parts of textbooks can have several links, while others have none. This depends entirely on the content of the textbooks.

The linking is static and computed offline. This means that the results are precomputed and can be accessed quickly when a user makes a query. The addition of new textbooks will require the linking to be recomputed, but almost no processing time will need to be expended on calculating the results of queries. Figure 1.3 illustrates the offline processing that will be described in detail in this thesis. Figure 1.4 shows how users will use the final product.

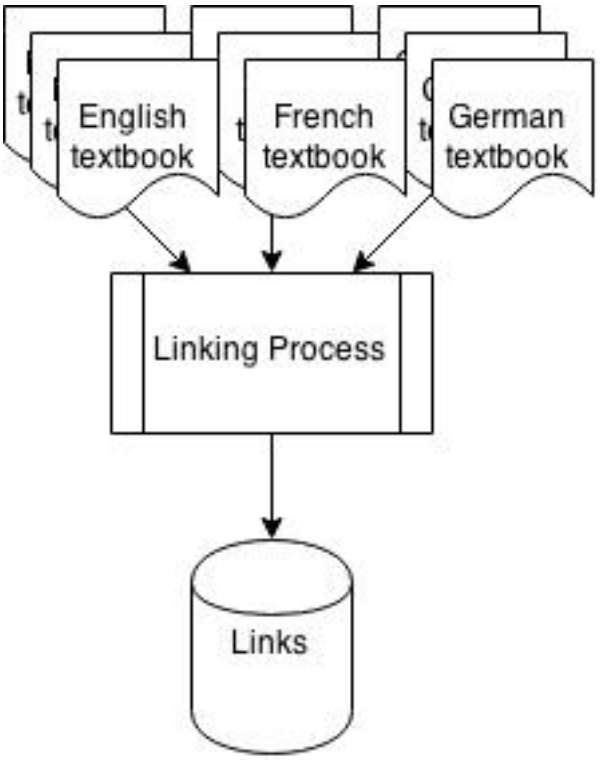

FIGURE 1.3: In a preprocessing step, collections of textbooks of various languages are given to the linker. The linker creates links out of them which are stored and later used to answer queries by users. This thesis is concerned with creating, improving and analyzing this linking process.

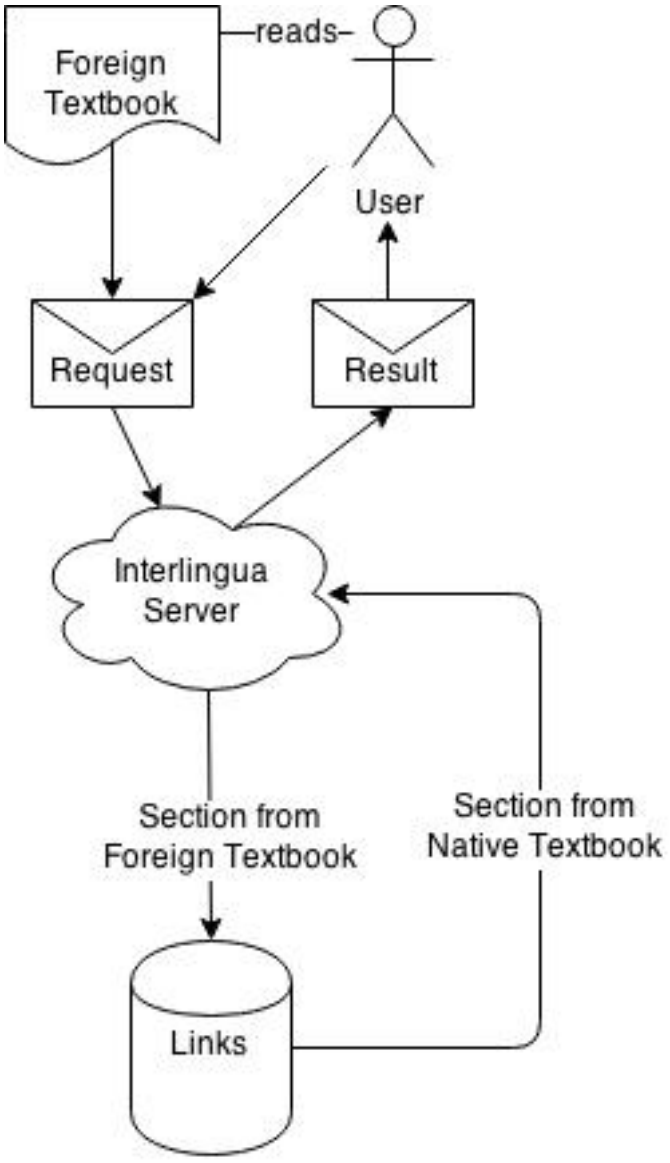

FIGURE 1.4: A student using the e-learning system is having trouble understanding a foreign textbook. They query the server with the part of the textbook they have trouble with. The server looks at the links that were computed previously and returns the best matching sections of other textbooks in the user's native language.

1.4 Approach

The textbooks used in this thesis are limited to the domain of Probability and Statistics, as an initial test of the linking system. The glossary of the International Statistical Institute (ISI) is used to match this domain. The Languages used for textbooks are English, French and German.

To perform the linking between parts of textbooks, the ISI glossary is used as a reference model to build a VSM. Three primary approaches to do so are tested and compared. The resulting VSM is modified in several ways. Many of these modifications include the search for optimal parameters, and their impact on performance is analyzed:

- Dimensionality reduction is applied to the VSM.
- The order of parts of the textbooks is used to enrich the VSM.
- The hierarchical structure of the textbooks is used to create baselines for the linking.

The quality of the linking is judged based on expert-feedback. As the Interlingua project will continue after this thesis, this is only a proof-of-concept. The final result will likely be much better, since far more data will be available for training once the project has matured and grown.

1.5 Specifics of the Thesis

The scope of the Interlingua project will grow over time, but for this thesis, only the domain of Probability and Statistics is used. Three textbooks in this domain are used as input, in three different languages:

- English: Probability & Statistics for Engineers & Scientists[3]
- French: Probabilités et Statistique[4]
- German: Statistik[5]

To match this domain, the glossary of the International Statistical Institute (ISI) is used. Specifics on this follow in Chapter 3.

1.6 Thesis Structure

This thesis is structured as follows:

Chapter 2 mentions related work. Chapter 3 explains the approach taken in this thesis. Chapter 4 describes implementational details. Chapter 5 analyzes the results. Chapter 6 concludes this thesis and mentions future work.

Chapter 2

Related Work

Multilingual Document Clustering (MLDC) is the field of research concerned with finding similarities between documents in different languages and grouping them together. It is particularly important for news-agencies, which often use automated systems to track the formation and development of topics of news. MLDC makes this possible across language barriers[6][7][8].

MLDC usually first determines the similarities between different documents (like individual news articles), and then groups them together with a clustering algorithm. The linking between parts of textbooks in the Interlingua project does not require such clustering. However, the techniques used in MLDC to get the similarities between documents are relevant for us. See Figure 2.1 for a visual summary of the rest of this chapter.

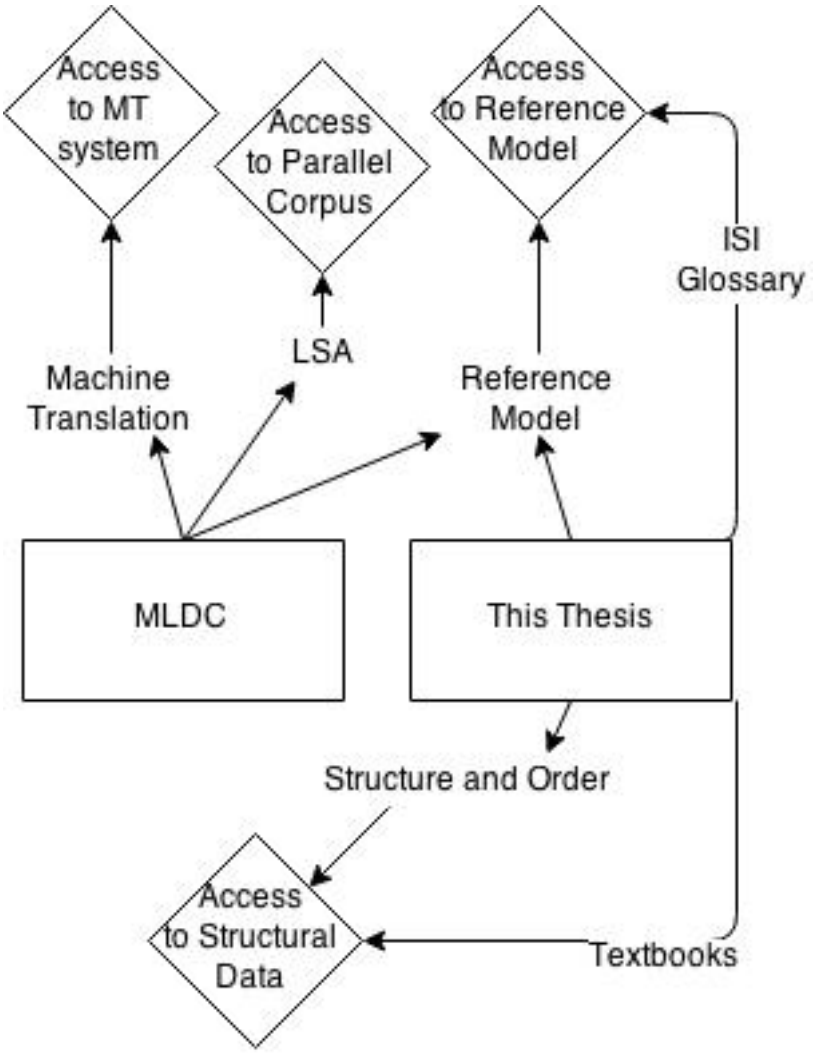

FIGURE 2.1: This Diagram summarizes the similarities and differences between MLDC and the task of this thesis. Of the three candidate approaches (MT, LSA, reference model), we use the latter because we have access to a reference model through the ISI Glossary. In addition to this, textbooks are structured data, which gives us possibilities that are not usually available in MLDC. Details will become clear over the course of this chapter.

2.1 Standard Approaches

There are a number of different standard approaches for finding the similarity between two documents in different languages[9]. Each of them has different requirements and uses, but what they all have in common is that they need some way of getting the Cross-Language information. Each of the approaches mentioned here comes with a diagram to show where it draws its Cross-Language information from.

2.1.1 Machine Translation

The most straightforward way of solving the problem of MLDC is to use an existing Machine Translation (MT) system: Simply translate one or both of the two documents so that they have the same language. This reduces the problem to monolingual document clustering. Unfortunately, using machine translation greatly reduces the quality of the results when compared either to other MLDC approaches or to monolingual document clustering performed on monolingual data[9][10].

The reason for this loss of precision lies primarily in the fact that the final translation is not the only possible solution for the translation task, but merely the one with the highest probability of being correct. A statistical MT system will actually create a probability distribution over output texts and then select the best of them. This selection causes an immense loss of information as all the alternative interpretations are dropped. Performance would be far higher if this additional information were available to the monolingual document clustering algorithm. Figure 2.2 illustrates this. Another downside of the MT approach is that most MT systems are designed for general language terms. This means that they may not be able to correctly translate many of the specialized technical terms that are most relevant for the task of linking textbooks.

The upside of using Machine Translation is that it doesn't require any special data for training. All it takes is an existing MT system, and while those are difficult to create, they are also in very high demand, so it is comparatively easy to find one, like the freely available translation API's of Google Translate[11] and MateCat[12]. However, if possible, one is always better off using the internal model of an MT system instead

of using the MT system directly, as this means that one can make use of the whole probability distribution.

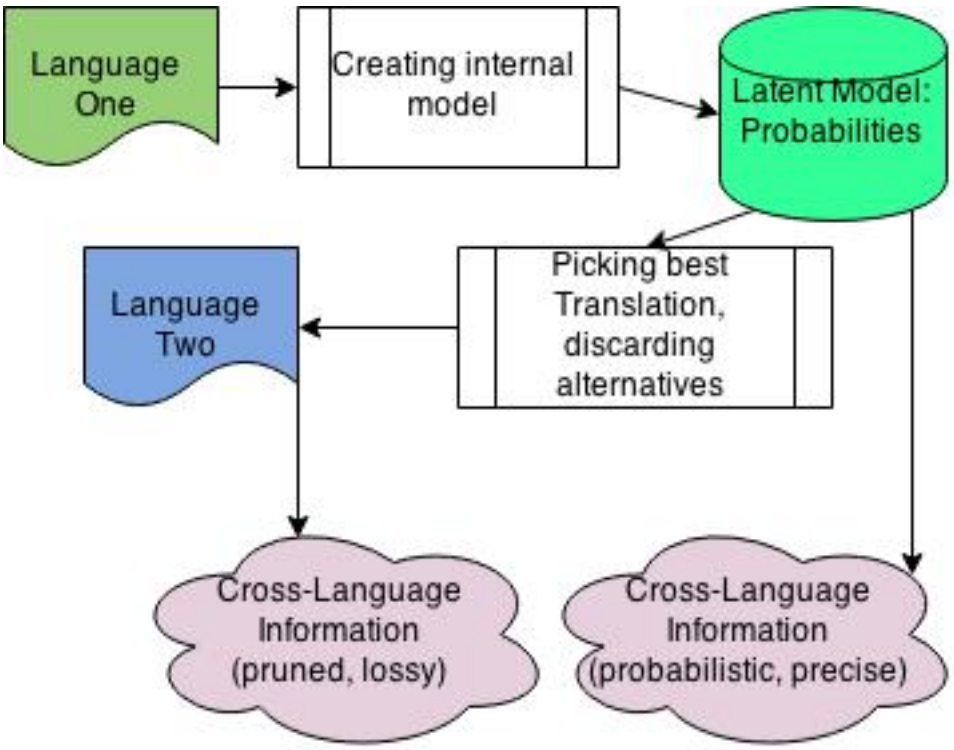

FIGURE 2.2: In this simplified illustration, Machine Translation operates in two steps. First, a Latent Model is created from the input text. This model holds a probability distribution over possible translations and contains good Cross-Language information. After that, the best of the possible translations is extracted from the probability distribution and presented as the final translation. This causes a loss of precision as all probabilistic information is removed.

2.1.2 Reference Models

Another approach is using a multilingual reference model to map each of the documents, in any language, to the same VSM[13]. Documents can then be compared directly in the shared VSM. Note that this approach goes under many names in the literature: Ontology, dictionary, thesaurus, glossary. Whatever the name, the basic idea is the same, although some have additional features: There is a database that contains a collection of concepts. Each of these concepts has example text for every language. For instance, a concept could be 'place where a person lives', and it could have the English terms "home", "place of residence" and "house" associated with it, among others. It would also have terms in other languages associated with it, like the German "Haus" and

"Zuhause". This makes it possible to map documents of different languages to the same space of concepts created by the ontology. Note that reference models can deal with both polysemy (several words having the same meaning) and with multiple languages simply by associating several terms or phrases to each concept. Such a group of terms/phrases that describe the same concept is sometimes called a synset (set of synonyms).

When constructing a VSM from an ontology, tf-idf is most popularly used as a measure to determine the strength with which a concept appears in a document, though other measures can also be used. Figure 2.3 illustrates how Cross-Language information is extracted from reference models.

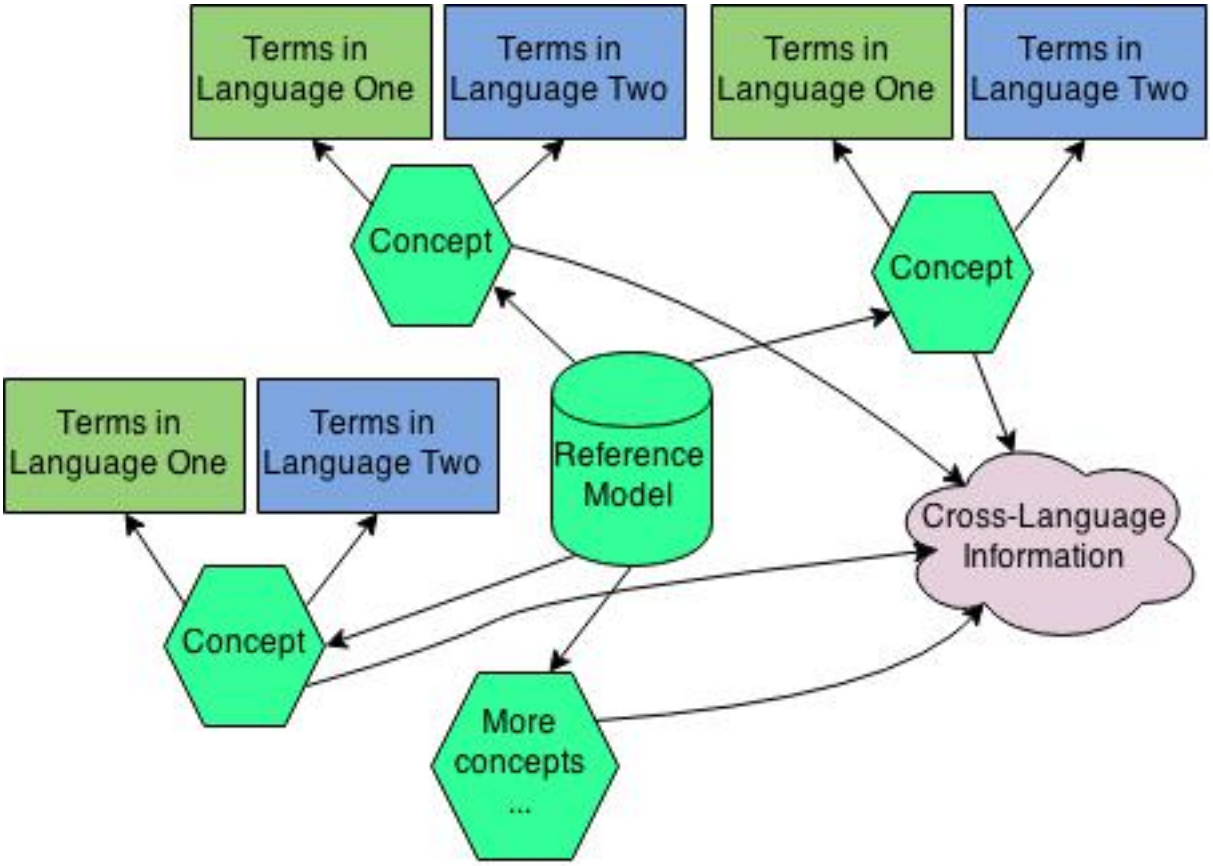

FIGURE 2.3: There is a single database, the reference model, which contains a collection of concepts. Each of these concepts has representations in several languages. These representations can take the form of individual words, or of phrases. Optionally, there can be connections between concepts in the reference model, such as "is a subtype of" or "is applicable for". Each concept thus directly captures its own Cross-Language information, which is especially good as translations are stored on a conceptual level rather than a word level.

When dealing with specialized domains, as is the case with textbooks, it is better if the

reference model is also specialized to the topic at hand. In specialized reference models the concepts in the VSM carry much more information that is actually salient to the task than in a general one. If for instance we are comparing textbooks about statistics and the reference model contains the concept 'dog', then this concept is little more than noise. A textbook may happen to contain a math problem that uses dogs as an example, but dogs don't inherently have anything to do with statistics. This means that if two statistics textbooks talk about very different things, but both of them happen to use dogs as an example, the system would falsely assign a greater similarity between the two because they both mention dogs. For this reason, concepts that are not semantically relevant to the topic cause many false positives that negatively influence the model and increase noise.

Of course, general reference models also have their uses: They are usually much larger than specialized ones, so even if each individual synset in the ontology carries less information, the sheer number of them can make up for this. Merging a specialized ontology with a global ontology is a way of getting the benefits of both, although this is not without its problems[14]. Of course, making the ontology too large slows down the program, which can make parameter optimization prohibitively expensive.

Unlike simpler reference models, like dictionaries, ontologies may additionally contain relations between concepts. 'car' may be marked as a subclass of 'vehicle', for instance. This can be modeled by making sure that every document that is marked as containing "car" is also marked as containing "vehicle". As part of the semantic web, the popularly used Web Ontology Language (OWL) has features like 'subclass of', 'property' and 'inverse of', among others[15]. This additional information can be useful, especially if there are complex hierarchies and interrelationships between concepts, but it is not strictly necessary.

If possible, it is preferable to use probabilistic ontologies over unweighted ones. It has been suggested to extend OWL to allow probabilistic connections[16]. While both "home" and "house" carry information about the concept 'place where a person lives', the former's connection is much more direct and it is useful if this fact is reflected in the ontology. Unfortunately, probabilistic ontologies are much harder to find, although

a way of generating probabilistic ontologies from deterministic ones using training data has been proposed[17].

2.1.3 LSA

A third popular approach for MLDC is Latent Semantic Analysis (LSA), also called Latent Semantic Indexing (LSI)[18][19][20]. LSA uses a mathematical technique called Singular Value Decomposition to identify patterns in the relationships between the concepts contained in an unstructured collection of text. LSA can be used in two different ways:

- If there already is a common VSM for all documents of all languages, it can be used for dimensionality reduction. By removing the low-variance components, it is possible to reduce the noise in a dataset without losing meaning. This works because high-variance dimensions of LSA tend to carry semantic information while low-variance dimensions tend to carry less meaningful information that is often just a result of chance. This usually improves the overall quality of the VSM[18].

- If a common VSM for all required languages can not be constructed, e.g. because no multilingual reference model is available, then LSA can be used to bridge the language gap[8]. Doing this requires access to a multilingual parallel corpus of documents for training, however. First, one needs to construct separate VSM's for each language. This can be done in the standard way, by creating one dimension in the VSM for each word that appears often enough and is expected to carry salient information (i.e. that is not in the stop word list). Then, all the documents that are parallel to each other have their VSM's combined, so that only a single VSM is left, which has components from each language. By running LSA on this shared VSM, the terms from the different languages are conflated together. It is then possible to extract simple rules for translating the VSM for each language into the common VSM generated by LSA, so that documents from different languages can be compared even if they do not appear in the parallel training corpus. Figure 2.4 illustrates how Cross-Language information is extracted from LSA.

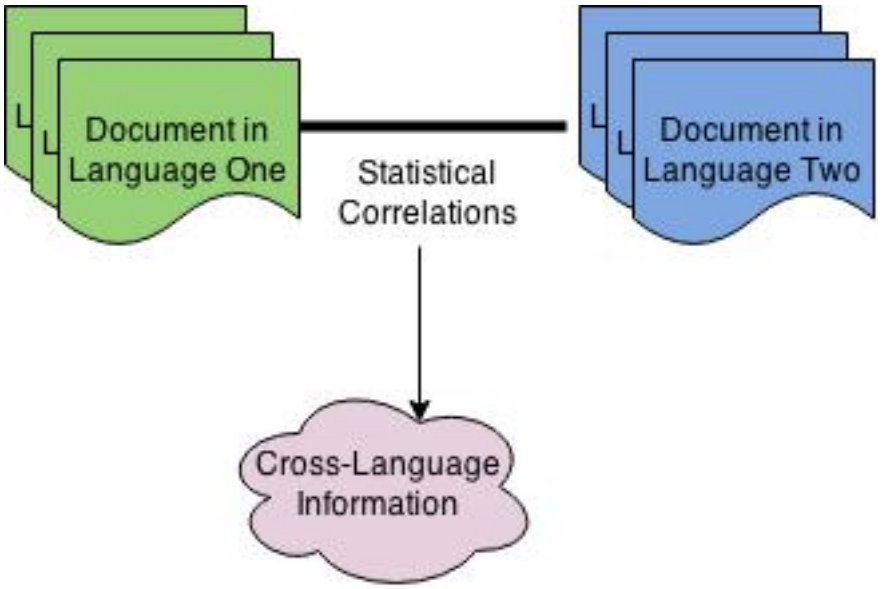

FIGURE 2.4: There is a parallel corpus of documents in two or more different languages. The document collections are combined and statistical correlations between words in different languages are measured to extract Cross-Language Information.

LSA is useful against two major problems in NLP: Synonymy (one word having several meanings) and polysemy (several words having the same meaning). It is also effective in dealing with noise and overlap between the meanings of words[21].

There are several other techniques for dimensionality reduction that can fulfill the same purpose as LSA. Latent Dirichlet Allocation (LDA) in particular is a strong competitor[22]. LSA is the older of the two and has proven itself over time. LDA is newer and has a stronger mathematical basis in probability theory. LDA may be more attractive if when using a probabilistic model, since it is itself based on probability theory, which makes it easier to integrate into probabilistic models.

2.1.4 Summary

Both multilingual ontologies and LSA give good results. The former requires a single multilingual ontology, the latter requires a parallel corpus of documents. When an ontology is available that matches the field well, using it gives good results[10]. Additionally, dimensionality reduction techniques like LSA can also be used to further improve an existing VSM.

2.2 Differences to our task

The task of this thesis differs in several ways from MLDC tasks. As mentioned before, we are only interested in getting similarities between documents, not creating clusters, but there are more important differences as well.

2.2.1 Training

We do not have access to a parallel corpus suitable for the Interlingua project. Even if we did have one, it would not be precisely what we need: What we want is a similarity-detection system, and training on a parallel corpus would result in something like a translation system. For these two reasons, we use a multilingual ontology to project all our documents to the same VSM.

2.2.2 Fields of Science

We only compare textbooks with the same topic/in the same field of science to each other. It would be pointless to link a history textbook to a physics textbook after all. While it would be possible to cluster the documents within each field of science further, the separation into different fields is already done by the authors of the textbooks.

This information can be utilized by using different reference models for each of the fields of science. Rather than using a single reference model of all terms, like Wordnet[23] or Cyc[24], where most concepts would be irrelevant for most fields, we can use reference models that are specialized for the area at hand. This leads to a higher information density in the VSM and reduces noise. Because these specialized ontologies can go deeper into the subject matter, they can also contain rarer technical terms, further improving our space of features.

For this thesis, there is only have a single field of science to test and therefore only one ontology is needed: the ISI glossary. It is a multilingual glossary and does not have connections between concepts. More on this in Chapter 3.

2.2.3 Named Entity Recognition

Named Entity Recognition (NER) has proven itself useful for linking documents[25] and there even are multilingual NER resources available[26]. However, most papers in MLDC operate on news corpora and other real-world data. This task, in contrast, operates on the far smaller domains of individual fields of science. Named Entity Recognition is likely far less useful in narrow domains, where special terminology is already strictly defined, than in general areas, where new people or places may be mentioned that haven't been recorded before. Although NER is generally useful and improves performance, it therefore likely won't be very interesting for us. Note that NER may be more useful in the social sciences than in the natural sciences, but even there it can be replaced with a Lexicon of known important entities. As the named entities referenced in textbooks do not tend to change much over time in most fields of science (compared to news and other real-world data), the recognition of new named entities is redundant if a Lexicon of already known entities exists.

2.2.4 Structure and Order

As just mentioned, news corpora dominate MLDC. They are usually unordered collections of documents with no structure between them, since most new stories are published independently and only rarely link to each other. This is also true of many other types of documents, which is why it often makes sense to ignore the order in which the documents are found. This is called the bag-of-documents assumption and it is implicitly shared by most papers on MLDC.

However, this assumption does not hold for our data, as we link *parts of* textbooks to each other. Textbooks have a fixed structure, usually hierarchical, and authors tend to be relatively consistent about the order in which they mention topics. When comparing two sections of different textbooks, it is very likely that the preceding and following sections of those textbooks are relevant in some way.

This data about the structure of the textbooks can be used for Linking. How exactly this can be done is described in Chapter 3. Additionally, textbooks usually include an index, which carries important information about the place in the text where important

terms are first introduced. This is likely relevant for linking, but goes beyond the scope of this thesis.

Chapter 3

Approach

Before Linking can start, the textbooks are split into documents in a preprocessing step. There is one document for each entry in the hierarchical model of the textbook. This means that there is some overlap between documents. The entire content of all subchapters of chapter 1 is included in chapter 1 as well, for instance. This means that the linker is free to link parts of any resolution in one textbook to parts of any resolution in another textbook. This is useful, since textbooks do not always use consistent levels of abstraction. Some larger textbooks have subchapters that are larger than entire chapters in smaller textbooks, so it makes sense to leave the choice of resolution open to the system. Additionally, it is hard to tell in advance at which level of resolution students will use the final product of the Interlingua project most often, so it is good to keep options open.

Figure 3.1 gives an example of the hierarchy of a textbook, in which every line becomes its own document for the system (except the first line, which represents the entire textbook):

```
Probability & Statistics for Engineers & Scientists
    1 introduction to statistics and data analysis
        1.1 overview statistical inference, ...
        1.2 sampling procedures; collection of ...
        1.3 measures of location the sample mean ...
            exercises
        1.4 measures of variability
            exercises
        1.5 discrete and continuous data
        1.6 statistical modeling, scientific ...
        1.7 general types of statistical studies ...
            exercises
    2 probability
        2.1 sample space
        2.2 events
```

FIGURE 3.1: part of the hierarchical structure of a textbook.

3.1 ISI glossary

As the textbooks all belong to the area of statistics, it makes sense to use an ontology that is also specialized in statistics. The glossary of the International Statistical Institute matches our requirements perfectly and is available in multiple languages, including English, French and German[27].

Note that while the ISI glossary is rather big and exhaustive for the area of statistics (it has 3564 terms for at least one of the three languages we use), it is not a proper ontology in the sense that it doesn't have connections between its terms, such as 'gaussian distribution' being a type of 'probability distribution'.

The ISI glossary has the following form:

The glossary consists of a collection of concepts. Each concept has a list of Phrases for each language. These lists can be empty if that language does not have any phrases

for that concept, or if the glossary is simply incomplete. Each phrase in turn consists of a list of words. The phrases describing the same concept are synonyms.

3.2 Creation of the VSM

The VSM of the documents is constructed from the text of the paragraphs, plus the titles of the chapter they are in, at a boost factor. Each term in the VSM corresponds to one entry in the ISI glossary. To determine the weight that each term should get for each document in the VSM, a query for each language is constructed from the term, and the strength of the association between the query and the document is used as the weight in the VSM. The following formula is used to determine the score for a given query/document pair:

$$score(q,d) = coord(q,d) \times queryNorm(q) \times \sum_{t \in q}(tf(t,d) \times idf(t)^2 \times t.getBoost() \times norm(t,d))$$

As will be explained in detail in Chapter 4, this is implemented using the Lucene API[28] and the above is Lucene's practical scoring formula[29]. Essentially, this is a standard tf-idf formula, plus additional factors to take the lengths of documents and queries into account. $coord(q,d)$ is a score factor based on how many of the query terms are found in the specified document. $queryNorm(q)$ is what ensures that different queries are comparable. $t.getBoost()$ is not used by this application. $norm(t,d)$ is a normative factor that depends on the lengths of the term and the document. See the reference for details on these factors. Note that the body of each document (i.e. the text of the paragraph) and its title are searched separately and their scores are then combined after re-weighting.

For the purpose of calculating the inverse-document-frequency, each language is considered independently. This ensures that if a term is rare in the documents of one language but appears frequently in other languages, that term has a higher IDF in the first language than in the other two. This is useful for our linking task. For example, assume that we want to link a document in English to one in French, and that there is a term that is rare in both the English and the French corpus, but appears frequently in

the German corpus. In this case, we would want the term to be considered important, since it is so rare in both corpora we are trying to link. As the corpora are treated as independent for the purpose of calculating the document frequency, the term gets a high inverse document frequency in both the English and French corpora and is not affected by the German corpus, just as needed. Since the quality of the glossary can not be guaranteed to be the same for every term in every language, something like this scenario can occur more frequently than one might expect.

Each term in the ISI glossary has a list of synonymous phrases for each language that represent the term. For instance, the term 'arithmetic mean' has the following synonymous phrases in English: "arithmetic mean", "arithmetic average", "mean", "average".

There are several ways to parse this information and to construct queries from these phrases, depending on how precise and restrictive the search is supposed to be. Three different ways of interpreting the phrases are used for constructing the VSM, and these can be given different relative weights. This means that the VSM may have not one but up to three terms for every entry in the glossary. The following three interpretations are used:

- FindPhrase: Each phrase is considered a match if its words occur in exactly the given order
- FindAll: Each phrase is considered a match if every word in it occurs somewhere in the text, regardless of the order
- FindAny: Each phrase is considered a match if any of the words in it occurs

FindPhrase gives very few matches, but is very accurate, while FindAny occurs very frequently, but has many false positives. FindAll is a compromise between the two. FindAny is so liberal that it can hardly be said to reflect the term it is supposed to describe, at all. For instance, the phrase "Abbe criterion" is considered found under this interpretation even if the document contains only the word "criterion" on its own. However, it is nevertheless possible that this interpretation captures some other form of latent information that is not obvious to humans and not explicitly listed in the glossary.

The term "criterion" alone, for instance, does look like it could capture useful information as well. Only empirical results will show if this is true, but it seems likely that all three interpretations carry at least some information, and so it is useful to at least consider each of them. This is equivalent to using three different types of feature extraction.

See Chapter 5 for a table summarizing how often terms are found under each of these interpretations and how they impact performance. See Figure 3.2 for an illustration of the VSM creation.

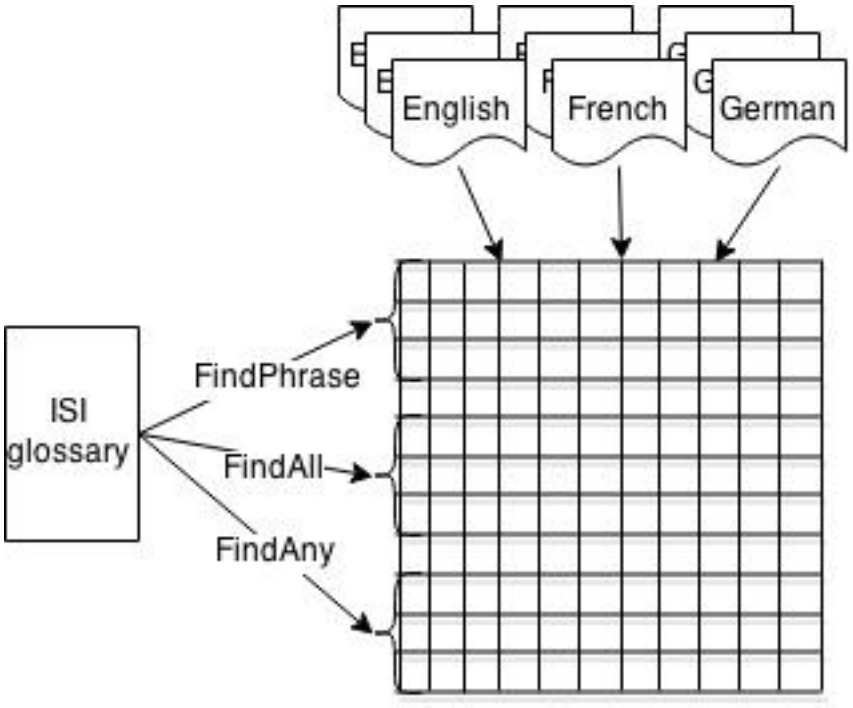

FIGURE 3.2: The VSM consists of several documents per book and three terms per concept in the glossary, one for each interpretation.

3.3 Improving the VSM

Several different kinds of transformations on the VSM are available to try to improve performance. These can be combined in any order and many of them take hyper-parameters, so it takes some experimentation to figure out the best combinations.

3.3.1 Pruning

Not all terms in the glossary are found in the corpora we have available. This also depends heavily on which of the above listed three interpretations for finding a term is used. Pruning away terms that don't appear does not change the results but speeds up the calculations.

More important is the question of what happens if one prunes away terms that do appear, but not very frequently. It may be that they are simply noise, after all, in which case removing them would increase performance. Two questions are of principal interest here:

- Does it help to prune away some of the terms with low variance?
- What happens when even more terms with low variance are pruned away? This is *not* supposed to increase performance. Instead, it is an experiment to test how our results would change if the glossary was smaller (it is likely that smaller glossaries still contain most of the important, high-variance terms, but lack some of the less frequently used, low-variance terms). This is interesting to know what to expect in the future, when there may be smaller glossaries for other scientific fields.

3.3.2 LSA

As mentioned before, LSA is a frequently used technique in NLP and can be used for dimensionality reduction. This reduces noise, which improves performance, and also speeds up subsequent operations by reducing the number of dimensions.

3.3.3 PCA

PCA is similar to LSA in that both are methods for dimensionality reduction. PCA operates on the term-covariance matrix while LSA operates on the term-document matrix directly. Thus the two optimize different things, but are still similar[30]. LSA is clearly the more popular of the two, but since they are so similar it was easy to implement PCA as well and it can't hurt to try out both.

3.3.4 Considering Order

As mentioned before, the bag-of-documents assumption does not hold for our data set, since the documents are extracted from textbooks, which are structured hierarchically. This is very rare in MLDC, which usually deals with unordered documents like news stories.

Since the documents we compare are extracted from textbooks, there is a natural order to them. It is reasonable to assume that the documents preceding a given document influence what it should be linked to, as authors tend to follow a thread when writing a textbook and don't just order their chapters randomly.

For instance, if a chapter about a difficult topic is preceded by a similar topic, it likely builds on previous topics and is used in detail. On the other hand, if it is preceded by a variety of very dissimilar topics, then chances are that this chapter is just part of a recap. The difference is relevant for linking.

To make use of the order of documents, we extend the VSM of each document with a copy of the VSM of a neighboring document. If no neighbor exists the average VSM is used. Parameters for this transformation are the range of neighbors to consider and the relative weights of each of them.

Note that for the first paragraph of a sub-chapter, we do *not* use the last paragraph of the preceding sub-chapter as its predecessor. In other words, whenever a new chapter or sub-chapter begins, the chain is considered broken. When this happens, the document gets a dummy-predecessor instead, which is just the average of the VSM, to minimize its influence. We do this because it seems far less likely for the last paragraph of the preceding section to be relevant than a preceding paragraph from the same section. The way in which we use the hierarchical structure of the textbook (explained below) has good synergy with this and can be used to make preceding sub-chapters or chapters relevant, after all.

3.3.5 First Paragraph

Many similar MLDC papers give the first paragraph of each document a greater weight, since it tends to contain summaries and the most important information about the document. This does not make sense for us, since each document we consider is only a single paragraph in the first place.

3.4 Linking

To determine the similarity between documents, their VSM representations are compared using the cosine similarity. Figure 3.3 illustrates the whole process of creating the actual Linking that is used for evaluation.

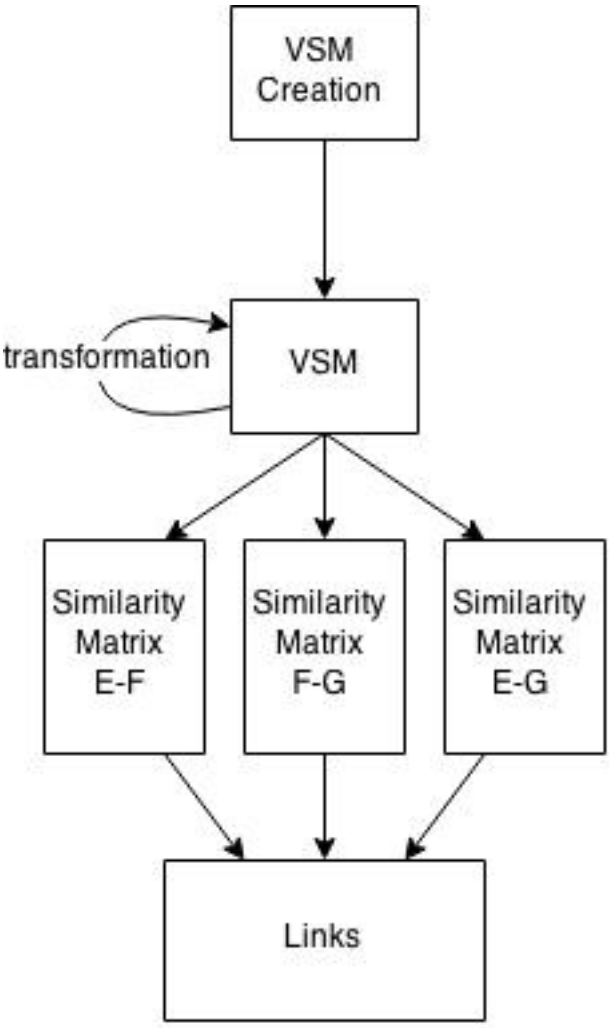

FIGURE 3.3: First, the VSM is created. It is then transformed in several ways in order to improve performance. Finally, the VSM is split into the three languages and a similarity Matrix is created for each language pair by multiplying the two appropriate parts of the VSM together and normalizing. These matrices contain the similarity value for any pair of parts of textbooks from different languages. They are used to create the final links.

3.5 Considering Structure

In addition to the order of the documents, their hierarchical structure can also be used to influence and hopefully improve the final similarity measures. Unlike the order however, this is not done by changing the VSM. Instead, this operates on the similarities directly:

Generally speaking, textbooks have a hierarchical structure, with chapters at the highest

level, followed by sub-chapters and possibly sub-sub-chapters. Each unit of the hierarchy tends to contain parts that have something in common. Therefore, the similarity between the parent units of each document make for a good baseline when comparing documents.

To give an example, it is possible that two paragraphs look like they are very similar, since they talk about the exact same concepts. However, looking at the chapters surrounding the paragraphs, it becomes apparent that they are actually talking about two very different things. The paragraphs look similar, but have different contexts. In this case, one wouldn't want them to be linked together.

This is how we implemented this idea: To get the similarity between two documents, we first get the similarity of their parents and use those as a baseline. The parents in turn can also use their own parents as a baseline, and so on recursively. Baselines can receive different weights relative to the direct comparison by raising them to a power. The higher the exponent, the more important the baseline. It remains to be seen up to what level of abstraction this is useful and how highly the baselines should be weighted for optimal results.

Note that the transformations for order and structure are similar, but not the same. Structure is only after the general theme of a chapter/sub-chapter/etc., while order is after the concrete sequence and captures change. In fact, since the order-based transformations can also be used on the structure-based VSM, the two methods can be combined. As mentioned above, there could be a lot of synergy between the two. Figure 3.4 illustrates which documents are considered neighbours for the purposes of the Order-based VSM transformation and how this relates to levels of abstraction in the hierarchy.

Note that different levels of abstraction may have different optimal parameters for the order-based transformations. For instance, it may turn out that the order is more informative on the level of chapters than on the level of paragraphs, as paragraphs have more noise than chapters. This may make training more complicated.

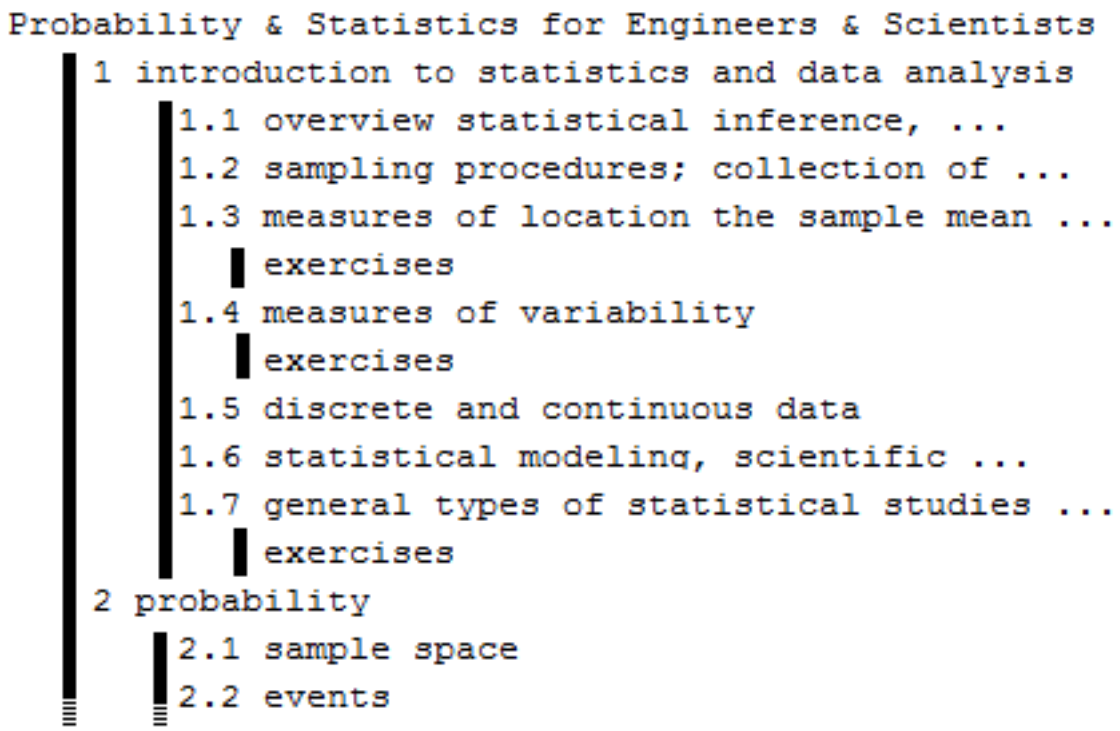

FIGURE 3.4: part of the hierarchical structure of a textbook, with neighbourhood links marked with black lines. Note that neighborhoods among subchapters extend only within the chapter containing them. Exercises, which are even lower in the hierarchy do not have any neighbours and will likely benefit a lot from using Structure, since the subchapters they refer to will then be used as their baselines.

3.6 Evaluation

For each pair of two of our three available textbooks, experts have manually created a mapping between them. Each of these consists of pairs of sections, one from each textbook, with a rating for how similar they are. Sections can refer to any level of the hierarchy, whether chapters or sub-chapters of any level. In practice of course, mappings tend to go between sections on the same level of the hierarchy.

There are a total of 128 links available in the mappings, 60 of which map parts of the English textbook to the German one, 46 match French to German, and 22 match English to French. This is relatively sparse data, which is less than ideal. While it should be enough to make rough estimates of the quality of the linking system, there is likely to be a problem with over-fitting parameters. Dealing with this problem effectively would require data for cross-validation, but since we already have so little data, splitting

off a subset for cross-validation would likely make things even worse. Fortunately, This problem will disappear once more expert-mappings become available in the future.

Evaluating the quality of a linking is complicated. Most related tasks use Mean Average Precision, the F-measure, or a precision-recall curve[31]. However, this is not sensible for us. The purpose of the linking is clearly defined: A student will be presented with the results, in ranked order, and can use this to find text in his native tongue to look at. Optimally, the first returned result is the one they are looking for. In any case, if the correct result is not in the top five or so returned results, it may take too long for the student to find what they are looking for so that they give up on using the tool. Precision-recall is more useful for unordered documents retrieval[32], but for our purposes the order is of paramount importance. For this reason we have chosen to use Normalized Discounted Cumulative Gain as our measure of quality.

The formula we use for DCG is $DCG@n = rel_1 + \sum_{i=2}^{n} \frac{rel_i}{log_2(i)}$, and NDCG@n is defined as the DCG@n value divided by the optimal possible DCG@n value that can by achieved by reordering the results. An important aspect here is the value of n. Lower values are more realistic, since a human user will rarely look at low search results in practice. Higher values, on the other hand, effectively give the system more feedback to work with. Since our data is already quite sparse, this is important. We compromised on setting n=10. As there are three different expert mappings available, the final evaluation value is the average of the NDCG@10 values for all three of them.

Chapter 4

Implementation

The textbooks used as input are available to us as PDF files and have been preprocessed by another part of the Interlingua project. Similarly, the ISI glossary is also parsed and preprocessed by another part of the project.

4.1 Lucene

Rather than writing our own methods for creating the VSM from the text data, we use the existing text search engine Apache Lucene, Version 4.10.2[28].

4.1.1 Indexing

Lucene provides standard analyzers for each language we need that come with a pre-implemented stop-word list and stemming rules. Stemming is used instead of the more complex lemmatization, but the difference in performance should not be important. Lucene is usually used for tasks with greater accuracy requirements than ours, as it is primarily intended for searching for short queries, where less data is available than when comparing large texts. It therefore stands to reason that the accuracy of Lucene's stemming will be sufficient for our task as well. This is supported by previous similar experiments[13].

As our task operates on the narrow domain of statistics, it could be useful to use a

custom stop-word list. However, since we are only interested in the predefined phrases in our glossary and those don't seem to contain unusual stop-words, we decided against using a custom list and went with the default built into Lucene.

4.1.2 Creation of the VSM

The VSM is constructed by creating a query from each entry in the glossary, running the query on the corpus, and reporting what weight Lucene assigns each document for each query. These weights are determined through Lucene's scoring formula, as described in the previous chapter.

The three different interpretations of the glossary mentioned in Chapter 3 are implemented as different ways of constructing a query from a glossary entry. All three are boolean queries:

- FindPhrase: Disjunction over the synonymous phrases: PhraseQuery for the phrase
- FindAll: Disjunction over the synonymous phrases: Conjunction over the words in the phrase
- FindAny: Disjunction over the synonymous phrases: Disjunction over the words in the phrase

4.2 improving the VSM

The improvements on the VSM are implemented as a modular system of transformations. Each transformation takes parameters and they can be applied in arbitrary order. This adds flexibility and makes it easy to extend the system in the future.

Some transformations remove or create terms of the VSM. This could potentially cause problems when transformations interfere with each others' inputs. To prevent this, the terms of the VSM are named after their origin and each transformation can use a TermFilter to operate on subsets of the available terms.

4.2.1 Transformations

The following transformation are used:

- Pruning: removes all terms that match the TermFilter
- LSA & PCA: performs LSA/PCA on the terms that match the TermFilter, reducing them to the specified number of dimensions
- Neighbor: used to make use of the order of documents. Goes a specified number of steps forwards or backwards in the list of documents and adds a copy of all the terms that match the TermFilter to the original document, at a specified relative weight. If there is no neighboring document at the specified distance, the average VSM is used instead.

Note that the use of the Structure of the textbook, unlike the use of the Order, is not implemented as a transformation on the VSM. Instead, it influences the linking by acting directly on the similarity matrix.

4.2.2 TermFilters

The following TermFilters are used:

- All: get all terms
- Unused: get all terms that don't appear in any document. Used for pruning to speed things up.
- Low variance: get a specified number of terms with the lowest variance. Used for experiments with pruning.
- Glossary: get the terms that were extracted using the specified interpretation of the glossary. Useful if different interpretations of the glossary are useful for different roles.
- Transformation: get the terms that were added by the specified transformation. This can be used to chain transformations after one another without interfering with other code.

Chapter 5

Results

5.1 The Glossary in all Three Languages

Table 5.1 lists the number of terms in the glossary that appear anywhere in the text for each of the three interpretations mentioned before. It lists how many appear in all languages, how many are missing in at least one language, and how many are not found at all. It also lists the average number of terms per document found for each language. Keep in mind that there is one textbook per language.

3561 terms in total	FindPhrase	FindAll	FindAny
Found in all languages	120	307	2330
Found in English	523	1785	3345
Found in French	298	682	3177
Found in German	365	761	2511
Not found in any language	2823	1512	117
Terms/document in English	47.6	160.8	1638.8
Terms/document in French	32.6	52.4	1030.3
Terms/document in German	21.2	35.0	513.8

TABLE 5.1: Analysis of glossary term appearances

As one can see from this table, using the FindPhrase interpretation yields only a small number of matches per document, and many terms are not found at all. On the other hand, terms under the FindAny interpretation are found almost everywhere. However, manual analysis shows that the results of the latter interpretation are often inaccurate, as expected. The differences between languages can be explained by the differences in

size of the three textbooks and by small particularities of the languages. They do not seem to be significant. Therefore, the glossary appears to map well to all three languages, although the English terms are noticeably more complete than the other two languages.

Note that some terms in the glossary do not have translations into French or German. 99 Terms are undefined for French and 72 for German, which makes it questionable if they will be useful for linking. 46 Are only defined for English, and are therefore useless. These terms are pruned away in a preprocessing step.

5.2 Choosing the Glossary Interpretations

As a first step, we will find a good way of constructing the initial VSM from our three available glossary interpretations. This baseline will then be improved iteratively by applying one improvement after another. Note therefore that the "baseline" will always refer to the optimum from the last experiment, since it is built consecutively in multiple steps.

Since there are three different ways to parse the data, the question arises with what relative weight these three techniques should be combined. To do this, we first determine which of the three is best when used on its own, then test what happens when the next best one is added with different weights relative to the first one. Once we have an optimal way to combine the first two, we proceed in the same way to get the best relative weight of the third parser. Lastly, we find a good way of weighting document titles relative to document bodies.

Figure 5.1 shows the relative performance of the three interpretations of the glossary terms when they are used alone. It turns out that the broader interpretations are more effective than the stricter ones. This is an interesting result, because it means that the expert information encoded in the glossary is not as useful as the more numerous but less strict information one gets when only interpreting the phrases loosely. It seems that quantity beats quality in this case.

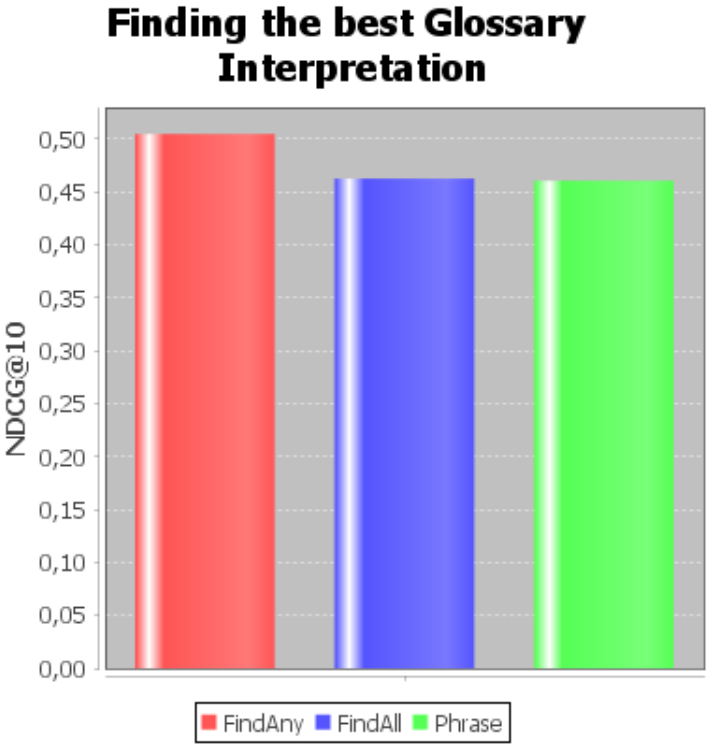

FIGURE 5.1: The performance of each of the three possible glossary interpretations when used separately.

Figure 5.2 shows what happens when the FindAny interpretation, which is best when measured on its own, is combined at different relative weights with the FindAll interpretation, which is the second best. A small gain is possible, reaching its optimum at a relative weight of about 1.45. Notably, this is larger than the weight of FindAny (1.0), even though FindAny on its own is better than FindAll. The reason for this is that FindAny is superior mostly due to its higher number of terms. The FindAll interpretation yields fewer non-zero values, but those it does find are more important and therefore work better when they receive a higher weight.

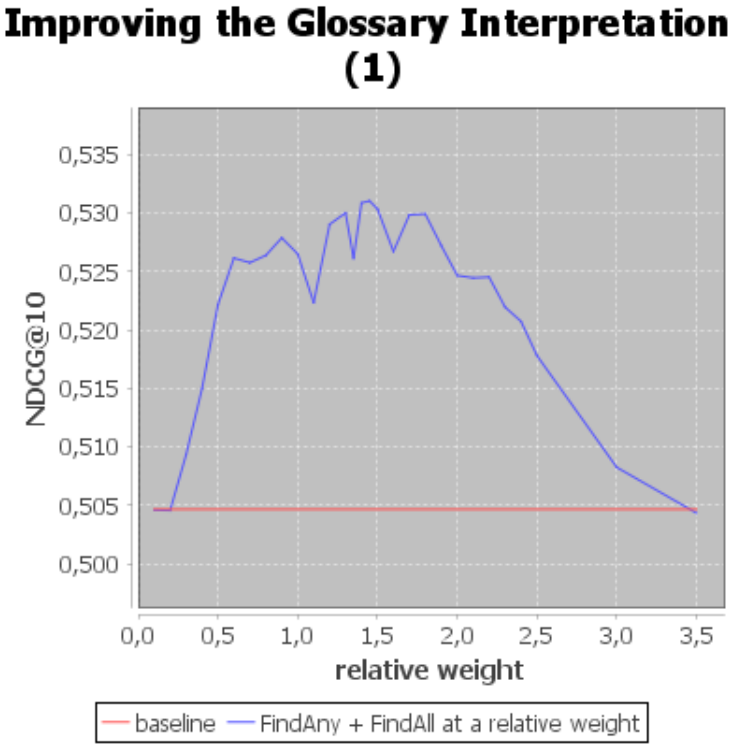

FIGURE 5.2: Uses the FindAny interpretation at weight 1.0 and iterates over the weight of the FindAll interpretation, combining both of their results.

Figure 5.3 shows when the last and most strict interpretation, FindPhrase, is added at different weights. The gain is again very small and the optimum is reached at a relative weight of 1.8. The reason for this high weight is again the same as before.

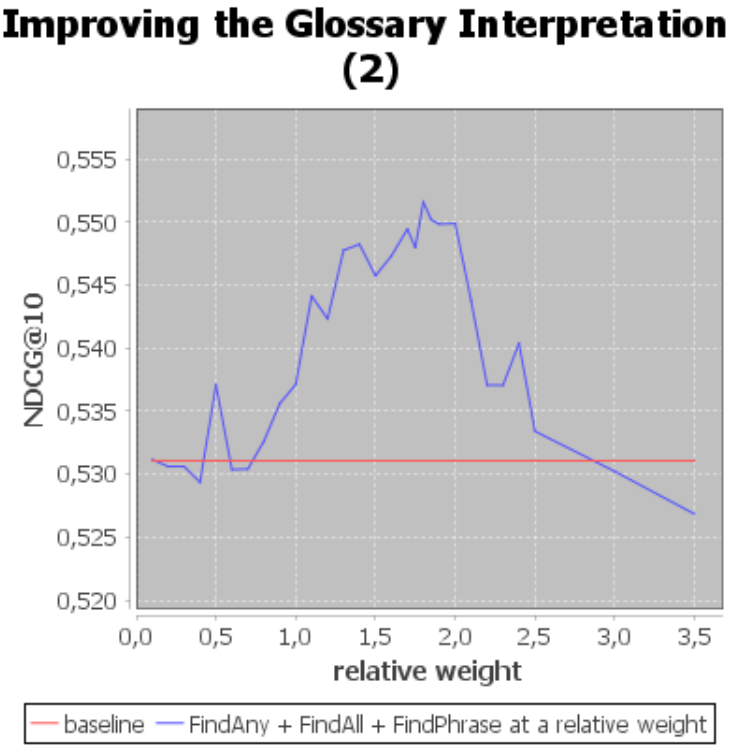

FIGURE 5.3: Uses the FindAny interpretation at weight 1.0 and the FindAll intepretation at weight 1.45 and iterates over the weight of the FindPhrase interpretation, combining all three of their results.

So far, all experiments have considered only the body of each document. It is possible that their titles also carry useful information. Figure 5.4 analyzes this possibility.

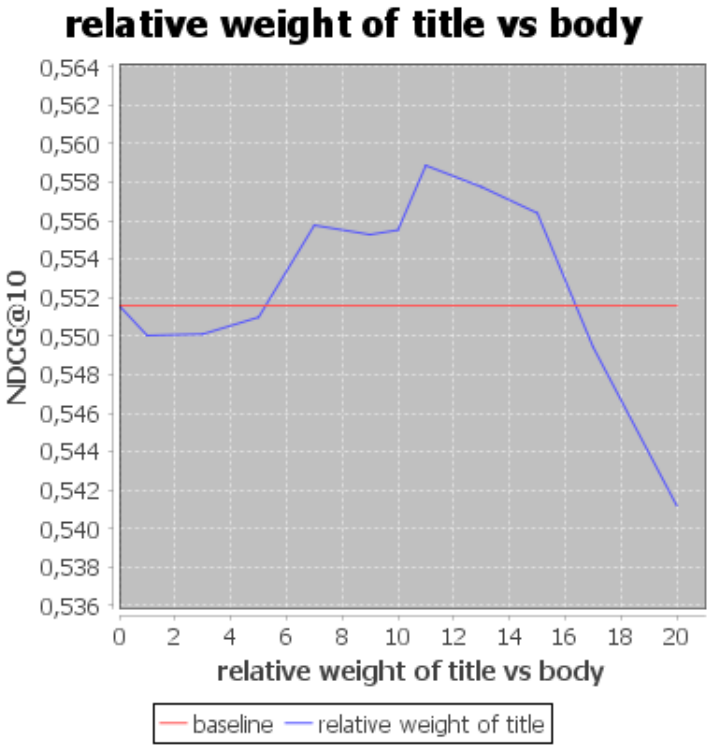

FIGURE 5.4: Uses the text of the document titles in a addition to the text of the document's body, at an increased weight.

There is only a marginal improvement, even at the optimum of weighting the title eleven times as much as the body, and a dropoff in performance follows beyond that. It seems reasonable to assume that titles become more useful if only the Phrase interpretation is used on them, since titles are the one place of a textbook where one would most expect technical terms to turn up verbatim. Unfortunately, it turns out that at least the textbooks we are looking very rarely have Glossary terms in their title word for word, and when they do, the term already appears in the text very often anyway. It actually makes no measurable difference to use them at all for any reasonable weight when using the FindPhrase interpretation alone.

5.3 Shrinking the Glossary

We found a good multilingual glossary for our tests. This may become harder in other areas of science. It is therefore useful to have a look at the importance of having a

comprehensive glossary. Less effective glossaries can easily be simulated by removing existing terms from our glossary and seeing how that affects performance. It is likely that poor glossaries would nevertheless contain most of the basic terms of their area and only lack some of the more advanced ones. To simulate this, we decided to remove terms from our glossary in ascending order of their variance. This is illustrated in Figure 5.5.

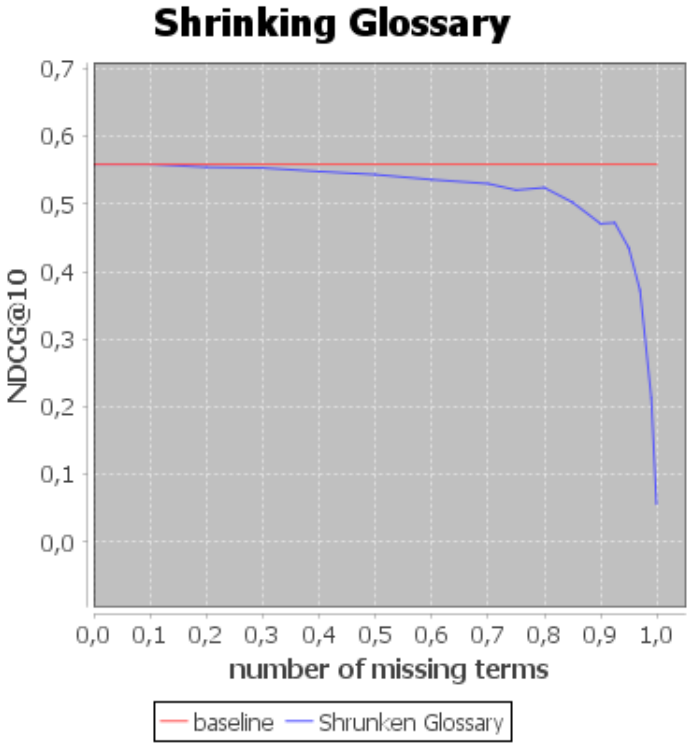

FIGURE 5.5: The performance of the system after removing the given percentage of the terms with lowest variance. The terms that do not appear at all have already been removed in a preprocessing step and are not considered here.

It turns out that most of the performance loss only happens after about half of the terms have been removed. This suggests that most of the information relevant for linking lies in the most important terms. Interestingly, the curve has local maxima in the second half. This means that some terms are actually harmful for the quality of the linking. However, this could just as well be an artifact of noise, since our data set is rather small.

Note also that the removal of the terms with lowest variance had almost no impact on performance. In particular, it did not improve performance to remove the terms with the very lowest amount of variance. This suggests that even these rare terms carry valuable information and are not very noisy.

5.4 Improving Results

After looking at an experiment about performance loss, we now come back to optimizing our results.

5.4.1 Dimensionality Reduction

There are two different ways of performing dimensionality reduction, LSA and PCA. Additionally, there are several alternative interpretations of LSA, so we opted to just test them all. Figure 5.6 shows their performances relative to the baseline.

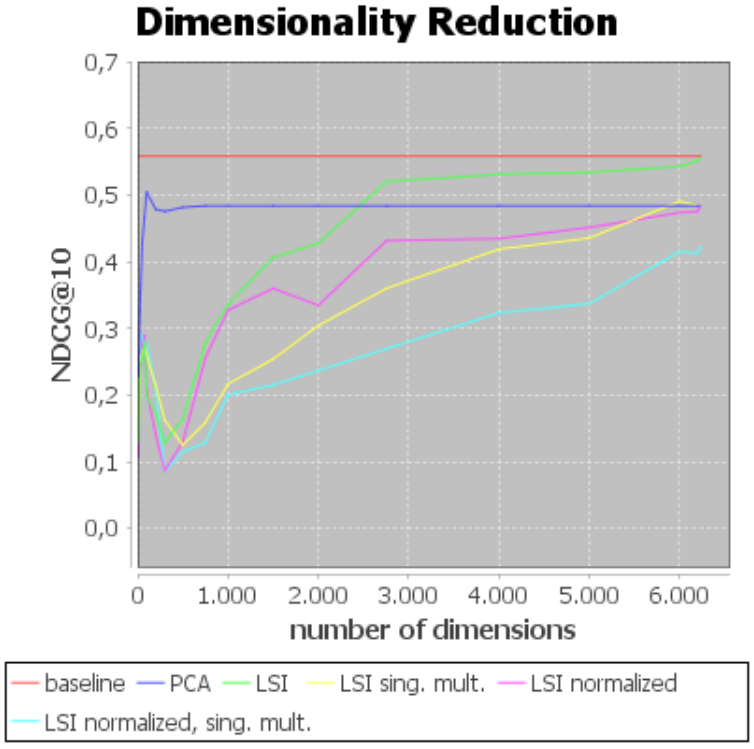

FIGURE 5.6: The baseline compared to PCA and four different interpretations of LSA: the term-document matrix may or may not have its mean normalized before LSA and the result of LSA may or may not be multiplied with the singular values again.

Surprisingly, they all actually reduce performance. There are several possible explanations for this:

- Our data set is too small. Normally, there should be far more than three textbooks worth of information to train with. This makes the analysis less accurate than it ordinarily would be. A good rule of thumb is that every correlation one wants

to capture should occur at least once in the training set. Since we have only one textbook per language, and no textbook is really comprehensive enough to cover the entirety of Statistics, this is not the case for our data. In the case of PCA, the fact that there are fewer documents (sections of textbooks) than glossary terms makes the compression lossy, which is also why the curve for PCA flattens off.

- LSA is used here for data compression, which is not the primary use of the technique for multilingual document collections.
- The way LSA and PCA are normally used, it operates on raw words, which are rather noisy. However, we have ontology terms, which are already designed to be very information dense and to concisely describe the subject. Since we have only few documents, it is likely that the dimensions found by LSA/PCA are just artifacts of chance and therefore less useful than the already very informative basis they operated on.

It may be relevant to note that there are very few papers that combine ontologies with dimensionality reduction techniques, as we did. It is possible that this is already a known negative result.

5.4.2 Order

What happens when one uses the order of the documents to influence the mapping? This is illustrated in Figure 5.7.

As expected, using the predecessor of a document as an additional factor improves results. This improvement is strongest at a relative weight of 0.7. Unfortunately, predecessors that are further away as well as succeeding documents are not helpful. This is not unexpected from looking at the titles of the chapters of the textbooks: While topics do tend to follow a thread, there are too many parallel threads that can be followed and too many sections that can be skipped.

It is likely that using predecessors at greater distances, as well as successors, will become more useful in the future, as more textbooks become available and general trends become visible through the noise. After all, we only have three textbooks, which means three separate authors thinking up a thread to cover all of Statistics. This means that

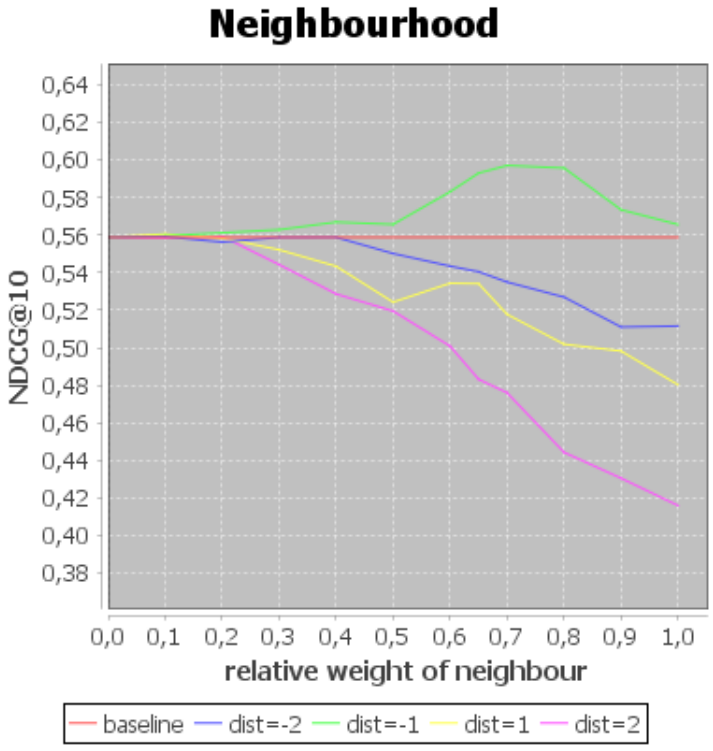

FIGURE 5.7: Conflates a document's VSM with that of a neighbouring document, at a relative weight. The value 'dist' gives the position of the neighbour relative to the document we are looking at (e.g. 'dist=-1' refers to the immediate predecessor). Note that neighbourhood relations are defined as illustrated in Chapter 3.

the size of our dataset for the purposes of using Order effectively is only the number of textbooks, not the number of textbook sections. Therefore, we should make improvements quickly once a couple dozen textbooks are available and it is easier to discern underlying similarities from authorial bias.

5.4.3 Structure

How useful are the parent elements for mapping? When linking a subchapter, how important should the links of its parent chapter be? The results are illustrated in Figure 5.8.

It appears that letting the parent documents influence the linking also leads to improvements, just as it helped to use preceding documents. The optimal weight here is 1.0, which means that the similarities of the parent document should be given the same importance as the document we are actually comparing.

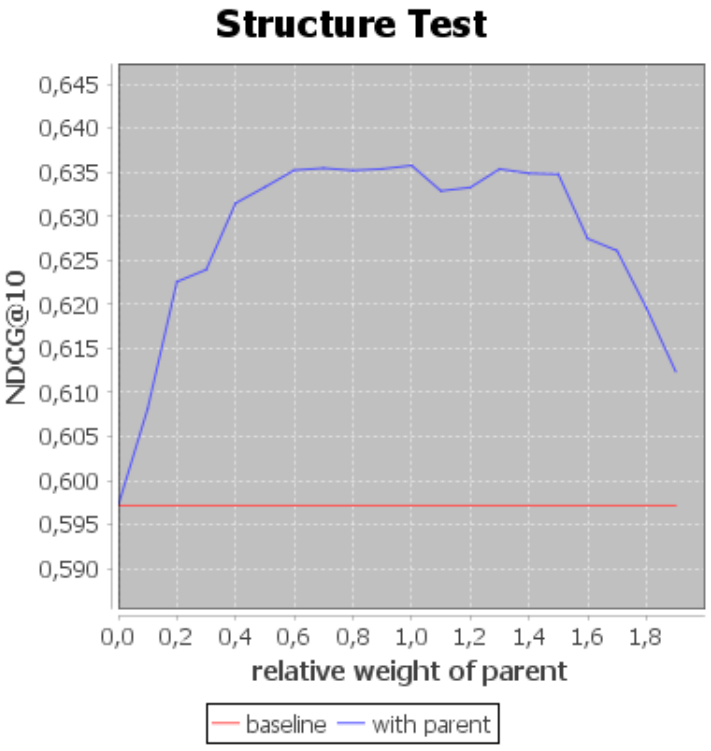

FIGURE 5.8: The performance when the similarities of the document's parent in the textbook's hierarchy are used as an additional factor, with a given weight relative to the similarities of the document itself.

Additionally, Figure 5.9 analyzes what happens when the preceding document of the parent is also considered. After all, since the predecessor is useful, and the parent is useful, it might turn out that the predecessor of the parent is also useful. Unfortunately, this does not work and just makes the results worse. The reason for this is probably simply that the predecessor of the parent is too far removed to be helpful and just introduces noise.

It should be noted that these results can probably be improved once the structure of textbooks can be compared more consistently: Manual analysis of the hierarchies created out of the textbooks showed that the German textbook's mapping is inconsistent with the other two, which causes the program to think that sub-chapters in one textbook correspond to chapters in the other. Unfortunately, since authors number their chapters inconsistently, fixing this problem would require a way to automatically determine what level of resolution a unit of text in a textbook has, which is beyond the scope of this thesis.

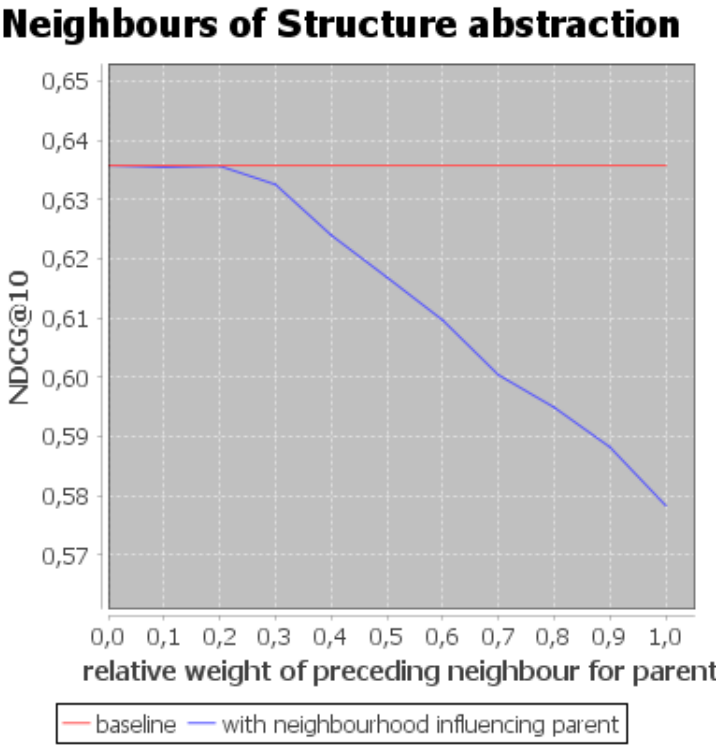

FIGURE 5.9: Using the document's parent at a relative weight of 1.0, as determined to be optimal above, what happens when the parent is influenced by its own predecessor at a given weight? This experiment combines Order and Structure.

5.4.4 Summary of Results

Table 5.2 summarizes the results as we iteratively improved our baseline. Note that all improvements have around the same size of 0.03. This is disappointingly little on its own, but the improvements do add up to a total boost of about 28% over the baseline of just using the best glossary interpretation on its own.

technique	NDCG@10
worst glossary interpretation: FindPhrase	0.46
second best glossary interpretation: FindAll	0.46
best glossary interpretation: FindAny	0.50
added FindAll	0.53
added FindPhrase	0.55
adding document title	0.56
dimensionality reduction	no improvements
Order (predecessor)	0.60
Order (other neighbours)	no improvements
Structure (parent)	0.64
Structure (parent's predecessor)	no improvements

Table 5.2: Final evaluation results

Chapter 6

Conclusion and Future Work

6.1 Conclusion

The task of linking textbooks in different languages has not been attempted before, as far as we could find. While multilingual document clustering is very similar, it is not usually done on documents that have a structure with respect to each other. This is a defining feature of textbooks, however.

There are two primary ways of performing MLDC. If one has a parallel corpus, one can use LSA or another dimensionality reduction technique. If one has a multilingual ontology, one can use that instead to map documents of different languages to the same VSM directly. We have chosen to use the latter approach.

It turned out that there is some ambiguity in how the ontology terms are to be interpreted, and that this seemingly simple decision actually influences performance a lot.

Lastly, we found that it is useful to take both the order and structure of the documents into account.

6.2 Future Work

There are several ways to improve results further. Since the Interlingua project is ongoing, some of these will likely be implemented in the near future. The following are some of the most interesting ways of improving the VSM:

- Use the index of the textbooks to determine where in the textbook a term first occurs. When index terms match ontology terms, this can be used to find the point where ontology terms first get introduced, which likely carries useful information for linking. This feature is currently being implemented.
- Experiment with different ways of splitting textbooks: Treat each paragraph as a separate document or even determine document boundaries dynamically based on term occurrences.
- Automatically determine the level of resolution of parts of textbooks so that hierarchies become more consistent. This would make using the structure of the hierarchy more effective.
- The ISI Glossary can be enriched and turned into a proper Ontology with connections between terms. This information can be used to improve the VSM.

In addition to these, the system could also be improved in other ways that do not change the VSM directly:

- Gain more training data, either by getting access rights to more textbooks or by using Wikipedia or DBpedia as a free source of documents (these could have lower quality, but might make up for that with their sheer quantity).
- Implement online learning by letting the system use the final decisions of its end-users as additional feedback data.
- Implement an adaptive retrieval mechanism that ensures that the presented matches are as diverse as possible, then narrow the search down once the user makes a preliminary decision.

Bibliography

[1] Julio Guerra, Sergey Sosnovsky, and Peter Brusilovsky. When one textbook is not enough: Linking multiple textbooks using probabilistic topic models. In *Scaling up Learning for Sustained Impact*, pages 125–138. Springer, 2013.

[2] the interlingua project, as described by interreg iv-a, 2015. URL http://www.interreg-4agr.eu/de/projet-detail.php.

[3] Ronald E Walpole, Raymond H Myers, Sharon L Myers, and Keying Ye. *Probability and statistics for engineers and scientists*. Macmillan New York, 9 edition, 1993. ISBN 978-0321629111.

[4] Y Velenik. Probabilités et statistique. *Université de Geneve*, 2011.

[5] Ludwig Fahrmeir, Rita Künstler, Iris Pigeot, and Gerhard Tutz. *Statistik*. Springer-Verlag, 2007. ISBN 978-3642019388.

[6] Thomas K Landauer and Michael L Littman. Fully automatic cross-language document retrieval using latent semantic indexing. 1990.

[7] Hsin-Hsi Chen and Chuan-Jie Lin. A multilingual news summarizer. In *Proceedings of the 18th conference on Computational linguistics-Volume 1*, pages 159–165. Association for Computational Linguistics, 2000.

[8] Chih-Ping Wei, Christopher C Yang, and Chia-Min Lin. A latent semantic indexing-based approach to multilingual document clustering. *Decision Support Systems*, 45 (3):606–620, 2008.

[9] Douglas W Oard and Bonnie J Dorr. A survey of multilingual text retrieval. 1998.

[10] Bruno Pouliquen, Ralf Steinberger, Camelia Ignat, Emilia Käsper, and Irina Temnikova. Multilingual and cross-lingual news topic tracking. In *Proceedings of the*

20th international conference on Computational Linguistics, page 959. Association for Computational Linguistics, 2004.

[11] the google translation api, 2015. URL https://cloud.google.com/translate/docs.

[12] the matecat translation api, 2015. URL http://www.matecat.com/machine-translation-software/.

[13] Bruno Pouliquen, Ralf Steinberger, and Camelia Ignat. Automatic annotation of multilingual text collections with a conceptual thesaurus. *arXiv preprint cs/0609059*, 2006.

[14] Bernardo Magnini and Manuela Speranza. Merging global and specialized linguistic ontologies. *Proceedings of Ontolex 2002*, 4348, 2002.

[15] Deborah L McGuinness, Frank Van Harmelen, et al. Owl web ontology language overview. *W3C recommendation*, 10(10):2004, 2004.

[16] Paulo CG Costa and Kathryn B Laskey. Pr-owl: A framework for probabilistic ontologies. *Frontiers in Artificial Intelligence and Applications*, 150:237, 2006.

[17] Zhongli Ding and Yun Peng. A probabilistic extension to ontology language owl. In *System Sciences, 2004. Proceedings of the 37th Annual Hawaii international conference on*, pages 10–pp. IEEE, 2004.

[18] Thomas K Landauer, Peter W Foltz, and Darrell Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284, 1998.

[19] S Dumais, G Furnas, T Landauer, S Deerwester, S Deerwester, et al. Latent semantic indexing. In *Proceedings of the Text Retrieval Conference*, 1995.

[20] Susan T Dumais. Latent semantic analysis. *Annual review of information science and technology*, 38(1):188–230, 2004.

[21] Hongyu Chen, Bronwen Martin, Caitlin M Daimon, and Stuart Maudsley. Effective use of latent semantic indexing and computational linguistics in biological and biomedical applications. *Frontiers in physiology*, 4, 2013.

[22] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[23] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[24] the website of the cyc project, 2015. URL http://www.cyc.com/.

[25] Nathalie Friburger, Denis Maurel, and A Giacometti. Textual similarity based on proper names. In *Proc. of the workshop Mathematical/Formal Methods in Information Retrieval*, pages 155–167, 2002.

[26] Ralf Steinberger, Bruno Pouliquen, Mijail Kabadjov, and Erik Van der Goot. Jrc-names: A freely available, highly multilingual named entity resource. *arXiv preprint arXiv:1309.6162*, 2013.

[27] the website of the cyc project, 2015. URL http://www.isi-web.org/glossary.

[28] the website of the lucene api, 2015. URL https://lucene.apache.org.

[29] the scoring formula used by lucene, 2015. URL https://lucene.apache.org/core/4_10_2/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html.

[30] Michael W. Trosset. Trading spaces: Measuring of document proximity and methods for embedding them. Document Space Workshop, Institute of Pure and Applied Mathematics (IPAM) at UCLA, CA, 2006.

[31] John Makhoul, Francis Kubala, Richard Schwartz, Ralph Weischedel, et al. Performance measures for information extraction. In *Proceedings of DARPA broadcast news workshop*, pages 249–252, 1999.

[32] Hinrich Schtze Christopher D. Manning, Prabhakar Raghavan. *Introduction to Information Retrieval*. Cambridge University Press, 2008. URL http://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-ranked-retrieval-results-1.html.